



Desarrollo Seguro usando OWASP

OWASP

Fabio Cerullo
Comité Global de Educación

Copyright © The OWASP Foundation
Permission is granted to copy, distribute and/or modify this document
under the terms of the OWASP License.

The OWASP Foundation
<http://www.owasp.org>

Quien soy?

- CEO & Fundador de Cyclic Limited
- 10+ años de experiencia en seguridad informática trabajando en sectores de Tecnología, Manufactura, Banca y Gobierno.
- Ingeniero en Informática.
- CISSP y CSSLP de ISC2
- Comité Global de Educación OWASP
- Líder del capítulo OWASP Irlanda



Agenda

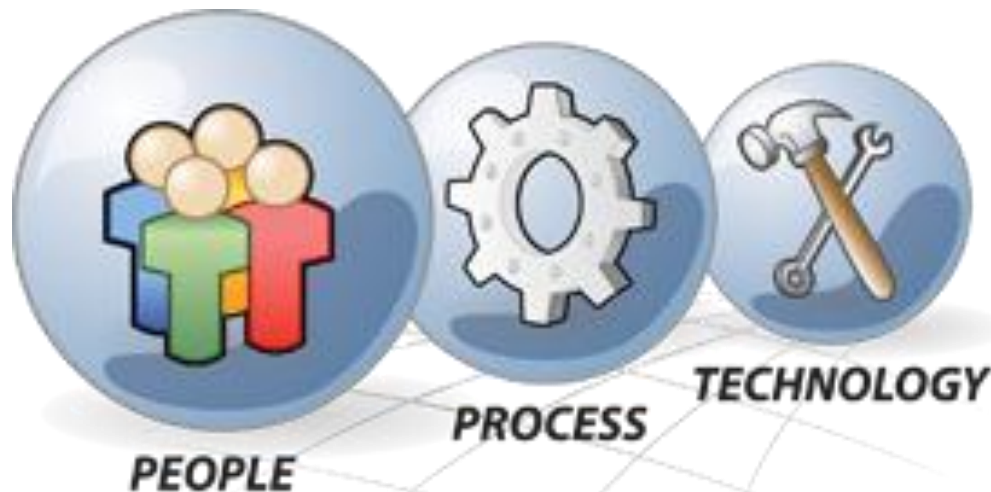
- Introducción a Seguridad de Aplicaciones
- Construyendo software seguro
- Un enfoque por fases
- Conclusiones
- Q & A

Introducción a Seguridad de Aplicaciones



Que es la **IN**Seguridad
de Aplicaciones?

Seguridad de Aplicaciones: Definición

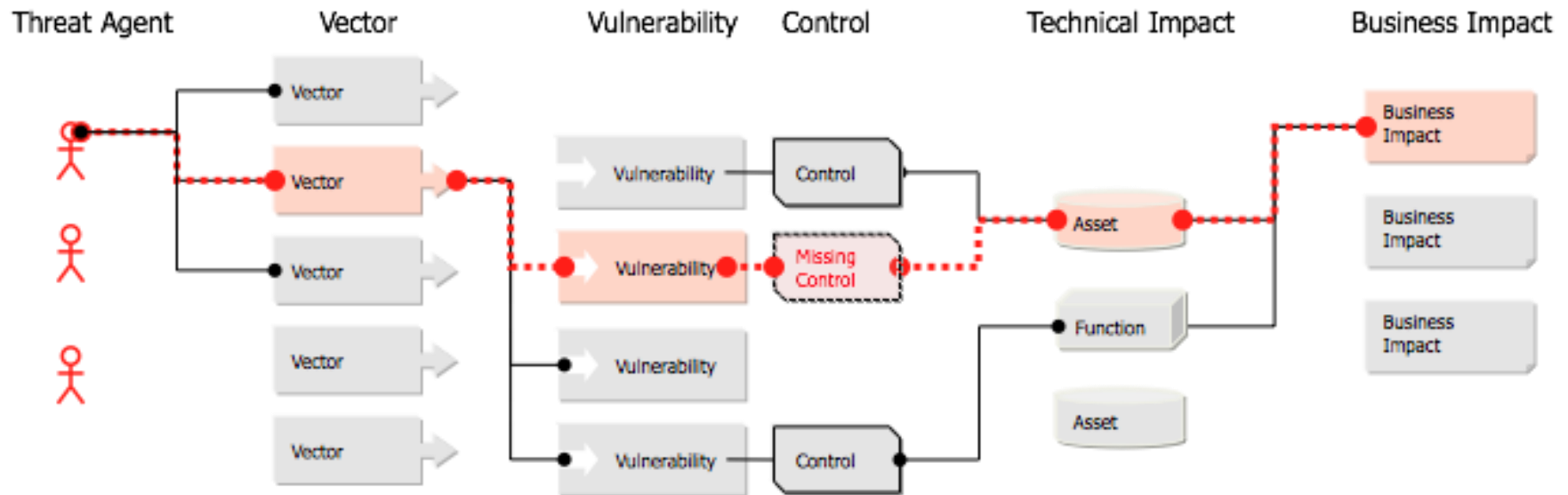


Un enfoque basado en principios involucra:

- Conocer las amenazas (Gente)
- Incorporar Seguridad en el SDLC (Procesos)
- Asegurar la red, host y **APLICACION** (Tecnología)

Introducción a Seguridad de Aplicaciones

Teoría de Riesgo



Introducción a Seguridad de Aplicaciones

Teoría de Vulnerabilidad

Cada vulnerabilidad se origina por:

Control Faltante

- Falta de validación de entradas
- Control de Acceso inexistente

ESAPI PUEDE AYUDAR AQUI

Control Roto

- Gestión de Sesiones Inapropiada
- Falla Abierta

Control Ignorado

- Encriptación Inexistente
- Desconocimiento de codificación de salidas

NADIE TE PUEDE AYUDAR AQUI



Introducción a Seguridad de Aplicaciones

TOP 10 RIESGOS DE SEGURIDAD EN APLICACIONES WEB

A1: Injection

A2: Cross Site Scripting (XSS)

A3: Broken Authentication and Session Management

A4: Insecure Direct Object References

A5: Cross Site Request Forgery (CSRF)

A6: Security Misconfiguration

A7: Failure to Restrict URL Access

A8: Unvalidated Redirects and Forwards

A9: Insecure Cryptographic Storage

A10: Insufficient Transport Layer Protection

http://www.owasp.org/index.php/Top_10

Riesgos en Seguridad de Aplicaciones

A1. Inyección

DEFINICION

Las fallas de inyección ocurren cuando una aplicación envía datos “no confiables” a un interprete.

EJEMPLO: SONY MUSIC JAPAN (MAYO 2011)

SQLi #1: <http://www.sonymusic.co.jp/bv/cro-magnons/track.php?item=7419> ‘

Un error SQL implica automáticamente una aplicación vulnerable a SQL

<http://www.sonymusic.co.jp/bv/cro-magnons/track.php?item=7419> union all select
1,concat(user,0x3a,pass,0x3a,email) from users // what we get here is user:pass:email from
table users. (0x3a is hex value for colon)

IMPACTO: SONY PICTURES (JUNIO 2011)

Mas de 1,000,000 contraseñas de usuarios, direcciones de correo electrónico, domicilios, fechas de nacimiento, además de las credenciales de administradores fueron comprometidas.

COMO PREVENIRLO?

- Evitar el uso del interprete utilizando procedimientos almacenados o consultas parametrizadas.
- Escapar caracteres especiales utilizando API's como OWASP ESAPI.
- Realizar validación positiva o “whitelisting” con adecuada canonicalizacion.

Riesgos en Seguridad de Aplicaciones

A2. Secuencia de Comandos en Sitios Cruzados

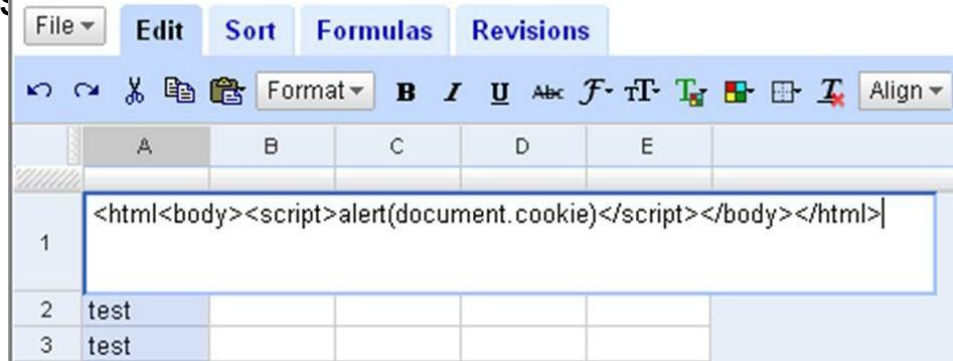
(XSS) EXPOSICION

Las fallas XSS ocurren cuando una aplicación incluye datos suministrados por el usuario en una pagina enviada a la aplicación sin validar adecuadamente el contenido. Existen tres tipos de fallas XSS:

1) Almacenado, 2) Reflejado, 3) XSS basado en DOM

EJEMPLO

- Google XSS en spreadsheets.google.com permite secuestro de sesión en todos los dominios



COMO PREVENIRLO?

- Escapar caracteres especiales utilizando API's como OWASP ESAPI.
- Realizar validacion positiva utilizando API's como OWASP ESAPI.
- HTTPOnly Cookie Flag

Riesgos en Seguridad de Aplicaciones

A3. Pérdida de Autenticación y Gestión de Sesiones

DEFINICION

Perdidas o fallas en las funciones de autenticación o gestión de sesiones (ej., cuentas expuestas, contraseñas, ID de sesión) debido a funciones customizadas.

EJEMPLO

Fijación de Sesión encontrado en el banco Mandiri (Mayor banco de Indonesia)

1) El atacante envía un mail con el siguiente enlace:

<https://ib.bankmandiri.co.id/retail/Login.do?action=form&JSESSIONID=JHAb6Q3Q1BGE5uCwNMfTDU1yxfxV9vhMODrP0krLdbem8FvqPA7I!56845468>

- dominio correcto
- URL valida
- Utilizando https

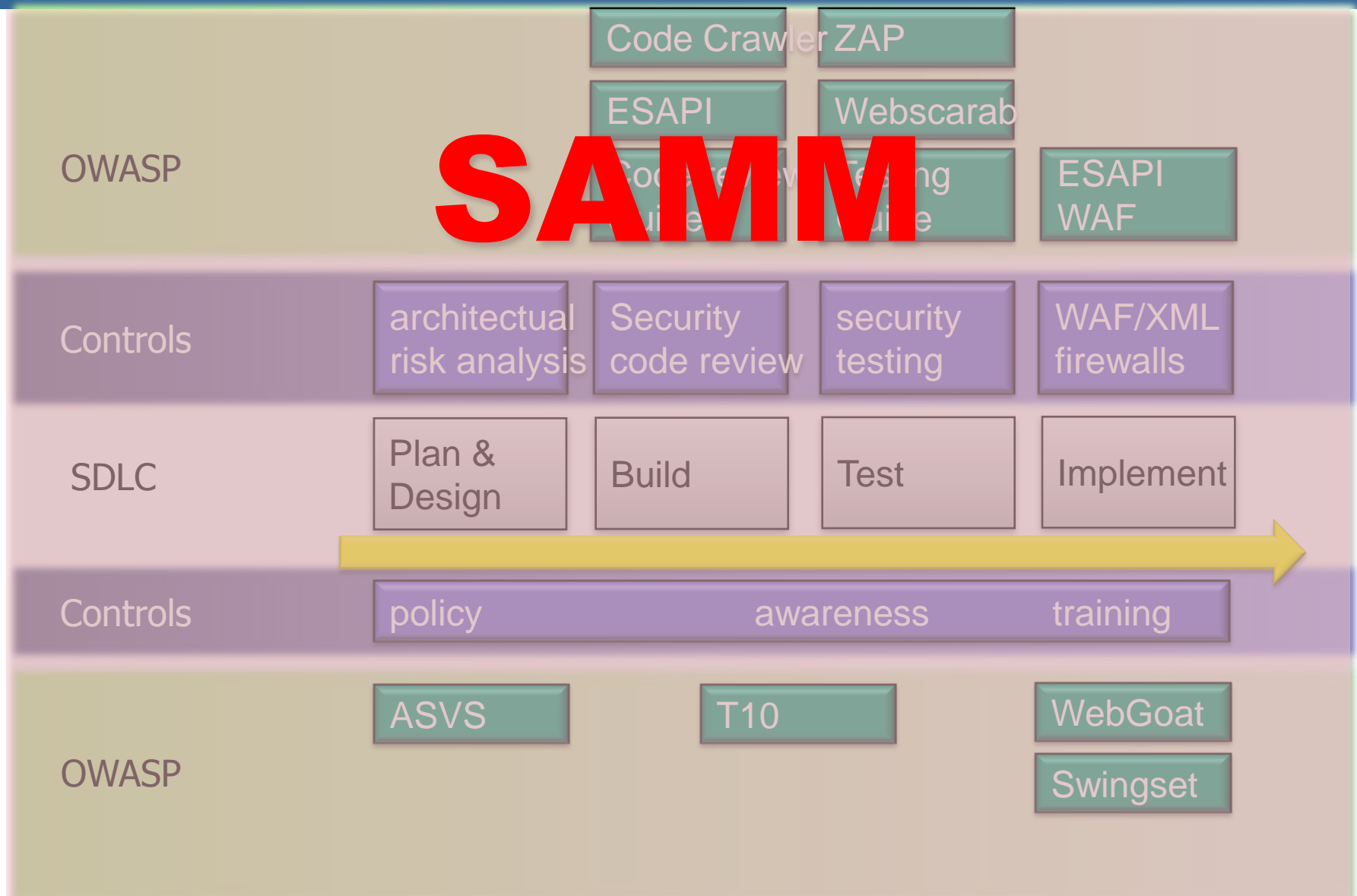
2) Victima hace clic en el link validando el SESSIONID

3) Atacante tiene acceso al Online Banking de la victima.

COMO PREVENIRLO?

- Cumplir con todos los requisitos del OWASP ASVS relacionados a Autenticación y Gestión de Sesiones.
- Brindar una interfase simple para los desarrolladores tal como ESAPI

Seguridad en el SDLC



Diseño Seguro – Estándares de desarrollo

ASVS puede ser utilizado para establecer un **nivel de confianza** en la seguridad de aplicaciones Web.



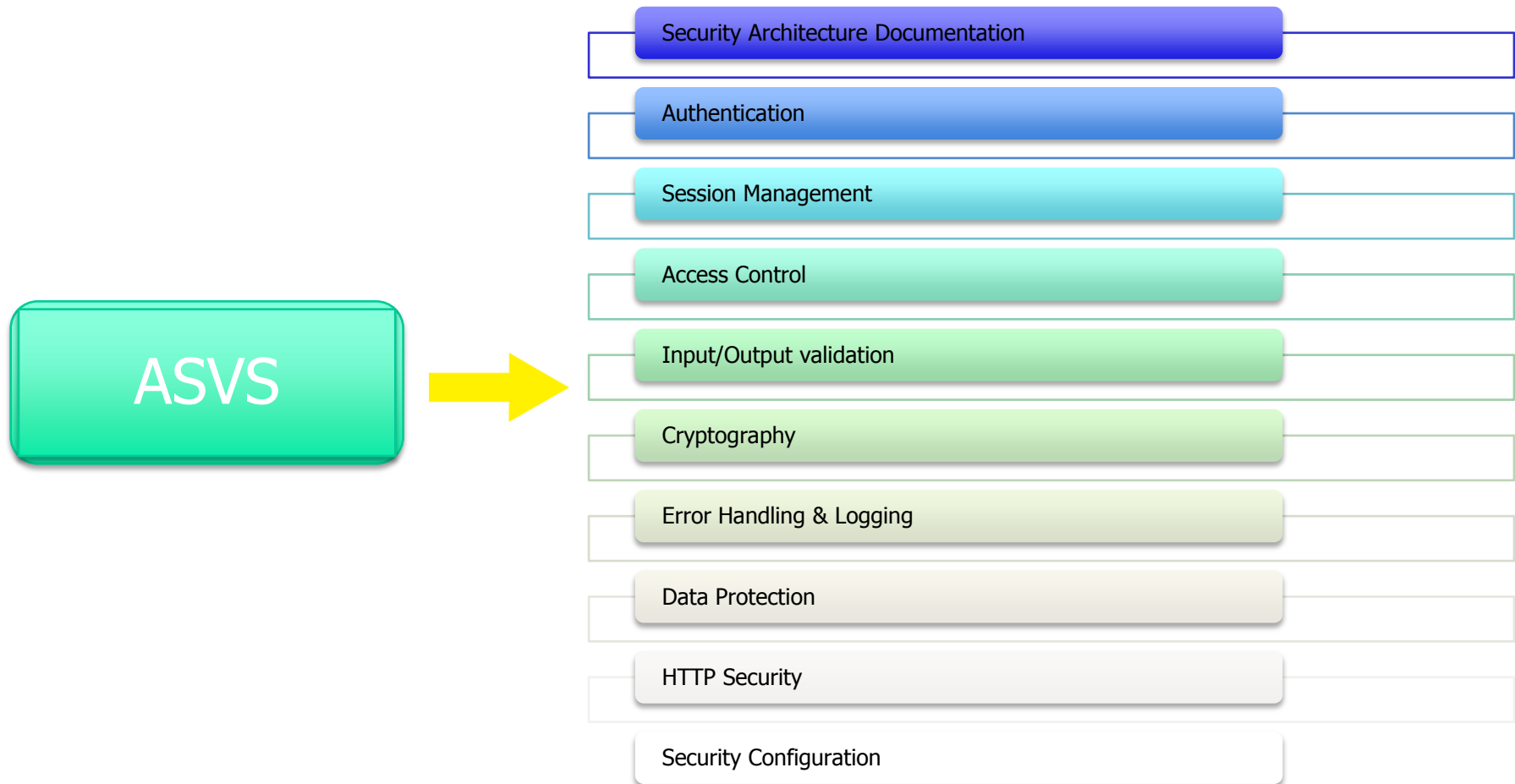
- Autenticación
- Gestión de Sesiones
- Control de Acceso
- Validación de Entradas
- Codificación de Salidas
- Criptografía
- Manejo de Errores
- Protección de Datos
- Seguridad HTTP

Table 6 - OWASP ASVS Output Encoding/Escaping Requirements (V6)

Verification Requirement	Level 1A	Level 1B	Level 2A	Level 2B	Level 3	Level 4
V6.1 Verify that all untrusted data that are output to HTML (including HTML elements, HTML attributes, javascript data values, CSS blocks, and URI attributes) are properly escaped for the applicable context.		✓	✓	✓	✓	✓
V6.2 Verify that all output encoding/escaping controls are implemented on the server side.			✓	✓	✓	✓
V6.3 Verify that output encoding /escaping controls encode all characters not known to be safe for the intended interpreter.				✓	✓	✓
V6.4 Verify that all untrusted data that is output to SQL interpreters use parameterized interfaces, prepared statements, or are escaped properly.				✓	✓	✓

Diseño Seguro – Estándares de desarrollo

Estándares basados en ASVS



Diseño Seguro – Estándares de desarrollo

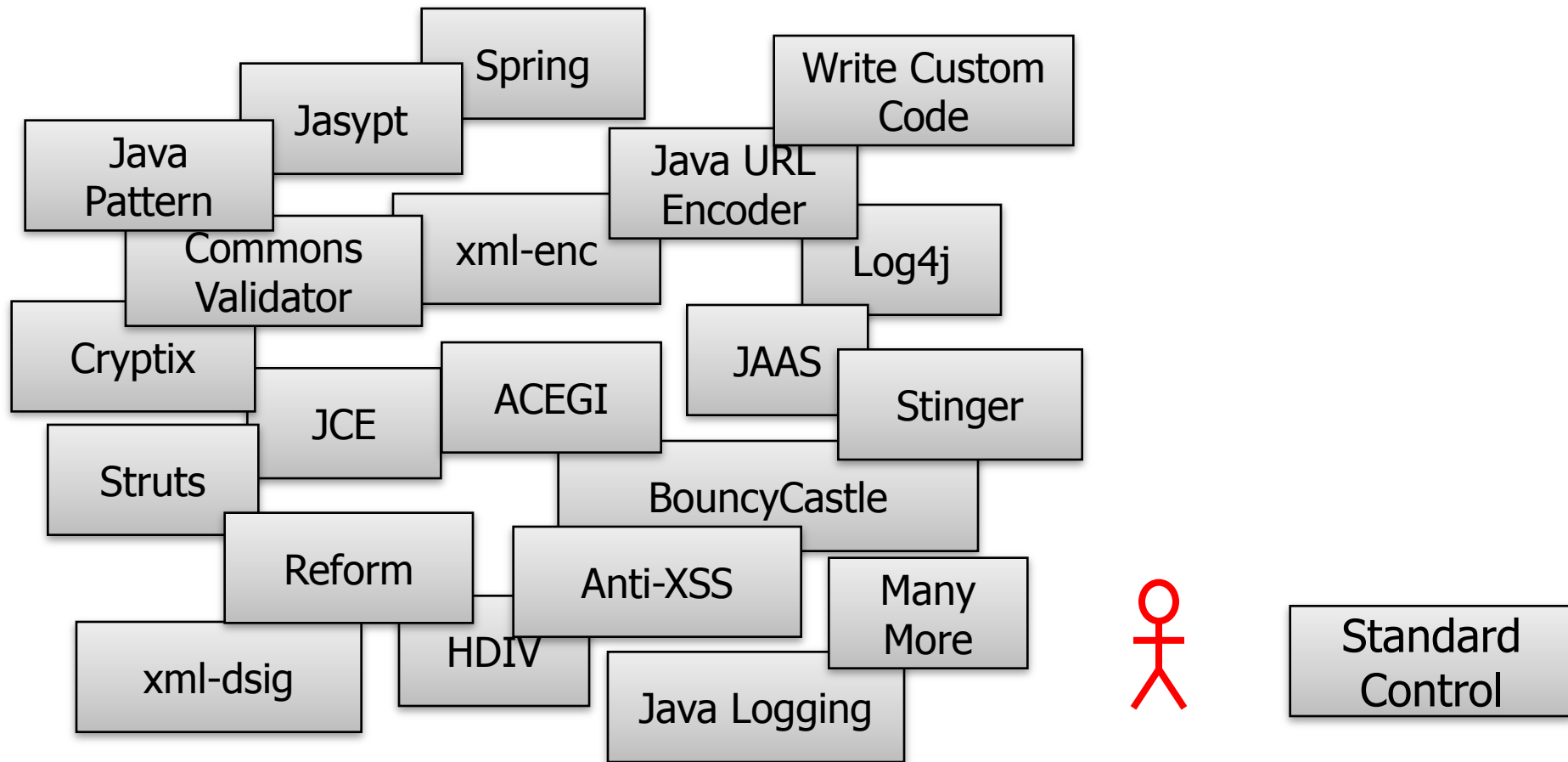
Los requerimientos de ASVS fueron desarrollados con los siguientes objetivos en mente:

- ☐ *Utilizar como una **métrica** – Provee a los desarrolladores y gerentes de aplicaciones con una métrica para determinar el nivel de confianza de las mismas.*
- ☐ *Utilizar como una **guía** – Provee a los desarrolladores de controles de seguridad con indicaciones en que funcionalidades incluir para cumplimentar con los requerimientos de seguridad.*
- ☐ *Utilizar **durante adquisiciones** – Provee una base para especificar los requerimientos de seguridad en aplicaciones adquiridas a terceros.*

Codificación Segura – El problema

Cual es el principal problema con la mayor parte de los frameworks/controles de seguridad?

Codificación Segura – El problema



NO Intuitivo, Integrado o Amigable (para el desarrollador).

Codificación Segura – ESAPI

OWASP ESAPI (Enterprise Security API) apunta a proveer a los desarrolladores con todos los controles de seguridad necesarios:

Estandarizados

Centralizados

Organizados

Integrados

Intuitivos

Testeados

Codificación Segura – ESAPI

Los Toolkits de OWASP Enterprise Security API ayudan a los desarrolladores de software a protegerse de problemas de seguridad relacionados con el diseño o implementación de una aplicación.

Colección de clases que encapsulan los controles de seguridad mas importantes para una aplicación.

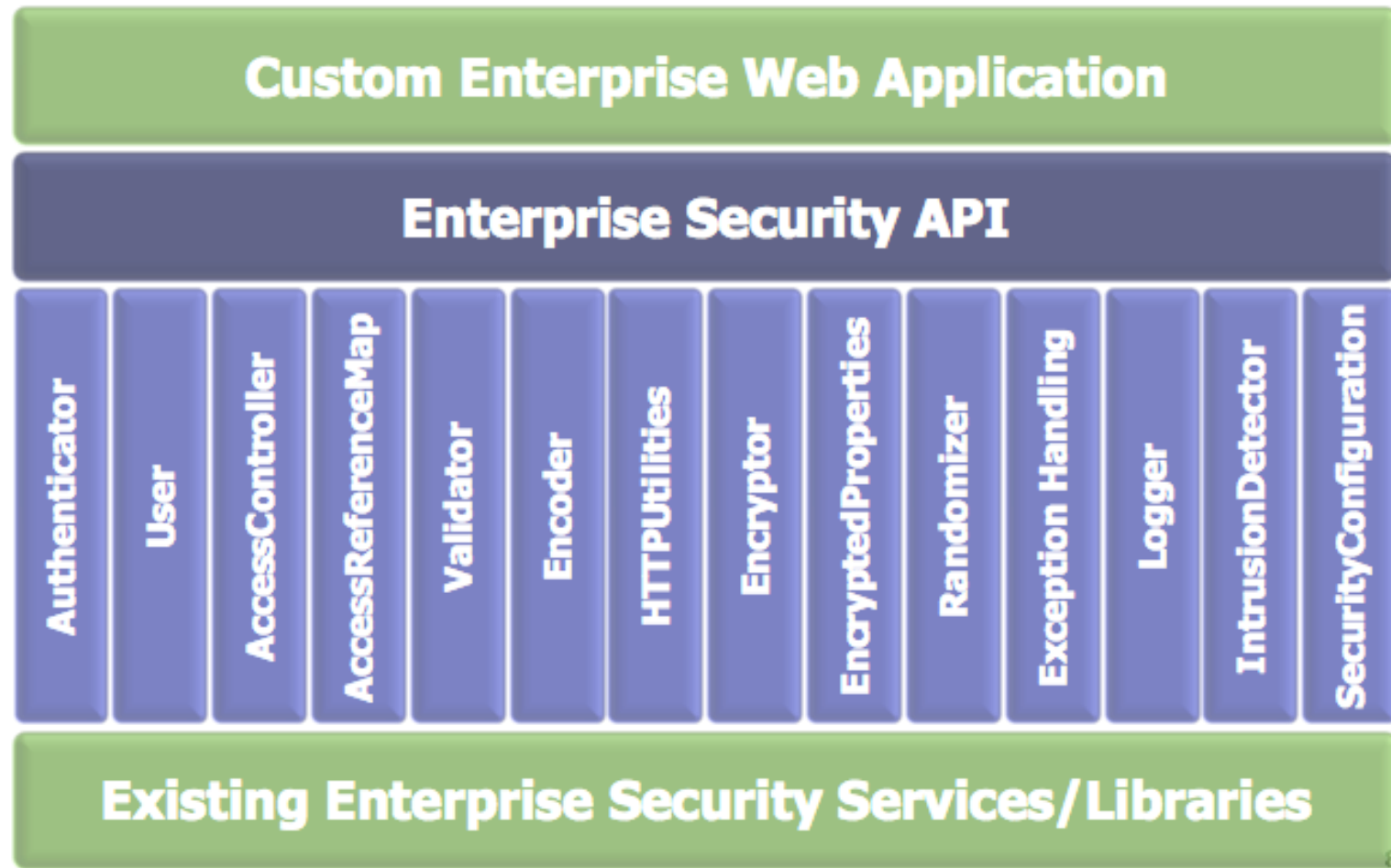
Existen versiones de Java EE, .Net, Javascript, Classic ASP ColdFusion/CFML, PHP y Python.

La version de ESAPI para JAVA EE incluye un Web Application Firewall (WAF) que puede ser utilizado mientras los equipos de desarrollo se focalizan en remediar los problemas.

Todas las versiones de ESAPI se encuentran bajo una licencia BSD de software libre.

Usted puede modificar o utilizar ESAPI como le parezca. Incluso puede incluirlo en productos comerciales de manera totalmente gratuita.

Codificación Segura – Áreas cubiertas por ESAPI

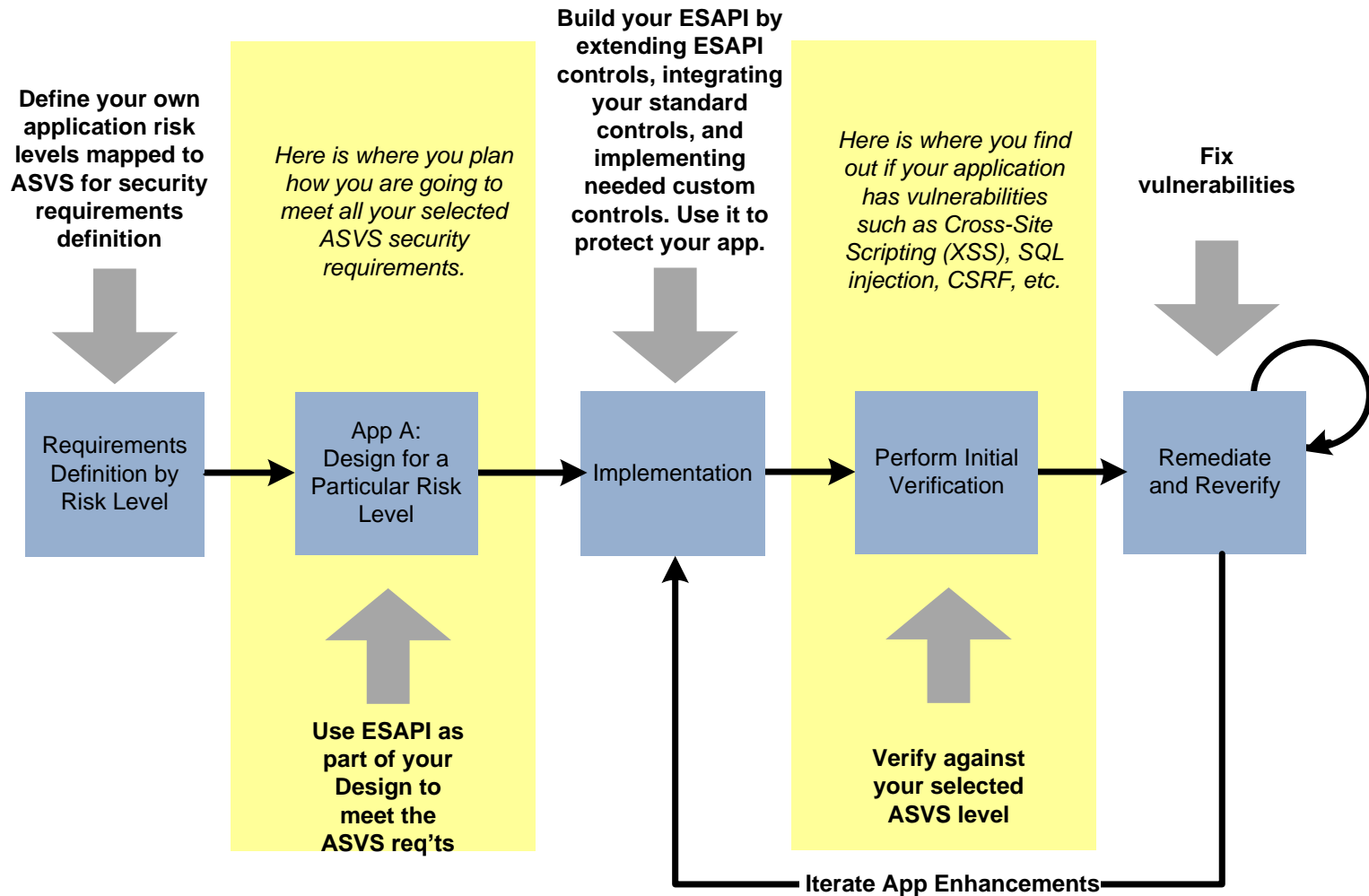


Existen mas de 120 métodos disponibles

OWASP



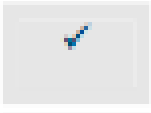
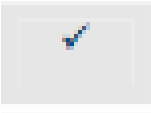
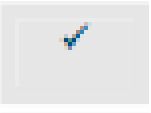
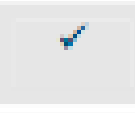


Como integrar ASVS y ESAPI en el SDLC



Mapea de ASVS a ESAPI – Un ejemplo

▶ ASVS Gestión de Sesiones

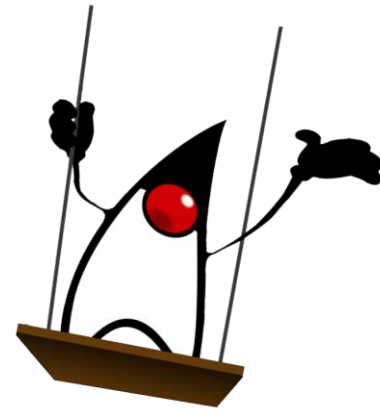
V3.7	Verify that the session id is changed on login.						
------	---	--	---	---	---	---	---

▶ Implementación ESAPI:

- ▶ ESAPI.httpUtilities().changeSessionIdentifier() changes the session id in the login process
- ▶ Además previene fijación de sesión.

Codificación Segura – ESAPI Swingset Interactive

- Aplicacion web que demuestra las ventajas de las librerias ESAPI.
- Alineada con ASVS.
- Apunta a entrenar desarrolladores en ESAPI
 - Cada lab presenta una vulnerabilidad
 - Desarrollador necesita remediarla utilizando ESAPI



El Modelo SAMM

- Metodología que sirve para evaluar las practicas actuales de desarrollo seguro en una organización.
- Puede ser utilizado para implementar un programa de seguridad de aplicaciones en forma **iterativa**.
- Demuestra mejoras concretas en un programa de aseguramiento de seguridad de aplicaciones.
- Define y mide actividades relacionadas a la seguridad en su organización.



Funciones de Negocio SAMM

- Se comienza con el núcleo de actividades presentes en cualquier organización que realiza desarrollo de software
- El nombre asignado es genérico, pero deberían ser identificables por cualquier desarrollador o gestor



Prácticas de Seguridad SAMM

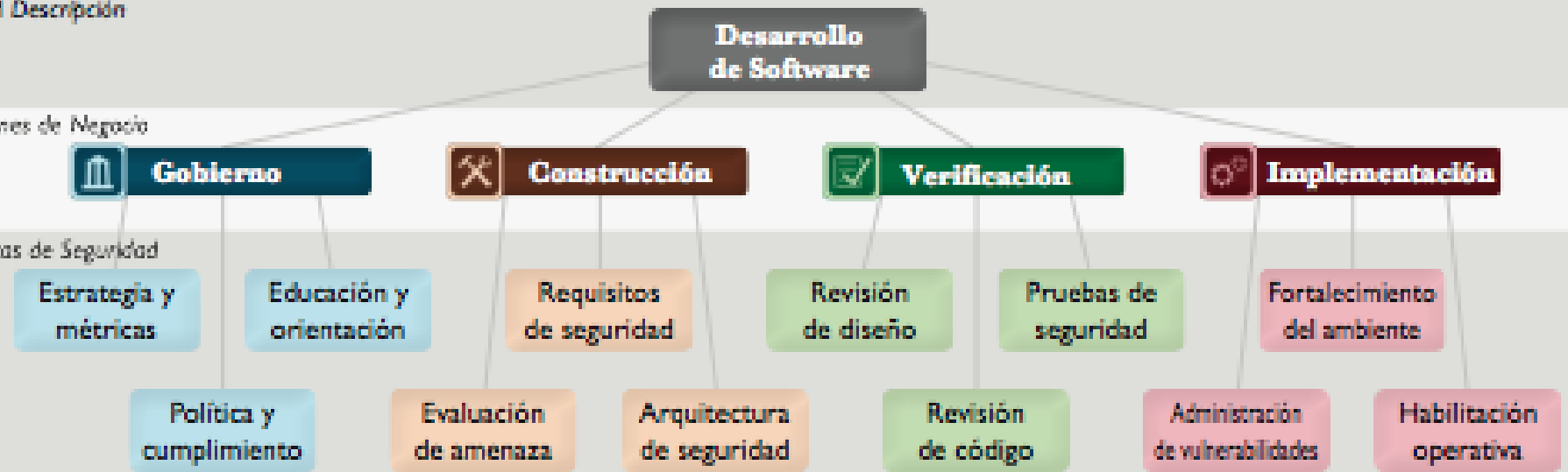
- Por cada una de las Funciones de Negocio se definen 3 Prácticas de Seguridad.
- Dichas prácticas cubren las áreas relevantes al aseguramiento de calidad en el software.
- Cada una de ellas en un 'nicho' de mejora.

El Modelo SAMM

SAMM Descripción

Funciones de Negocio

Prácticas de Seguridad



Prácticas de Seguridad SAMM

Cada Práctica tiene objetivos específicos que definen cómo ir mejorando a lo largo del tiempo.

Esto establece el concepto de **Nivel** en el que una organización se encuentra al cumplir una determinada Práctica.

Los distintos niveles son:

- (0: Punto de partida implícito cuando la Práctica es incumplida)
- 1: Comprensión inicial y disposición específica para adoptar la Práctica
- 2: Incrementar la eficacia y/o eficiencia de la Práctica
- 3: Dominio completo de la Práctica

Prácticas de Seguridad SAMM

EJEMPLO

Educación y orientación

...continúa en página 42



Objetivos

Ofrecer acceso al personal de desarrollo a recursos alrededor de los temas de programación segura e implementación

Educar a todo el personal en el ciclo de vida de software con lineamientos específicos en desarrollo seguro para cada rol

Hacer obligatorio el entrenamiento de seguridad integral y certificar al personal contra la base de conocimiento.

Actividades

- A. Realizar entrenamiento técnico de concientización en seguridad
- B. Crear y mantener lineamientos técnicos

- A. Realizar entrenamiento de seguridad en aplicaciones específico para cada rol
- B. Utilizar mentores de seguridad para mejorar los equipos

- A. Crear un portal formal de soporte de seguridad en aplicaciones
- B. Establecer exámenes o certificaciones por rol



Un enfoque por fases

Fase 1

- Generar concientización sobre seguridad de aplicaciones

Fase 2

- Mejorar la seguridad de aplicaciones

Fase 3

- Implementar metricas para la seguridad de aplicaciones

Un enfoque por fases – Fase 1

Generar concientización sobre seguridad de aplicaciones

- Entrenamiento sobre Seguridad de Aplicaciones.
- Desarrollo de una wiki/foro de discusión interno sobre esta temática.



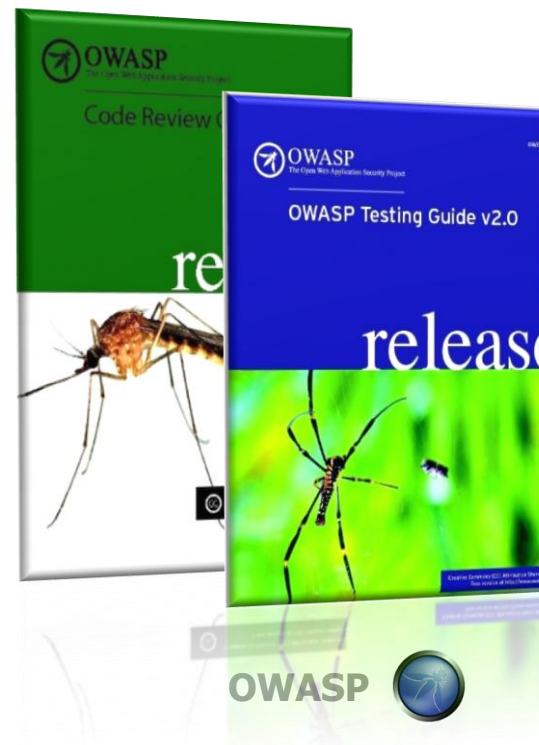
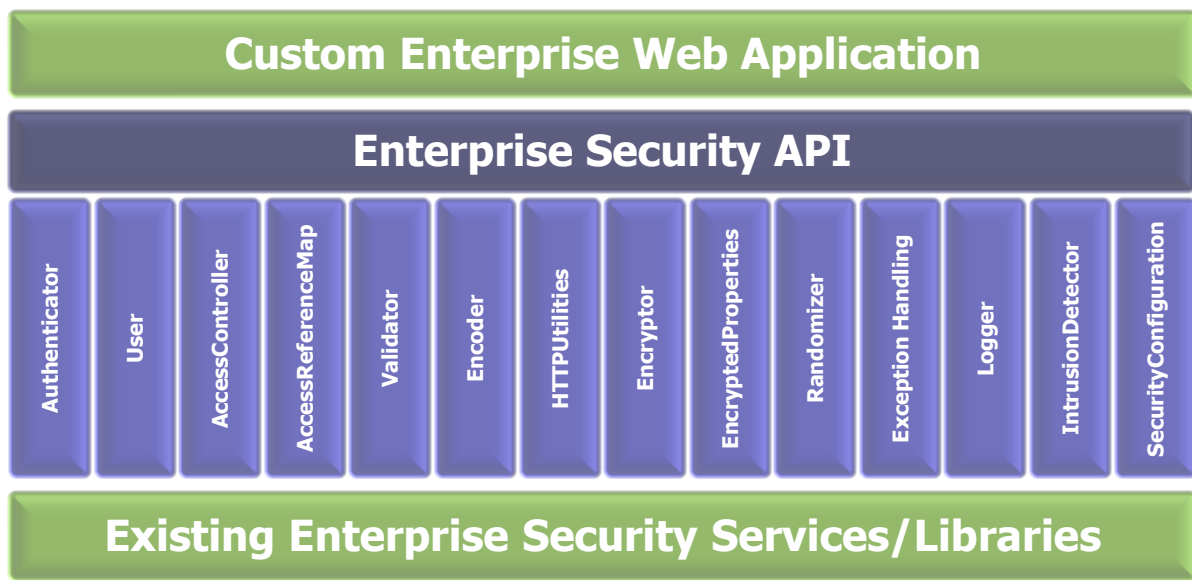
The composite image consists of three main elements:

- Left:** A small goat (the 'Web Goat' mascot) sitting next to a white mug that has 'WEB GOAT' written on it in red.
- Center:** A blue CD-ROM with a glowing blue dragonfly on it. The text 'OWASP Live CD' is printed on the disc.
- Right:** A screenshot of the 'Application Security Group' (ASG) homepage. The page has a title bar 'Application Security Group' and a 'Home' link. Below this is a navigation bar with links: 'View', 'Edit', 'Attachments (2)', and 'Info'. To the right of these links are icons for 'Browse Space', 'Add Page', and 'Add News'. Below the navigation bar, it says 'Added by Fabio Cerullo , last edited by Fabio Cerullo on Feb 05, 2010 (view change)'. There are also labels: 'favourite' and 'EDIT'. The main content area has a heading 'Welcome to the ASG Homepage' and a paragraph: 'ASG's main goal is to help developers across AIB Group on how to design, build and test the security of web applications and web services. Here you will find the tools, standards, procedures and controls in order to improve the security of your application. We advocate approaching application security as a methodology which involves people, process and technology.' There is also a small graphic of a globe with a padlock and a green circular arrow.

Un enfoque por fases – Fase 2

Mejorar la seguridad de aplicaciones

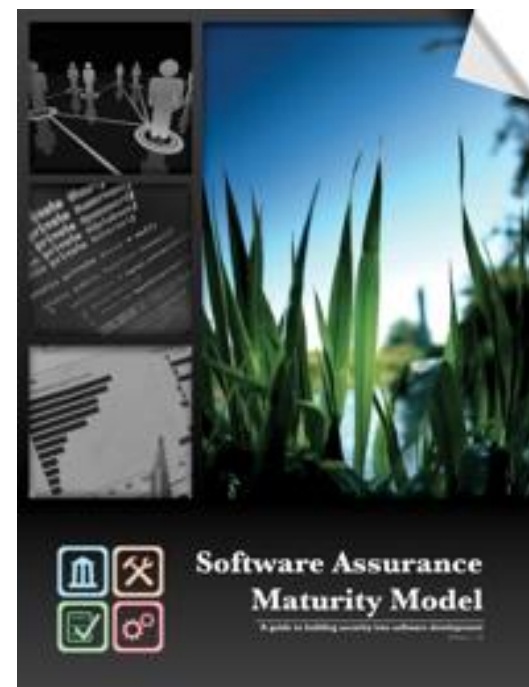
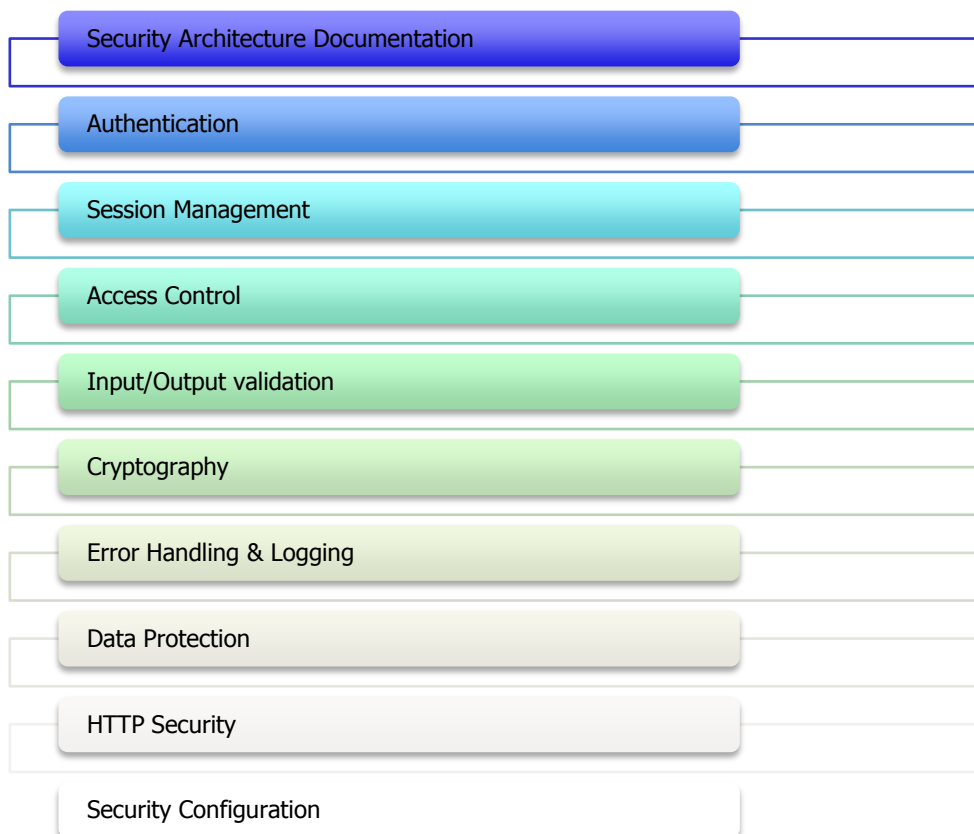
- Entrenamiento para desarrolladores sobre OWASP ESAPI & Swingset
- Promover el testeo cruzado de seguridad de aplicaciones.
- Implementar un proceso de revision de codigo.
- Desarrollar Estandares de Desarrollo (ASVS)



Un enfoque por fases – Fase 3

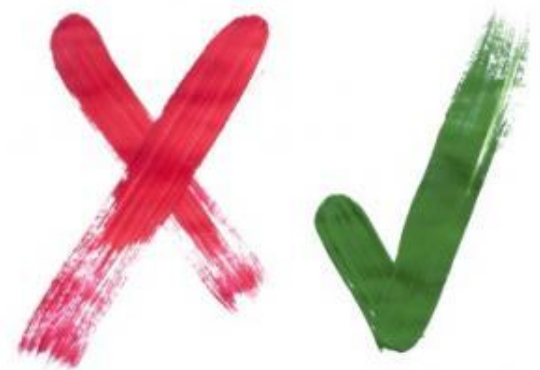
Implementar métricas sobre seguridad de aplicaciones

- Promover Metodologías de Desarrollo Seguro (OpenSAMM)
- Realizar auditorias de aplicaciones criticas (ASVS)



Recomendaciones

1. Definir alcance (comenzar con proyecto pequeño)
2. Obtener apoyo de Gerencia.
3. Acercarse a un equipo de desarrollo específico.
4. Entrenarlos sobre Seguridad de Aplicaciones.
5. Realizar una revisión de Seguridad de dicho proyecto.
6. Medir y documentar los resultados.
7. Comenzar nuevamente (extender el alcance)



Conclusiones

Por que implementar seguridad de aplicaciones?

- Reduce costos de desarrollo, recuperacion ante incidentes y parches.
- Reduce costo de testeo de seguridad por terceros.
- OWASP provee recursos y herramientas gratuitas.
- Costos de implementacion son minimos.
- El enfoque por fases resulta positivo.

Preguntas y Respuestas

Muchas gracias!

FCERULLO@OWASP.ORG

