Universitat Oberta de Catalunya

uoc.edu

Madurez del ciclo de vida del desarrollo de software seguro: OWASP Software Assurance Maturity Model (SAMM)

Nombre Estudiante: Rafael Medina Casas

Máster Universitario en Ciberseguridad y Privacidad

Área del trabajo final: Seguridad empresarial

Nombre Consultor/a: Pau del Canto Rodrigo

Nombre Profesor/a responsable de la asignatura: Víctor García Font

Fecha Entrega: Junio 2021

FICHA DEL TRABAJO FINAL

Título del trabajo:	*OWASP Software Assurance Maturity Model (SAMM)*
Nombre del autor:	*Rafael Medina Casas*
Nombre del consultor/a:	*Pau del Canto Rodrigo*
Nombre del PRA:	*Pau del Canto Rodrigo / Víctor García Font*
Fecha de entrega:	Junio/2021
Titulación:	*Máster Universitario en Ciberseguridad y Privacidad*
Área del Trabajo Final:	*M1.887 - Seguridad empresarial*
Idioma del trabajo:	*Castellano*
Palabras clave	*SAMM* *Secure Software Development Life Cycle*

Resumen del Trabajo (máximo 250 palabras): *Con la finalidad, contexto de aplicación, metodología, resultados y conclusiones del trabajo.*

La finalidad de este trabajo es analizar y describir el modelo de madurez del aseguramiento del software SAMM de OWASP.

En la actualidad, la seguridad de los sistemas de información es un aspecto clave para asegurar la prestación de servicios seguros, resilientes y confiables, así como para garantizar algunos derechos fundamentales de las personas como puede ser la privacidad y la protección de los datos personales. De forma paralela, se observa un aumento en el número de vulnerabilidades de los productos software, que crece año tras año.

En este contexto, las organizaciones empiezan a ser conscientes de la necesidad de gestionar la seguridad en el desarrollo de software y precisan de poner en marcha programas de mejora de la seguridad en el desarrollo de software.

El modelo de madurez SAMM constituye un instrumento de especial utilidad para implementar un programa de seguridad consistente. El presente trabajo analiza este modelo de madurez y su aplicación para gestionar la seguridad en el desarrollo de software.

Como conclusión principal, SAMM facilita en gran medida la puesta en marcha de un programa de mejora de la seguridad del software de cualquier organización, siendo especialmente útil para proporcionar orientación a aquellas organizaciones con poca o nula madurez en materia de seguridad del software.

Abstract (in English, 250 words or less):

The main purpose of this paper is to analyze and describe the OWASP SAMM software assurance maturity model.

Currently, the security of information systems is a key aspect to ensure the provision of secure, resilient and reliable services, as well as to guarantee some fundamental rights of individuals such as privacy and personal data protection. At the same time, the number of vulnerabilities in software products is increasing year after year.

In this context, organizations are becoming aware of the need to manage security in software development and need to implement programs to improve security in software development.

The SAMM maturity model is a particularly useful tool for implementing a consistent security program. This paper analyzes this maturity model and its application to manage safety in software development.

As a main conclusion, SAMM greatly facilitates the implementation of a software safety improvement program in any organization, being especially useful to provide guidance to those organizations with little or no software safety maturity.

Índice

Lista de tablas

Lista de figuras

1 Introducción

1.1 Contexto y justificación del Trabajo

En el contexto actual, la seguridad de los sistemas de información es un aspecto clave necesario para asegurar la prestación de servicios seguros, resilientes y confiables, así como para garantizar algunos derechos fundamentales de las personas como puede ser la privacidad y la protección de los datos personales.

Tradicionalmente, la seguridad de la información se ha implementado como una capa adicional con la que se protegía a los diversos componentes y sistemas una vez que éstos se desplegaban en los entornos de producción. Es decir, se actuaba en la última fase del proceso del ciclo de vida del software: la operación. La seguridad, por tanto, se confiaba (en algún caso todavía se sigue confiando) casi exclusivamente a diversos componentes software y hardware que actuaban como escudo de las aplicaciones, por ejemplo, firewalls, firewalls de aplicación (WAF), sistemas de prevención de intrusiones (IPS), etc.

Este mismo enfoque, por cierto, se ha aplicado tradicionalmente al aseguramiento de la calidad en el desarrollo de sistemas de información. Así, normalmente la fase de pruebas, se ejecutaba al final del proceso del desarrollo con el fin de verificar que aquel sistema que se desplegaba en producción, cumplía con las expectativas respecto a su funcionalidad, fiabilidad y estabilidad.

En lo que respecta al aseguramiento de la calidad, desde hace unos años se ha observado la tendencia a “desplazar” las actividades de “testing” hacia etapas más tempranas del ciclo de vida del desarrollo. Idealmente, desde el inicio del proceso y a lo largo de todo el ciclo de vida del proyecto. Este concepto, es conocido como *shift left* (viendo el proceso de desarrollo como una secuencia temporal de actividades ordenadas en una línea temporal que va de izquierda a derecha).

De la misma forma que ha sucedido con los aspectos relacionados con la calidad, en los últimos años se viene observando la misma tendencia *shift left* en lo que a seguridad se refiere, integrando las actividades relacionadas con la seguridad en las diversas etapas del ciclo de vida de desarrollo, desde las fases más tempranas en la concepción del sistema de información hasta la fase de operación. Este enfoque, denominado *security by design*, tiene como objetivo actuar y resolver de forma preventiva potenciales problemas de seguridad antes de que éstos se materialicen en vulnerabilidades, amenazas o incidentes de seguridad.

Como se mencionó en el párrafo anterior, integrar la seguridad en el ciclo de vida del desarrollo de software conlleva la aplicación de prácticas de seguridad específicas para cada una de las fases del proceso. Para llevar a cabo este ejercicio, existen diversas guías de y catálogos de principios y buenas prácticas de seguridad en el desarrollo de software que sirven de gran ayuda para la implantación de un proceso de desarrollo seguro en una organización.

La incorporación de buenas prácticas de seguridad en el proceso de desarrollo de software, contribuye, sin duda, a la mejora en la seguridad de los productos

obtenidos y, por consiguiente, a reducir el riesgo al que está expuesta una organización.

Sin embargo, la implantación de un proceso de desarrollo seguro no es tarea sencilla. Las organizaciones deben dedicar un esfuerzo nada despreciable para ello y deberán enfocar sus esfuerzos adecuadamente para desarrollar un proceso de gestión del ciclo de vida de las aplicaciones que sea óptimo y se adapte a sus circunstancias. Así, durante el diseño e implantación del proceso, las organizaciones deben responder a un conjunto de cuestiones tales como: ¿Qué prácticas de seguridad son más adecuadas o más prioritarias en el contexto de la organización? ¿Cuál es el estado de la seguridad del proceso actual? ¿Qué fases del proceso me interesa mejorar inicialmente? ¿Cómo se puede evaluar en qué medida ha mejorado la seguridad del proceso? ¿Cuál es el estado de la seguridad del nuevo proceso implantado en comparación con la situación anterior? ¿Y con respecto a la industria?

Los modelos de madurez de la capacidad (CMM) son un instrumento muy útil para obtener respuestas a las cuestiones planteadas anteriormente, pues proporcionan un modelo de referencia sobre prácticas de madurez constatada y adoptadas ampliamente por industria de referencia. Así, las organizaciones pueden comparar las prácticas adoptadas y evaluar de forma objetiva su situación con respecto al modelo de referencia, así como identificar potenciales áreas de mejora. Un ejemplo de un modelo de madurez ampliamente conocido en el ámbito del desarrollo de software es CMMI.

En el contexto de la seguridad en el ciclo de vida del software, existen varios modelos de madurez. Dos de los más conocidos son BSIMM y SAMM. El presente trabajo se centra en el modelo *Software Assurance Maturity Model* (SAMM) desarrollado por la fundación *Open Web Application Security Project*® (OWASP), una organización con amplia trayectoria y cuya principal misión es la de "*mejorar la seguridad del software a través de iniciativas de código abierto y educación comunitaria*".

El modelo SAMM constituye una herramienta muy útil para medir, analizar y mejorar de forma evolutiva el proceso del ciclo de vida del software de cualquier organización, ya sea pequeña, mediana o grande, es aplicable sobre cualquiera de los procesos o metodologías de desarrollo adoptadas (cascada, agile, etc.) y es independiente de la tecnología. Adicionalmente, la aplicación del modelo y las herramientas de evaluación de madurez que éste proporciona, constituyen un instrumento muy efectivo para el gobierno de la seguridad en el desarrollo de software de la organización.

En el presente trabajo, se profundizará sobre los diversos aspectos del modelo y se intentará llevar a cabo una propuesta de aplicación práctica basada en soluciones de código abierto para las diversas prácticas de seguridad que constituyen el marco de referencia SAMM.

1.2 Objetivos del Trabajo

Los objetivos de este Trabajo son:

- Presentar el *framework* SAMM de OWASP y describir los elementos principales que constituyen el modelo: funciones de negocio, prácticas de seguridad y niveles de madurez.

- Comparar OWASP SAMM con otros modelos de madurez.
- Investigar acerca de modelo de madurez SAMM y cómo aplicar el modelo para la puesta en marcha de un programa de mejora incremental de la seguridad en el ciclo de vida de desarrollo del software.
- Investigar la aplicación del modelo para el gobierno de la seguridad en el proceso de desarrollo de software.
- Desarrollar un ejemplo práctico de aplicación de SAMM a una organización ficticia con un equipo de desarrollo de tamaño pequeño y obtención de resultados de proceso de implantación incremental del modelo.
- Elaborar propuesta basada en soluciones de código abierto para implementar diversas prácticas recogidas en el modelo.

1.3 Enfoque y método seguido

El enfoque general del trabajo consiste en investigar sobre el modelo SAMM, conocer los elementos fundamentales del modelo y estudiar su aplicación para:

- la puesta en marcha de programas de mejora de la madurez en la seguridad del ciclo de vida de desarrollo de software, y
- la gobernanza del estado de la seguridad a lo largo del ciclo de vida de desarrollo de software

Para conseguir los objetivos perseguidos, se analizará la documentación de SAMM así como otras fuentes de documentación de diferente procedencia, tales como artículos, presentaciones, material audiovisual de seminarios y conferencias. Adicionalmente se recopilará información sobre otros modelos de madurez con el fin de realizar una comparación entre SAMM y modelos alternativos.

Se realizará un análisis pormenorizado de la documentación para conocer en detalle las características diferenciales de SAMM y los conceptos esenciales sobre los que se sustenta el modelo. A partir de este conocimiento se llevará a cabo un estudio de la aplicación práctica de este modelo a los dos casos de uso mencionados al inicio del presente apartado.

Por otro lado, se analizarán las herramientas proporcionadas por el grupo de trabajo encargado del mantenimiento y evolución del modelo, para desarrollar un ejemplo práctico del proceso de evaluación de la madurez de una organización, incluyendo la simulación de una iteración del proceso iterativo de mejora de la madurez. De esta forma, se mostrará el modo en que utilizaría el modelo en la práctica para la puesta en marcha de programas de mejora de la madurez de la seguridad en el ciclo de vida de desarrollo de software.

Finalmente, otro de los objetivos del trabajo es la elaboración de una propuesta de solución para el cumplimiento de las diversas prácticas y actividades especificadas en SAMM. Por tanto, se recopilará información sobre herramientas y soluciones, preferentemente de código abierto, con las que satisfacer los requerimientos del modelo. Se estudiará cómo dichas herramientas pueden ser aplicadas de forma práctica para cubrir los diversos niveles de madurez de las buenas prácticas referidas en el modelo.

1.4 Planificación del Trabajo

El esfuerzo estimado para llevar a cabo el trabajo previsto es de unas 300 horas, que equivalen a los 12 créditos asociados al TFM.

Durante el desarrollo del TFM se realizarán tres entregas parciales y una entrega final.

El proyecto se ha estructurado en 4 fases: Planificación, Investigación, Aplicación Práctica y Entrega

Cada una de estas cuatro fases forma un paquete de trabajo que comprende las diferentes actividades que se llevaran a cabo para la finalización del TFM.

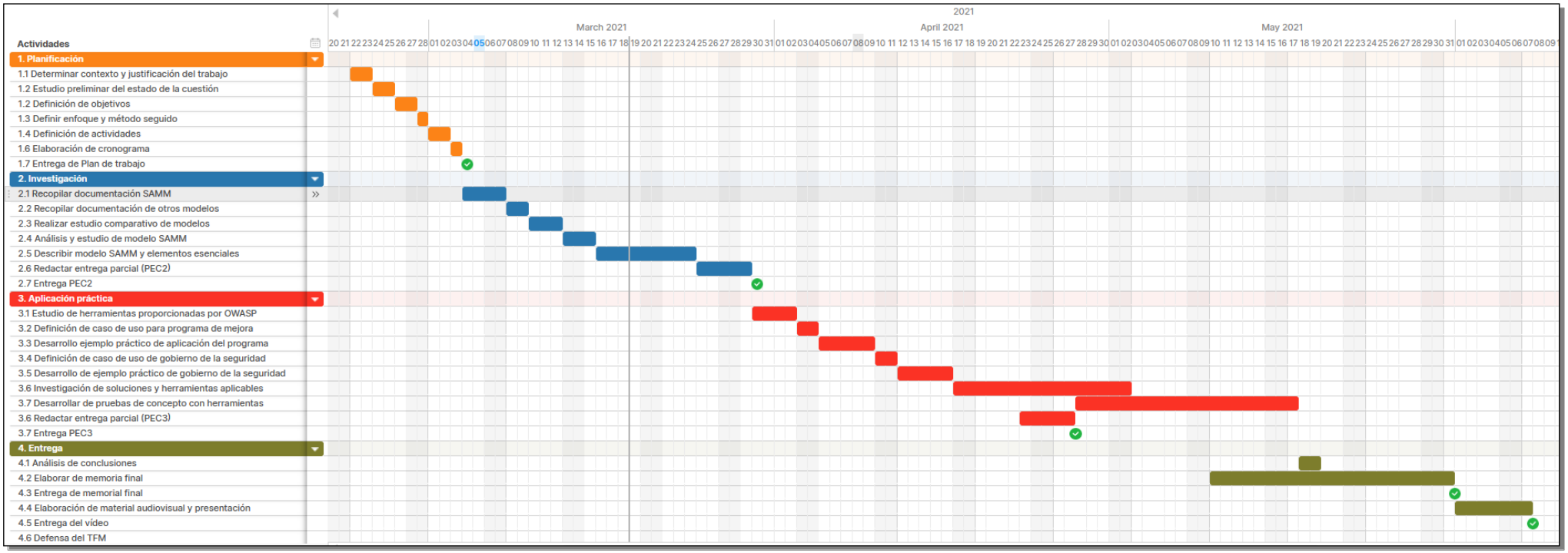
A continuación se muestran la descomposición de tareas en cada una de las fases.

Tabla 1. Descomposición de tareas y estimación de duración

Actividades	Start	End	Days
1. Planificación	**22-02-21**	**04-03-21**	**10.0**
1.1 Determinar contexto y justificación del trabajo	22-02-21	23-02-21	2.0
1.2 Estudio preliminar del estado de la cuestión	24-02-21	25-02-21	2.0
1.2 Definición de objetivos	26-02-21	27-02-21	2.0
1.3 Definir enfoque y método seguido	28-02-21	28-02-21	1.0
1.4 Definición de actividades	01-03-21	02-03-21	2.0
1.6 Elaboración de cronograma	03-03-21	03-03-21	1.0
1.7 Entrega de Plan de trabajo	04-03-21	04-03-21	
2. Investigación	**04-03-21**	**30-03-21**	**26.0**
2.1 Recopilar documentación SAMM	04-03-21	07-03-21	4.0
2.2 Recopilar documentación de otros modelos	08-03-21	09-03-21	2.0
2.3 Realizar estudio comparativo de modelos	10-03-21	12-03-21	3.0
2.4 Análisis y estudio de modelo SAMM	13-03-21	15-03-21	3.0
2.5 Describir modelo SAMM y elementos esenciales	16-03-21	24-03-21	9.0
2.6 Redactar entrega parcial (PEC2)	25-03-21	29-03-21	5.0
2.7 Entrega PEC2	30-03-21	30-03-21	
3. Aplicación práctica	**30-03-21**	**17-05-21**	**59.0**
3.1 Estudio de herramientas proporcionadas por OWASP	30-03-21	02-04-21	4.0
3.2 Definición de caso de uso para programa de mejora	03-04-21	04-04-21	2.0
3.3 Desarrollo ejemplo práctico de aplicación del programa	05-04-21	09-04-21	5.0
3.4 Definición de caso de uso de gobierno de la seguridad	10-04-21	11-04-21	2.0
3.5 Desarrollo de ejemplo práctico de gobierno de la seguridad	12-04-21	16-04-21	5.0
3.6 Investigación de soluciones y herramientas aplicables	17-04-21	02-05-21	16.0
3.7 Desarrollar de pruebas de concepto con herramientas	28-04-21	17-05-21	20.0
3.6 Redactar entrega parcial (PEC3)	23-04-21	27-04-21	5.0
3.7 Entrega PEC3	27-04-21	27-04-21	
4. Entrega	**10-05-21**	**18-06-21**	**31.0**
4.1 Análisis de conclusiones	18-05-21	19-05-21	2.0
4.2 Elaborar de memoria final	10-05-21	31-05-21	22.0
4.3 Entrega de memorial final	31-05-21	31-05-21	
4.4 Elaboración de material audiovisual y presentación	01-06-21	07-06-21	7.0
4.5 Entrega del vídeo	07-06-21	07-06-21	
4.6 Defensa del TFM	18-06-21	18-06-21	
			126.0

En la siguiente figura se muestra la planificación temporal estimada inicialmente para la ejecución de las actividades del trabajo.

Tabla 2. Planificación de tareas (Gantt)



1.5 Breve sumario de productos obtenidos

- Comparativa de diversos modelos de madurez aplicables a la seguridad en el proceso del ciclo de vida del desarrollo de software.
- Descripción detallada de las diversas prácticas de seguridad del modelo, agrupadas por función de negocio.
- Ejemplo práctico de aplicación del modelo a programa de mejora de la seguridad en el ciclo de vida del desarrollo de software.
- Ejemplo práctico de aplicación del modelo para el gobierno de la seguridad en el ciclo de vida del desarrollo de software.
- Relación de soluciones de código abierto para dar cumplimiento a controles asociados a las actividades del modelo.

1.6 Breve descripción de los otros capítulos de la memoria

A continuación se resume el contenido de los diversos capítulos del trabajo:

- **Capítulo 2**. Estado de la cuestión (estado del arte). En este capítulo se realiza una revisión de las circunstancias actuales del desarrollo de software en relación a la seguridad. Se muestra la necesidad de mejorar los procesos de gestión del ciclo de vida del software y la tendencia a incorporar la seguridad desde las primeras etapas del desarrollo. Adicionalmente, se presentan los modelos de madurez como instrumento para evaluar el estado de la seguridad del software y mejorar el proceso de desarrollo.
- **Capítulo 3**. Modelos de madurez para el desarrollo de software seguro. En este capítulo se presentan los modelos de madurez SAMM y BSIMM y se realiza una breve comparativa de ambos.
- **Capítulo 4**. Modelo SAMM de OWASP. Se describe en detalle la estructura y elementos esenciales del modelo de madurez SAMM de OWASP.
- **Capítulo 5.** Aplicación de SAMM para la implementación de un programa de mejora de la seguridad del software. En este capítulo se describe cómo implantar un programa de seguridad basado en SAMM. Se enumeran y detallan los pasos a ejecutar para desarrollar el proyecto de implantación del modelo de madurez en una organización.
- **Capítulo 6.** Gobierno de la seguridad en el desarrollo de software. Tras la puesta en marcha del programa de seguridad en el software, se trata brevemente cómo gestionar el ciclo de mejora continua del mismo.
- **Capítulo 7.** Herramientas y utilidades para cumplimiento de prácticas y actividades mediante herramientas *open source*. En este capítulo se enumerarán algunas herramientas y referencias que pueden servir de gran ayuda para abordar diferentes actividades de seguridad del modelo.
- **Capítulo 8.** Conclusiones. Se resumen los contenidos tratados y se exponen las principales conclusiones obtenidas. Adicionalmente, se proponen algunas opciones para desarrollar los siguientes trabajos en el área.
- **Capítulo 9.** Bibliografía y referencias.

2 Estado de la cuestión (estado del arte)

2.1 Desarrollo de software seguro

En la actualidad se podría decir que, en cuanto a calidad se refiere, en la industria del software existe cierta madurez en los procesos de desarrollo de software y, en general, los productos finales ofrecen un grado de calidad bastante aceptable. Ya en los años 90 comenzaron a aparecer los primeros estándares, normas y modelos para asegurar la calidad del software, entendida esta como “el grado con el que un sistema componente o proceso cumple los requerimientos especificados y las necesidades o expectativas del cliente o usuario” [1].

Desde las etapas iniciales del desarrollo de un producto software, se trabaja enormemente en la funcionalidad que debe ofrecer el producto, el diseño del interfaz de usuario, el tiempo de respuesta, la capacidad para atender la demanda, etc. Todo ello con el objetivo de satisfacer los requerimientos, necesidades y expectativas del cliente. Es decir, para obtener un producto con la mayor calidad posible.

Sin embargo, en el desarrollo de software, hoy en día se sigue descuidando con bastante frecuencia un aspecto que cada vez toma mayor relevancia: la seguridad. En un momento en el que el número de ciberataques aumenta globalmente, parece imprescindible prestar la debida atención a la seguridad del software para reducir el número de vulnerabilidades presentes en las aplicaciones. Aplicaciones que, además, cada vez se vuelven más necesarias e imprescindibles para los usuarios, empresas e instituciones y que sostienen servicios tan esenciales como la salud, finanzas, administración electrónica, la energía, comunicaciones, etc.

Paralelamente al crecimiento del volumen de software, así como de la complejidad del mismo, es de esperar un aumento global del número de vulnerabilidades (ver Figura 1). Un mayor número de vulnerabilidades supone un aumento del nivel de riesgo global, pues estas pueden ser aprovechadas y explotadas por los atacantes provocando incidentes de seguridad.

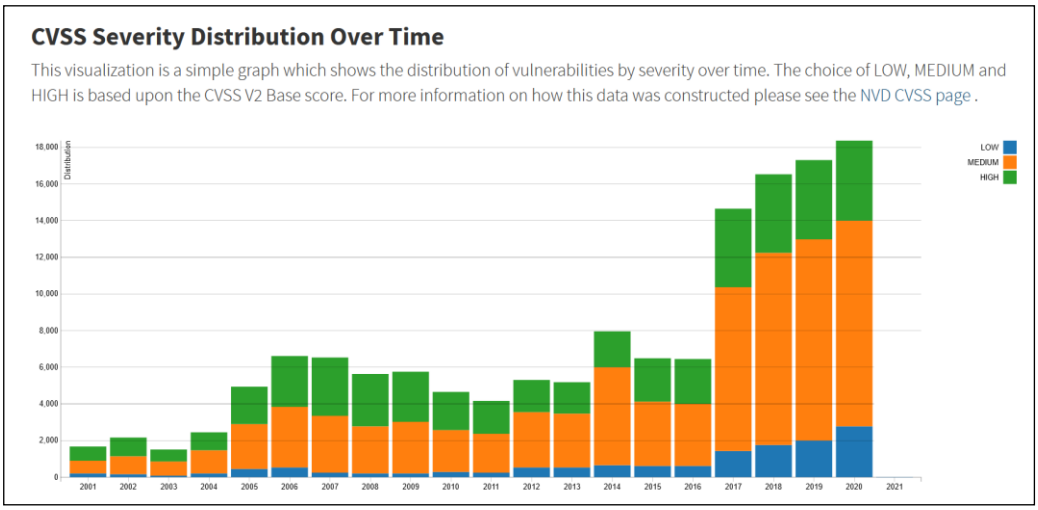


Figura 1. Distribución de vulnerabilidades en el tiempo. Fuente: NVD. NIST

El escenario que se presenta no parece muy halagüeño: gran dependencia de productos software, mayor número de vulnerabilidades y un gran número de amenazas. Las organizaciones empiezan a ser conscientes del impacto que un ciberataque puede ocasionar: daños de reputación, demandas y costosos tiempos de inactividad, etc. Para tener una idea de la magnitud de los costes, según el estudio "Informe sobre el coste de una brecha de datos 2020" [2] realizado por IBM, indica que el coste medio por cada registro de información personal perdido o robado alcanza la cifra de 150$. Se puede deducir del dato anterior las graves consecuencias económicas que puede suponer para las organizaciones el software inseguro. Tomemos además en consideración la llegada del Internet de las Cosas (IoT) y el impacto adicional que un problema de seguridad puede suponer para el usuario medio. Por ello, desde la alta dirección se está prestando una mayor atención a las prácticas de seguridad de las organizaciones.

Gran parte del problema podría mitigarse teniendo en cuenta los aspectos de seguridad desde el inicio del proceso de desarrollo de software. Es decir, adoptando una estrategia preventiva en lugar de actuar de forma reactiva. Evitar la herida en vez de recurrir a una tirita. Pasar de la gestión de parches al aseguramiento del software Esto es lo que se conoce con el término "*security by design"*[1].

El escenario presentado en el párrafo anterior ha obligado a las organizaciones a adoptar un enfoque "*security by design",* considerando la seguridad desde el inicio del proceso en lugar de prestarle atención en la fase posterior al mismo. Se ha observado que el 90% de las vulnerabilidades tienen su origen en defectos de diseño y de programación [3]. Teniendo en cuenta la seguridad desde el principio, se consigue eliminar gran parte de estos defectos, obteniendo como resultado productos más seguros y, por tanto, una reducción de los riesgos de negocio de las organizaciones.

Cuando se añaden las características de seguridad desde la concepción del software y forman parte de los criterios de diseño, los desarrolladores trabajan sobre dichos requisitos o especificaciones de la misma forma que lo hacen con los requisitos funcionales, y los aspectos de seguridad se abordan como una parte más del producto final, que será validado mediante las correspondientes pruebas de aceptación. Si las especificaciones de seguridad no se contemplan desde el inicio, es muy probable que los desarrolladores no traten dichas cuestiones debidamente y la seguridad final del producto quede en manos del azar. Según informe del Fraunhofer Institute For Secure Information Technology [4], aplicar un proceso de desarrollo seguro mejora sustancialmente la seguridad del producto final reduciendo significativamente el número de vulnerabilidades (Figura 2).

---

[1] En contextos relacionados con la privacidad o RGPD se utiliza el término "*privacy by design*"

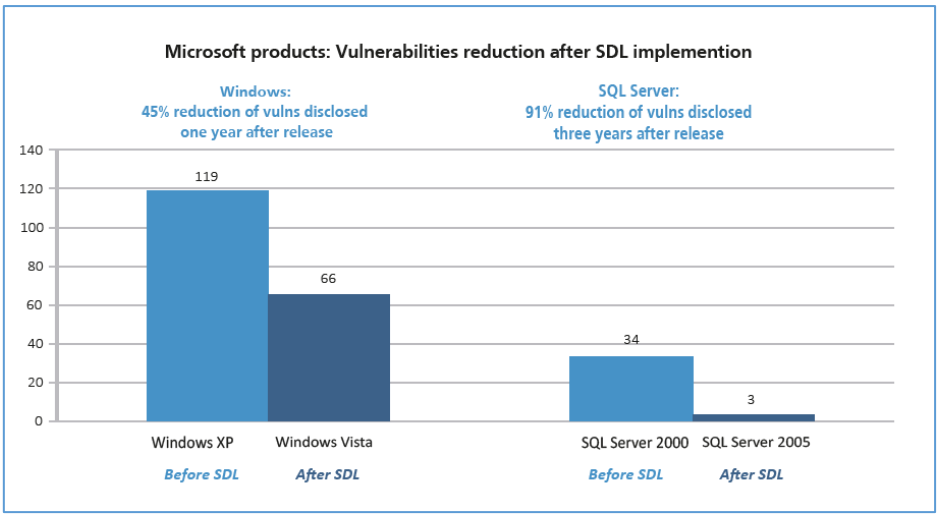


Figura 2. Impacto del ciclo de desarrollo seguro en la seguridad del software. Fuente **[4]**

Otro beneficio de adoptar una estrategia preventiva, es la reducción de costes asociada a la reparación de defectos en las aplicaciones. Según estudio realizado por el IBM System Science Institute [5], el coste de solucionar un defecto encontrado en la fase de implementación era 6 veces superior que arreglarlo si se hubiera identificado durante la fase de diseño. El mismo defecto detectado en la fase de pruebas, supondría un coste 15 veces mayor que si se hubiera detectado durante el diseño (Figura 3)

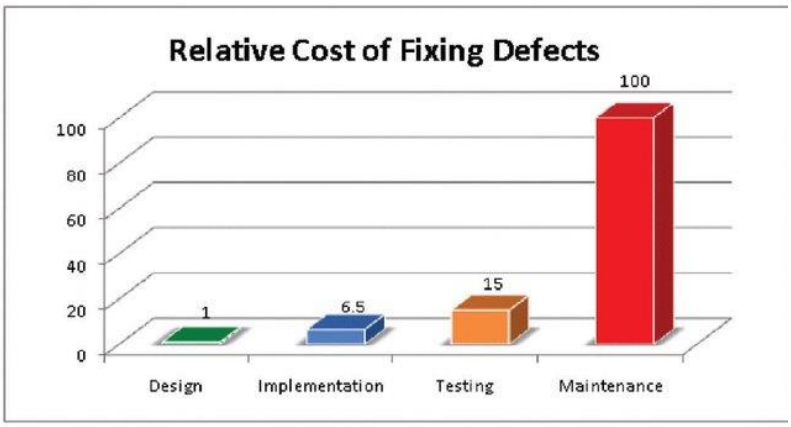


Figura 3. Coste de reparación relativo según fase del proceso de desarrollo. Fuente IBM S.S. Institute **[5]**

Además de los problemas mencionados respecto a la seguridad y calidad de los productos software, es necesario mencionar los riesgos que supone para la organización el incumplimiento de las normas que regulan el negocio de la organización, como los estándares de cumplimiento Payment Card Industry Data Security Standard (PCI-DSS), Health Insurance Portability and Accountability Act (HIPPA), Reglamento General de Protección de Datos Europeo (RGPD), etc. Los riesgos de incumplimiento afectan directamente a aspectos relacionados con la continuidad de negocio de las organizaciones, pues pueden verse expuestas a demandas, sanciones, retirada de facultades para desempeñar su función, etc. De forma equivalente a la estrategia de seguridad preventiva mencionada

anteriormente, la forma más eficiente de abordar este problema es adoptar un enfoque "*compliance by design"*. En este caso, al igual que sucede con los requisitos de seguridad, las especificaciones relacionadas con aspectos de cumplimiento normativo son identificadas ("*compliance by detection*") y gestionadas desde el principio del proceso. Así, los desarrolladores trabajan sobre dichos requisitos o especificaciones de la misma forma que lo hacen con el resto de requisitos y el cumplimiento normativo se aborda como una característica más del producto final, que será validado mediante las correspondientes pruebas de aceptación.

Los mismos conceptos expresados anteriormente para las vulnerabilidades, son aplicables en este caso para el cumplimiento normativo. Es decir, teniendo en cuenta las características de cumplimiento desde las primeras fases del ciclo de vida del desarrollo de software, se consiguen reducir los riesgos de negocio de la organización, los costes derivados de la reparación de fallos y un mejor producto final.

Por todo lo anterior, las organizaciones que directa o indirectamente desarrollen productos software, tanto para la ejecución o transformación de sus procesos internos como para la prestación de servicios relacionados con la cadena de valor, deberían adoptar una estrategia preventiva en la gestión del ciclo de vida del software, con el fin de mejorar la seguridad de sus productos, reducir costes asociados a la gestión de defectos y mantenimiento, mantener su reputación y reducir riesgos. En definitiva, para mejorar su competitividad y garantizar la continuidad de su negocio. Un reflejo de esta necesidad, lo muestra el informe de tendencias en seguridad realizado en 2019 por Cognizant Technology Solutions [6], donde la integración de la seguridad en el ciclo de vida del desarrollo de software aparece como una de las 4 principales tendencias, entre las organizaciones encuestadas.

2.2 Modelos de madurez

Existen diversas metodologías para la integración de la seguridad en el proceso de gestión del ciclo de vida del software. Algunos de los más conocidos son el Security Development Lifecycle (SDL) de Microsoft [7] , SAFE Code [8], Touchpoints de Gary McGraw [9]. Adicionalmente, están disponibles diferentes estándares que proporcionan orientaciones para implementar un proceso de desarrollo de software seguro, como la guía SP 800-160 del NIST "Systems Security Engineering Considerations for a Multidisciplinary Approach in the Engineering of Trustworthy Secure Systems" [10], ISO/IEC 15026-4 "Systems and software engineering - Systems and software assurance - Part 4: Assurance in the life cycle" [11], etc.

Todas las referencias mencionadas, pese a que son elementos de gran utilidad, no están pensadas para la medir o evaluar el grado de excelencia en el desarrollo de software seguro de una organización. El instrumento adecuado para ello son los denominados modelos de madurez. Un modelo de madurez proporciona a la organización una hoja de ruta que sirve de guía para la implementación de buenas prácticas, estableciendo el punto de partida, definiendo el camino evolutivo y permitiendo la evaluación de los progresos. Los modelos de madurez permiten a las organizaciones identificar debilidades, evolucionar aquellos aspectos que le proporcionan más valor y

trazar estrategias de mejora a medida para alcanzar los objetivos fijados por la propia organización, en un tiempo determinado y en función de sus intereses y necesidades de negocio.

El presente Trabajo, se centra en la aplicación de un modelo de madurez para la implementación de un proceso de gestión del ciclo de vida del software seguro.

3 Modelos de madurez del ciclo de desarrollo de software seguro

Un modelo de madurez es un instrumento que ayuda a evaluar la eficacia de los procesos o prácticas de personas, grupos u organizaciones y a determinar qué capacidades deben ser adquiridas o evolucionadas para mejorar su desempeño.

Los modelos de madurez contienen los elementos esenciales de procesos eficaces, para una o más disciplinas, y proporcionan una hoja de ruta para la mejora evolutiva, desde procesos ad-hoc e inmaduros, hasta procesos disciplinados y maduros de mayor calidad y eficacia [12].

En la industria existen diversos *frameworks* de modelos de madurez, relacionados con las tecnologías IT. En relación al desarrollo de software, quizá el más famoso es el modelo CMMI [12] del Software Engineering Institute (SEI) debido a que es el que fue de los primeros en publicarse (su origen es de finales de los años 80) y a que es ampliamente reconocido en la industria de desarrollo de software. Otro modelo bastante conocido y aplicado es el estándar ISO/IEC 15504, también conocido como *Software Process Improvement Capability Determination* (SPICE), publicado a finales de los noventa.

En ninguno de los dos *frameworks* mencionados se tratan de forma explícita la seguridad en el desarrollo de software, aunque por supuesto se pueden adaptar para cubrir aspectos de seguridad en el desarrollo de software. En ambos casos existen propuestas de adaptaciones o extensiones específicas que buscan cubrir dichos aspectos. Por ejemplo, en el caso de CMMI, existe una propuesta de filosofía "*security by design*" elaborada por Siemens [13] y en el caso de ISO, podemos encontrar alguna propuesta como "*Implementing information security best practices on software lifecycle processes: The ISO/IEC 15504 Security Extension*" [14], relacionando dicha norma con el estándar ISO/IEC 27002.

No obstante, aunque en los dos *frameworks* mencionados, existen adaptaciones para adaptar los modelos y tratar el tema de la seguridad, esto se realiza con un enfoque más cercano a la gestión de riesgos de seguridad en la propia gestión del proyecto y no en la seguridad final del producto resultante.

En relación al desarrollo de software seguro, encontramos dos modelos de madurez específicos que se centran en la seguridad en el ciclo de vida del software. Estos dos modelos son el Building Security in Maturity Model (BSIMM) de Synopsys y el Software Assurance Maturity Model (SAMM) de OWASP, objeto del presente Trabajo.

3.1 Breve descripción de BSIMM

BSIMM [15] es, en realidad, un estudio sobre las iniciativas de seguridad en el desarrollo de software, que llevan a cabo de forma habitual una serie de organizaciones, fundamentalmente empresas estadounidenses. En concreto, el modelo se construye a partir de los datos observados en 130 grandes empresas, recopilando las diversas iniciativas de seguridad del software, homogeneizando la información para describirlas de manera uniforme, a pesar de que estas puedan utilizar diferentes metodologías y terminología. Esta "estandarización" permite disponer de un *framework* cuantificable y utilizable por cualquier

organización, permitiendo comparar las iniciativas de seguridad a pesar de que operen a diversas escalas, se lleven a cabo en diferentes partes del organigrama o creen diferentes artefactos de trabajo.

El modelo BSIMM no es un modelo de madurez tradicional, en el que un conjunto de actividades se repite a lo largo de los diversos niveles de madurez. Por ejemplo, se hace algo en el nivel 1, se extiende la práctica en el nivel 2, se hace mejor en el nivel 3, etc. En el caso de BSIMM las actividades son únicas con niveles de actividad utilizados únicamente para mostrar la frecuencia relativa con la que las actividades se observan en las diferentes organizaciones. Así, las actividades que se observan con más frecuencia en el conjunto de organizaciones analizado se designan como nivel 1, las que se observan con menor frecuencia se designan como nivel 2 y las que se observan con muy poca frecuencia se designan como nivel 3. Es decir, se asume que hay un conjunto de prácticas que son realizadas por la mayoría de las organizaciones y que se considerarían como fundamentales (nivel inicial de madurez). Las actividades menos frecuentes, más avanzadas o complejas, se asociarían con niveles de madurez más altos.

BSIMM no establece ninguna obligación de llevar a cabo ciertas prácticas observadas en otras organizaciones ni prescribe “recetas”. Simplemente observa e informa de las prácticas de seguridad que se llevan a cabo en las organizaciones evaluadas. Cada organización determinará, en función de sus objetivos de seguridad, que prácticas va a ejecutar y si necesita llevar a cabo actividades adicionales. El conocimiento de lo que la mayor parte, algunas o conjunto reducido de organizaciones está haciendo será la base para orientar la estrategia de seguridad del software de la organización.

Según define el propio modelo, BSIMM es “*principalmente una vara de medir la seguridad del software*” [15], pues permite comparar y contrastar las iniciativas de seguridad de una organización respecto a lo que hacen las organizaciones analizadas. Esto proporcionará una idea del estado de seguridad del software de la organización, en relación al grado de madurez medio del resto de organizaciones. A continuación (Figura 4) se puede observar el resultado de comparar una organización de ejemplo con el conjunto de organizaciones analizadas por BSIMM.

Figura 4. Comparación de grado de madurez de organización de ejemplo con el global. Fuente BSIMM **[15]**

Adicionalmente, el modelo puede utilizarse para establecer una hoja de ruta del programa de seguridad en el software. Para ello se comparará nuestra organización con el resto, se evaluará la distancia que nos separa y se preparará el plan de acción oportuno para reducirla o eliminarla. De la misma forma, el modelo se podrá emplear para evaluar o medir la evolución en el tiempo de los esfuerzos invertidos en nuestro programa de seguridad.

3.2 Comparativa de BSIMM y SAMM

Lo primero que es necesario entender, y que constituye una de las diferencias principales entre los dos modelos, es que BSIMM es un modelo descriptivo, mientras que SAMM es un modelo prescriptivo. Como se comentó en el apartado anterior, BSIMM proporciona información sobre qué prácticas de seguridad está aplicando la industria en sus procesos de desarrollo, mediante el análisis de las iniciativas de seguridad de un grupo amplio de organizaciones. Según se define en el propio informe del modelo, "*BSIMM no dice "todas las casas deben tener aspiradores robot", "los robots son el único tipo de aspiradores aceptable", "los aspiradores deben usarse todos los días" o cualquier otro juicio de valor. Ofrecemos simples observaciones simplemente informadas*." [15]. Es decir, no indica lo que se debe hacer, sino que muestra lo que la industria hace con el fin de proporcionar orientaciones a las organizaciones que deseen poner en marcha un programa de seguridad en el software.

Por el contrario, SAMM sí define un conjunto de prácticas que se han de llevar a cabo para que una organización establezca un programa de seguridad que la ayude a tener un proceso de desarrollo de software más seguro. Las prácticas prescritas por SAMM son el resultado de un proceso de evaluación por parte de los equipos del proyecto OWASP que, sobre la base de su conocimiento y experiencia, determinan la eficacia de las actividades sugeridas por el modelo. Su objetivo es ayudar a las organizaciones a definir las actividades relacionadas

con la seguridad, a implementar un programa de seguridad por iteraciones y evaluar los progresos realizados.

En relación a los niveles de madurez, ambos modelos establecen 3 niveles de madurez, si bien se conciben de forma diferente. Como ya se comentó en el apartado anterior, BSIMM basa sus modelos de madurez según la frecuencia relativa en la que se observan diferentes prácticas en las organizaciones. Por su parte, SAMM establece un enfoque más tradicional, en el que las diferentes prácticas se repiten a lo largo de los niveles de madurez, con un incremento de la exigencia a medida que se avanza de nivel.

Otra de las diferencias importantes es que BSIMM es un modelo propietario de la compañía Synopsys, por lo que para realizar una evaluación es preciso contactar con dicha organización con el fin de presupuestar y planificar el proceso. Por su parte, SAMM es un modelo abierto y puede ser utilizado sin limitaciones. Además, OWASP ofrece diversas herramientas para realizar una autoevaluación e incluso realizar seguimiento de la evolución del programa de seguridad en el tiempo. Por supuesto, también se puede contratar la evaluación a empresas auditoras del modelo.

La frecuencia de actualización de los modelos también presenta ciertas diferencias. BSIMM lleva a cabo un análisis anual de las empresas que participan en el desarrollo del modelo, evaluando sus prácticas de seguridad. Por tanto, el modelo se actualiza anualmente[2]. En cambio, el modelo SAMM se actualiza cada 2-3 años[3].

En lo que respecta a la comparación del grado de madurez de una organización con respecto otras, BSIMM sí permite contrastar los resultados para conocer el estado de seguridad de una organización en relación con el resto de empresas que participan en este *framework.* De hecho, tal y como ellos mismos reconocen, este es unos principales objetivos de BSIMM. Esto es algo que por el momento no es posible realizar con SAMM, aunque, según se informa en su espacio Web, es algo en lo que están trabajando [16].

Con respecto a su organización y estructuración, los modelos presentan bastante similitud, pues se definen unas categorías de alto nivel (dominios o funciones según el modelo), cada una de las cuales contienen un conjunto de prácticas (3 por dominio o función), que a su vez incluyen un conjunto de actividades. Obviamente la estructura organizativa no es exactamente la misma, pero se pueden establecer equivalencias y mapeos entre los dos modelos [17].

A continuación se muestra un resumen comparativo (**Tabla 3**) de la estructura organizativa de ambos modelos.

---

[2] En la fecha de elaboración del presente Trabajo, está vigente BSIMM11. Es decir, estamos ante la undécima iteración del modelo.

[3] En la fecha de elaboración del presente Trabjo, versión más reciente del modelo es SAMM 2.0, liberada en enero del 2020.

Elemento	BSIMM	SAMM
Dominios / Funciones de negocio	**Dominios** • Gobierno • Inteligencia • Secure SDL Touchpoints • Despliegue	**Funciones de negocio** • Gobierno • Diseño • Implementación • Verificación • Operaciones
Prácticas	Tres prácticas por dominio. **Gobierno** • Estrategia y Métricas • Cumplimiento y Política • Formación **Inteligencia** • Modelos de ataques • Funciones de seguridad y diseño • Estándares y Requisitos **Secure SDL Touchpoints** • Análisis de arquitectura • Revisión de código • Pruebas de seguridad **Despliegue** • Test de penetración • Entorno software • Gestión de configuración / Gestión de vulnerabilidades	Tres prácticas por función de negocio **Gobierno** • Estrategia y Métricas • Política y Cumplimiento • Formación y orientación **Diseño** • Evaluación de amenazas • Requisitos de seguridad • Arquitectura de seguridad **Implementación** • Construcción segura • Despliegue seguro • Gestión de defectos **Verificación** • Análisis de arquitectura • Pruebas orientadas a requisitos • Pruebas de seguridad **Operaciones** • Gestión de incidentes • Gestión del entorno • Gestión operativa
Actividades	Entre 7 y 12 actividades por práctica	2 líneas de acción complementarias por práctica, por 3 niveles de madurez. Es decir, 6 actividades por práctica

Tabla 3. Comparación estructura organizativa BSIM vs SAMM

4 Modelo SAMM de OWASP

4.1 Introducción.

OWASP SAMM (Software Assurance Maturity Model) es un modelo de madurez enfocado en la seguridad del proceso de construcción del software. Constituye un marco de trabajo (*framework*) cuya finalidad es ayudar a las organizaciones a establecer e implementar una estrategia para la mejora de la seguridad en los procesos del ciclo de vida del software. Es decir, para desarrollar software más seguro.

El modelo considera la seguridad en el proceso de desarrollo de forma holística, teniendo en cuenta las diferentes funciones o subprocesos que intervienen en el ciclo de vida del desarrollo del software y estableciendo, para cada una de ellas, un conjunto de prácticas y actividades orientadas a la mejora de la seguridad.

SAMM es una herramienta de gran utilidad para evaluar la madurez de las organizaciones en lo que a la seguridad en el desarrollo de software se refiere, así como para establecer un programa estratégico de mejora incremental de la seguridad de los productos software finales. En definitiva, SAMM ayudara a las organizaciones a:

- Evaluar las prácticas de seguridad de los diversos procesos llevados a cabo a lo largo del ciclo de vida de desarrollo de software y medir el grado de madurez de la organización en este aspecto.
- Evidenciar la ejecución de buenas prácticas en el desarrollo de software.
- Definir las prácticas y actividades que se llevarán a cabo para la mejora de la seguridad en el proceso de desarrollo de software.
- Desarrollar un programa incremental e iterativo de aseguramiento de la seguridad del software.
- Demostrar progresos concretos del programa de mejora de la seguridad en el desarrollo de software.

4.2 El modelo.

SAMM es un modelo prescriptivo (vs descriptivo) que establece de forma concreta un conjunto de buenas prácticas que se deben llevar a cabo en el proceso de desarrollo de software para garantizar un nivel adecuado de la seguridad del producto final.

El modelo se ha desarrollado con un enfoque orientado al riesgo que permite a las organizaciones ajustar y personalizar las buenas prácticas a sus necesidades en función de su perfil y tolerancia al riesgo.

Adicionalmente, SAMM se ha elaborado con un planteamiento agnóstico en cuanto al paradigma de desarrollo que cada organización tenga establecido, aceptando metodologías de desarrollo en cascada, iterativas, *agile*, DevOps, etc. Por tanto, puede ser empleado por cualquier tipo de organización.

Como se ha comentado anteriormente, SAMM proporciona un medio para conocer en qué estado se encuentra una organización y determinar qué pasos dar para elevar su nivel de madurez al siguiente nivel. Los “pasos” a dar

consisten en llevar a cabo diferentes prácticas concretas durante el ciclo de vida del desarrollo que conducirán a mejorar el estado de seguridad de los productos finales. Estas prácticas se agrupan en *funciones de negocio* (*business functions*) que se corresponden con categorías de actividades relacionadas con el desarrollo de software que cualquier organización que se dedique o realice desarrollo de software debe cumplir en mayor o menor medida.

Las *funciones de negocio* son los elementos de más alto nivel en los que se fundamenta el modelo. A continuación se enumeran las cinco *funciones de negocio* definidas en SAMM:

1. Gobierno
2. Diseño
3. Implementación
4. Verificación
5. Operación

Para cada una de las funciones enumeradas, SAMM define tres prácticas cuya finalidad es garantizar la seguridad de dicha función de negocio.

En cada práctica de seguridad, se definen a su vez tres niveles de madurez. Cada nivel persigue un objetivo más sofisticado que el anterior, con actividades más exigentes y métricas más estrictas.

Adicionalmente, para cada una de las prácticas SAMM define dos líneas de acción que agrupan varias actividades. Cada línea de acción persigue un objetivo que puede alcanzarse con cierto nivel de madurez. Las líneas de acción vinculan las diferentes actividades de la práctica con los niveles de madurez establecidos. En la Figura 5 se muestra estructura organizativa de los elementos que forman el modelo:

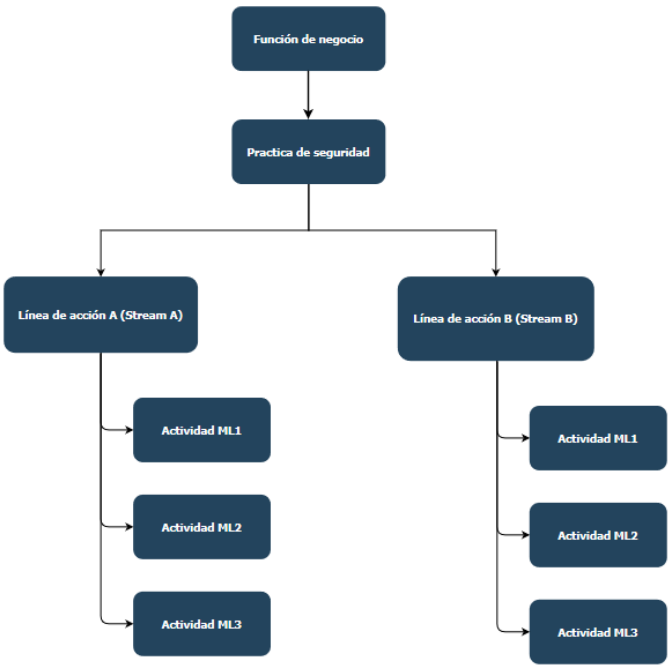


Figura 5. Estructura organizativa de SAMM.

Respecto al modo de aplicar el modelo, SAMM no obliga a las organizaciones a alcanzar el máximo nivel de madurez en cada una de las categorías. El modelo está pensado para que cada organización ajuste la aplicación del modelo según sus intereses y circunstancias específicas, determinando el nivel de madurez objetivo para cada una de las categorías.

En la siguiente figura, se muestra un diagrama del modelo donde se pueden observar diversos elementos que componen el modelo: *funciones de negocio*, prácticas de seguridad, y líneas de acción (*Streams*):

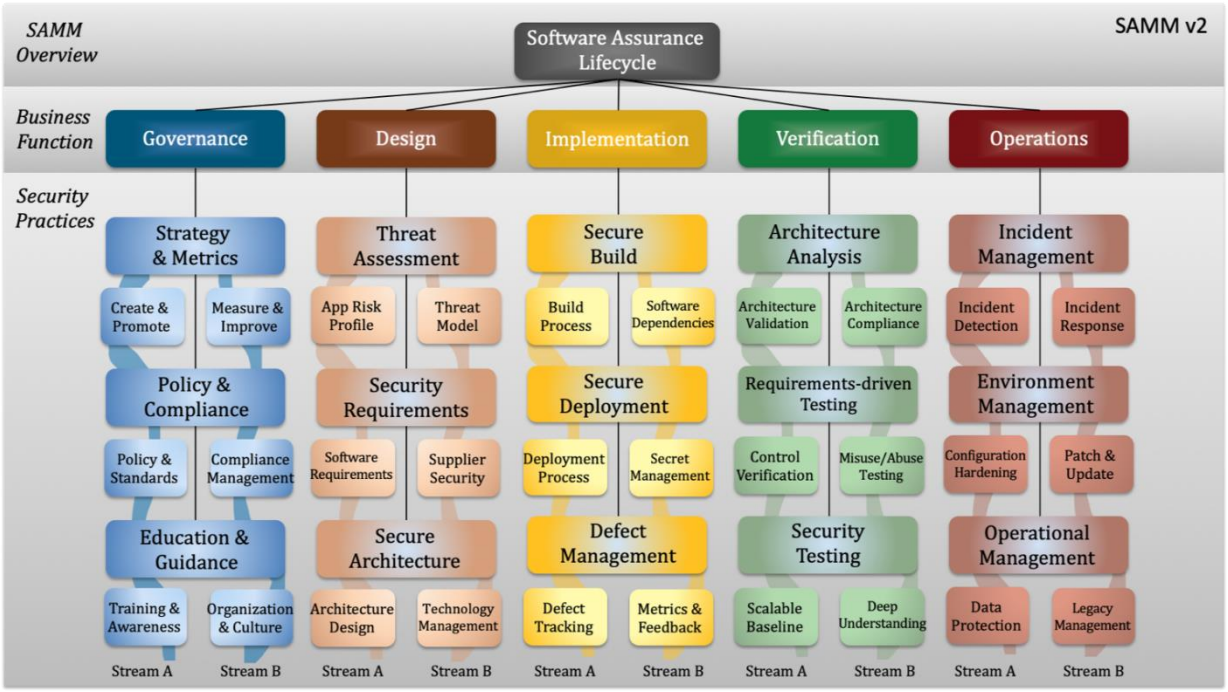


Figura 6. Modelo SAMM v2.0. Fuente OWASP SAMM

En definitiva, SAMM define un total de 90 actividades relacionadas con la seguridad de diverso nivel de madurez, agrupadas en 15 prácticas de seguridad que corresponden a cada una de las 5 funciones de negocio sobre las que se fundamenta el modelo.

En los siguientes apartados se describe en detalle los diversos elementos sobre los que se fundamenta SAMM.

4.2.1 Función de negocio: Gobierno de la seguridad

Esta función se centra en las actividades relacionadas con la manera en la que la organización gestiona el proceso de desarrollo de software, incluyendo aquellos aspectos que afectan a los diversos grupos funcionales implicados en el desarrollo de software, así como los procesos generales de gestión de la organización.

Las tres prácticas comprendidas en el ámbito de la función de gobernanza del desarrollo seguro son:

- **Estrategia y métricas**. Constituye los cimientos de todas las actividades orientadas a la gestión de la seguridad en el desarrollo de software mediante el establecimiento de un plan general dentro de la organización.
- **Política y cumplimiento.** Impulsa el cumplimiento de las normas y regulaciones internas y externas en materia de seguridad por parte de los productos resultantes de la actividad de desarrollo.
- **Formación y orientación.** Fomenta el conocimiento dentro de la organización con respecto al software seguro.

A continuación se describirá cada una de las tres prácticas que sustentan la función de Gobierno del modelo SAMM.

4.2.1.1.1 Práctica Estrategia y Métricas (SM)

El aseguramiento del proceso de desarrollo de software requiere tener en consideración un buen número de actividades y prestar atención a un conjunto de cuestiones nada despreciable. Para obtener éxito en esta misión, es fundamental disponer de una estrategia y un plan general que asegure el alineamiento, proporcionalidad y efectividad de las actividades de mejora. De lo contrario, se corre el riesgo de dedicar muchos esfuerzos a incorporar seguridad en los procesos pero que estos estén desalineados, sean desproporcionados o, incluso, contraproducentes.

El objetivo de la práctica de Estrategia y Métricas (SM) es la elaboración de un plan eficiente y efectivo para alcanzar los objetivos de seguridad en el software de la organización. Dicho plan o programa de mejora de la seguridad deberá seleccionar las prácticas y actividades que se llevarán a cabo, priorizándolas conforme a las necesidades y objetivos de la organización. El programa de seguridad actuará como base para dirigir y gestionar los esfuerzos de la organización.

En la siguiente figura, se muestran las actividades asociadas a los distintos niveles de madurez comprendidos en la práctica de Estrategia y Métricas.

Maturity level		Stream A Create and Promote	Stream B Measure and Improve
1	Identify objectives and means of measuring effectiveness of the security program.	Identify organization drivers as they relate to the organization's risk tolerance.	Define metrics with insight into the effectiveness and efficiency of the Application Security Program.
2	Establish a unified strategic roadmap for software security within the organization.	Publish a unified strategy for application security.	Set targets and KPI's for measuring the program effectiveness.
3	Align security efforts with the relevant organizational indicators and asset values.	Align the application security program to support the organization's growth.	Influence the strategy based on the metrics and organizational needs.

Figura 7. Actividades de la práctica de Estrategia y Métricas. Fuente OWASP SAMM.

Como se comentó anteriormente, SAMM define, para cada una de las prácticas, dos líneas de acción que agrupan actividades con objetivos específicos. En la Figura 7, se observan las dos líneas de acción de esta práctica: Crear y Fomentar (Stream A) y Medir y Mejorar (Stream B).

La línea que agrupa las actividades relacionadas con la estrategia (Stream A) tiene como finalidad la de proporcionar las directrices necesarias para la construcción del programa de mejora de la seguridad, así como para el mantenimiento y difusión del mismo.

Por su parte, el la línea de acción asociada a medir y mejorar (Stream B) está orientado al seguimiento del estado de la seguridad y los avances del programa, estableciendo un conjunto de métricas que proporcionan una visión precisa del grado de implantación de las medidas aplicadas.

Las actividades relacionadas con la definición de indicadores y métricas son fundamentales para el éxito del programa. Recordemos la famosa frase del físico William Thomson Kelvin: "*Lo que no se define no se puede medir. Lo que no se mide, no se puede mejorar. Lo que no se mejora, se degrada siempre.*"

A continuación, se describirá en detalle cada una de las seis actividades (dos por cada nivel de madurez) comprendidas en la práctica de Estrategia y Métricas.

4.2.1.1.2 Estrategia y Métricas (SM1)

El objetivo de la práctica en este nivel de madurez es identificar los objetivos y los mecanismos para medir la eficacia del programa de seguridad.

Línea de acción "Crear y Fomentar" (Stream A): Identificar la tolerancia al riesgo de la organización.

El objetivo de esta actividad es reflexionar y entender la propia postura de la organización en lo que respecta a la seguridad.

Basándose en la exposición al riesgo de las aplicaciones, se analizarán qué amenazas existen o pueden existir, así como el grado de tolerancia al riesgo por parte de la dirección. Este ejercicio es clave para determinar las prioridades de las acciones que han de llevarse a cabo para garantizar la seguridad del software. Para determinar las amenazas a las que se exponen las aplicaciones amenazas, se entrevistará a los responsables ejecutivos de la organización y a otras partes interesadas. Se documentarán los factores clave del sector donde opera la organización, así como los factores específicos de la organización. La información recopilada incluirá los peores escenarios que podrían afectar a la organización, así como las oportunidades que un ciclo de vida de desarrollo de software optimizado y aplicaciones más seguras podrían proporcionar a la organización.

La información recopilada proporcionará una línea de base para que la organización desarrolle y fomente el programa de seguridad de aplicaciones. Los elementos y actividades del programa se priorizarán conforme a los intereses de la organización para abordar las amenazas y oportunidades más importantes.

La información extraída de los escenarios de riesgo planteados se utilizará para construir un perfil de riesgo de cada una de las aplicaciones desarrolladas por la organización, documentando cómo esta puede verse afectada si dichas aplicaciones se ven comprometidas.

Tanto la línea de base como los factores de riesgo propios de la organización, deben ser aprobados por la dirección, publicados y puestos a disposición de los equipos de desarrollo para garantizar un proceso de creación de perfiles de riesgo de las aplicaciones más transparente e incorporar las prioridades de la organización en el programa de mejora de la seguridad.

Línea de acción "Medir y Mejorar" (Stream B): Definir unas métricas básicas de seguridad.

El objeto de esta actividad es que la organización disponga de información básica sobre la eficacia y la eficiencia de su programa de seguridad en el software.

Se definirán y documentarán una serie de métricas para evaluar la eficacia y la eficiencia del programa de seguridad del software. De esta manera, las mejoras mejoras realizadas son medibles y pueden utilizarse para realimentar y asegurar el apoyo y financiación del programa en el futuro. Las métricas deben incluir medidas de las siguientes categorías:

- Esfuerzo. Las métricas de esfuerzo miden el esfuerzo dedicado a la seguridad. Por ejemplo, las horas de formación, el tiempo dedicado a la

revisión del código, el número de aplicaciones analizadas en busca de vulnerabilidades, etc.

- Resultados. Las métricas de resultados miden los resultados de los esfuerzos de seguridad. Algunos ejemplos son el número de defectos de seguridad sin parchear y el número de incidentes de seguridad relacionados con vulnerabilidades de las aplicaciones desarrolladas.
- Entorno. Las métricas de entorno miden el entorno en el que tienen lugar los esfuerzos de seguridad. Los ejemplos incluyen el número de aplicaciones, número de líneas de código, etc., como medida de dificultad o complejidad.

Cada medida por sí misma es útil para un propósito específico, pero la combinación de dos o tres métricas ayudara a explicar mejor los picos en las tendencias. Por ejemplo, un pico en el número total de vulnerabilidades puede deberse a que la organización está incorporando varias aplicaciones nuevas que no han sido desarrolladas de forma segura, un aumento en las métricas del entorno sin que incremente el esfuerzo o el número de defectos, podría ser un indicador de un programa de seguridad maduro y eficiente.

En la identificación de métricas se recomienda seleccionar aquellas que cumplan varios criterios:

- Que se midan de forma consistente
- Poco costosas de recopilar
- Expresadas como un número cardinal o un porcentaje
- Expresadas como una unidad de medida

Se deberán desarrollar, documentar y publicar las métricas elegidas y se describirán lo métodos más eficaces para la recopilación de datos y los métodos recomendados para combinar medidas individuales en métricas significativas. Por ejemplo, el número total de aplicaciones y el número total de defectos en todas las aplicaciones pueden no ser útiles por sí mismos, pero cuando se combinan como el número de defectos graves por aplicación, proporcionan una métrica más valiosa.

4.2.1.1.3 Estrategia y Métricas (SM2)

El objetivo de la práctica en este nivel de madurez es establecer la hoja de ruta global para mejorar la seguridad del software dentro de la organización.

Línea de acción "Crear y Fomentar" (Stream A): Definir la estrategia de seguridad.

El objeto de esta actividad es establecer y comunicar la hoja del programa de mejora de la seguridad en el desarrollo de software.

Se desarrollará un plan estratégico de mejora de la seguridad conforme a la importancia de los activos, las amenazas y la tolerancia al riesgo de la organización. Dicho plan tendrá asignado el presupuesto correspondiente para acometer las acciones necesarias para la mejora de la seguridad, y estará alineado con las prioridades de negocio.

El plan de mejora de la seguridad abarcará un período de uno a tres años, e incluirá los hitos coherentes con los objetivos de la organización y los riesgos detectados. El plan definirá diferentes iniciativas de carácter táctico y estratégico, proporcionando una hoja de ruta que muestra el alineamiento del plan con las prioridades y necesidades de la organización.

En la hoja de ruta establecida se buscará un equilibrio entre los cambios que implican un coste económico, cambios en procesos y procedimientos y cambios relacionados con la cultura de la organización. Este equilibrio facilitará que se lleven a cabo diversas acciones de forma simultánea sin agotar los recursos económicos disponibles o sobrecargar a los equipos de desarrollo.

Se definirán hitos con la frecuencia suficiente, de tal forma que se facilite la supervisión del desarrollo del programa y se realicen los ajustes oportunos en la hoja de ruta en caso necesario.

Para asegurar el éxito del programa de mejora, se obtendrá aprobación por las diversas partes interesadas de la organización y de los equipos de desarrollo. Una vez aprobado, el plan de mejora será publicado y estará a disposición de cualquiera que tenga que participar o apoyar su desarrollo.

Línea de acción "Medir y Mejorar" (Stream B): KPIs estratégicos

El objetivo es obtener visibilidad sobre el desempeño del programa de mejroa de la seguridad del software.

Una vez que la organización ya tiene definidas las métricas básicas de seguridad de sus aplicaciones (llevado a cabo en SM1), se recopilará la información suficiente para establecer objetivos de mejora realistas.

Se planificará una fase de pruebas inicial para verificar que las métricas identificadas pueden ser recopiladas de forma coherente y eficiente. Tras un corto período de prueba y una vez validadas las métricas, se establecerá una serie de metas y objetivos en forma de indicadores clave de rendimiento (KPI).

Los KPIs se construirán sobre métricas que se consideren valiosas, tanto por parte de los responsables de seguridad como por los responsables de las aplicaciones y por la dirección de la organización. Se deberá prestar atención y eliminar la influencia que puedan tener sobre los KPIs la volatilidad habitual de los entornos de desarrollo de aplicaciones, con el fin de reducir la posibilidad de falsear u obtener resultados desfavorables que sean consecuencia de mediciones engañosas.

Los KPIs definidos constituirán el instrumento de medida con el que se evaluará el éxito de todo el programa de mejora. Por ello, se documentarán debidamente y se distribuirán a los todos aquellos que contribuyen al éxito del programa, así como a la dirección. Para documentar cada KPI, se incluirá una breve explicación de las fuentes de información en las que se basa y el significado si las cifras son altas o bajas. Adicionalmente, para cada KPI se definirán objetivos a corto, medio y largo plazo, así como umbrales que determinen valores inaceptables que requieran una intervención inmediata.

4.2.1.1.4 Estrategia y Métricas (SM3)

El objetivo de la práctica en este nivel de madurez es alinear los esfuerzos realizados en actividades de mejora de la seguridad con los valores de los activos e indicadores globales de la organización.

Línea de acción "Crear y Fomentar" (Stream A): Alinear estrategias de seguridad y negocio

El objetivo es lograr una alineación continua del programa de seguridad con los objetivos empresariales.

El plan de seguridad se revisará de forma periódica para revalidar su conformidad con los objetivos de la organización y garantizar su adaptación a las necesidades cambiantes y crecimiento futuro de la organización. Para ello, se repetirán los pasos asociados a las actividades de los dos primeros niveles de madurez de la Práctica Estrategia y Métricas (SM). Esto se realizará, al menos, una vez al año.

Además de revisar los objetivos y necesidades de la organización, se supervisará de cerca el cumplimiento de cada uno de los hitos de la hoja de ruta establecida. Se analizará los hitos no alcanzados o insatisfactorios y se evaluarán los posibles cambios en el programa de mejora global.

Se desarrollarán cuadros de mando para que la dirección de la organización y los responsables del desarrollo de software puedan realizar el seguimiento adecuado sobre el cumplimiento de los hitos u objetivos intermedios. Estos cuadros de mando deberán proporcionar suficiente detalle como para identificar situación de proyectos e iniciativas individuales, así como para proporcionar una clara visión de si el programa está desarrollándose con exitoso o si está alineado con las necesidades de la organización.

Línea de acción "Medir y Mejorar" (Stream B): Impulsar el programa de seguridad mediante métricas

El objetivo es llevar a cabo la mejora continua del programa de seguridad basándose en los resultados obtenidos a través de las métricas.

Se establecerán directrices para modificar o influir sobre el programa de seguridad en función de los KPIs y otras métricas. Estas directrices combinarán la madurez del proceso de desarrollo con diferentes métricas con el fin de hacer que el programa de mejora sea cada vez más eficiente. A continuación se muestran algunos ejemplos que relacionan las métricas con formas de mejorar el programa:

- Centrarse en la madurez del ciclo de vida del desarrollo y aplicar seguridad de forma proactiva, provoca que el coste relativo por defecto sea menor.
- Monitorizar la relación entre las métricas de esfuerzo, resultados y entorno, mejora la eficacia del programa y justifica la automatización de los procesos.
- Algunas prácticas individuales podrían proporcionar, a través de sus indicadores, información para determinar el éxito o fracaso de ciertas iniciativas.
- Las métricas de esfuerzo ayudarán a asegurar que los trabajos se enfocan o dirigen hacia las técnicas y disciplinas más beneficiosas.

Por lo comentado, al definir la estrategia global de métricas, se tendrá siempre en consideración el objetivo final perseguido y se definirán qué decisiones pueden tomarse basándose en los KPIs y métricas lo antes posible.

4.2.1.2 Práctica Política y Cumplimiento (PC)

La práctica de Políticas y Cumplimiento (PC) se centra en comprender y cumplir los requisitos legales y reglamentarios externos, a la vez que impulsa las normas de seguridad internas para garantizar el cumplimiento de las obligaciones de la organización de forma alineada con los objetivos de negocio.

Uno de los elementos que fomenta el cumplimiento de los objetivos de esta práctica es la definición, descripción y publicación de las normas internas de la organización y otras obligaciones impuestas por terceros. Dichas normas y obligaciones tomarán la forma de requisitos para las aplicaciones, lo que asegurará su cumplimiento y facilitará la realización de auditorías que demuestren de forma continua que se satisfacen las expectativas en materia de seguridad en el ciclo de desarrollo del software.

En su forma más sofisticada o avanzada, la realización de esta práctica implica el conocimiento amplio y profundo de toda la organización, de sus normas internas y de los requisitos externos que le aplican, a la vez que se establecen puntos de control continuos que garantizan que ningún proyecto transcurre al margen de las expectativas sin que sea detectado.

En la siguiente figura, se muestran las actividades asociadas a los distintos niveles de madurez comprendidos en la práctica de Política y Cumplimiento.

Maturity level		Stream A Policy & Standards	Stream B Compliance Management
1	Identify and document governance and compliance drivers relevant to the organization.	Determine a security baseline representing organization's policies and standards.	Identify 3rd-party compliance drivers and requirements and map to existing policies and standards.
2	Establish application-specific security and compliance baseline.	Develop security requirements applicable to all applications.	Publish compliance-specific application requirements and test guidance.
3	Measure adherence to policies, standards, and 3rd-party requirements.	Measure and report on the status of individual application's adherence to policies and standards.	Measure and report on individual application's compliance with 3rd party requirements.

Figura 8. Actividades de la práctica de Política y Cumplimiento. Fuente OWASP SAMM

Como se observa en la Figura 8, se definen dos líneas de acción para esta práctica: Política y Estándares (Stream A) y Gestión de la Conformidad (Stream B).

Las actividades asociadas a la línea de políticas y estándares (Stream A) se enfocan en la elaboración y mantenimiento de políticas y normas, y ponerlas a disposición de los equipos de desarrollo con el fin de que se incorporen desde el inicio en los proyectos de desarrollo de software de la organización.

Las actividades asociadas a la línea de gestión de la conformidad se centran en la identificación y provisión de los requisitos de cumplimiento normativo a los equipos de desarrollo, con el fin de que éstos sean incorporados en los proyectos de la organización desde su inicio.

A continuación, se describirá en detalle cada una de las seis actividades (dos por cada nivel de madurez) comprendidas en la práctica de Política y Cumplimiento.

4.2.1.2.1 Política y Cumplimiento PC1

El objetivo de la práctica en este nivel inicial de madurez es identificar y documentar los elementos clave relacionados con la gobernanza y el cumplimiento de las obligaciones de la organización.

Línea de acción "Política y Estándares" (Stream A): Definir políticas y estándares de seguridad

El objetivo de la actividad es definir, establecer y unos mínimos de seguridad para el desarrollo de software de la organización.

Se desarrollará un conjunto de políticas y normas que rijan sobre los diversos aspectos relacionados con el desarrollo de software dentro de la organización. Estas estarán basadas en los estándares aplicables en el sector de la organización.

Dada la amplia variedad de normas, estándares y mejores prácticas existentes, se consultará con los diversos equipos de producto la selección de normas y estándares que se aplicarán dentro de la organización. Adicionalmente, se procurará que los equipos realicen comentarios y observaciones sobre cualquier aspecto de las normas que no sea factible o rentable aplicar, así como las oportunidades de ampliar la extensión de las normas en algunos puntos que se puedan mejorar con poco esfuerzo por parte de los equipos de producto.

En relación a las políticas, la organización se deberá enfocar en las cuestiones de alto nivel y en aspectos que no dependan de tecnologías o contextos tecnológicos específicos. Es decir, se centrará en los objetivos de alto nivel de la organización en relación a la protección de la integridad del entrono IT, la seguridad y la privacidad de los datos, y la madurez de los ciclos de vida de desarrollo del software

En el caso de las normas o estándares, se incorporarán los requisitos establecidos por las políticas. Los estándares proporcionarán guías de implementación específicas para aprovechar las características de seguridad de los distintos lenguajes, *frameworks* y otras tecnologías que se empleen en la organización. Para la elaboración de los estándares se contará con la aportación de desarrolladores, arquitectos y personal experto de la organización en las distintas tecnologías empleadas.

Los estándares se elaborarán en un formato que permita y facilite su actualización periódica. Además, se etiquetarán y relacionarán los requisitos definidos en el estándar con las políticas, la normativa de aplicación y otros requisitos de alto nivel, para facilitar el mantenimiento de los estándares y la ejecución de auditorías.

Línea de acción "Gestión Cumplimiento" (Stream B): Identificar los requisitos de cumplimiento

El objetivo es asegurar que las políticas y estándares de la organización estén alineados con los requisitos de cumplimiento que apliquen a la organización.

Se elaborará una relación exhaustiva de todos los requisitos de cumplimiento que apliquen a la organización, así como de todos los factores que ayuden a determinar las aplicaciones que quedan dentro del alcance. Los requisitos de cumplimiento pueden ser de aplicación en función de diversos factores, como la

ubicación geográfica, el tipo de datos que maneja una aplicación, obligaciones contractuales con clientes o socios comerciales, etc,

Se revisará y analizará cada uno de los requisitos de cumplimiento identificado por parte del personal experto y/o el departamento legal correspondiente, con el fin de asegurar que la organización comprende bien las obligaciones derivadas de los requisitos. Hay que tomar en consideración que la aplicabilidad de algunos requisitos de cumplimiento puede variar en función de determinadas circunstancias que están bajo el control de la organización, por lo que los responsables de cumplimiento deberían aprovechar para identificar oportunidades de reducir la carga global de cumplimiento introduciendo o sugiriendo cambios en los procesos. Por ejemplo, la aplicabilidad de requisitos de cumplimiento en materia de privacidad, varía en función de cómo se procesan, almacenan o transmiten los datos, y cambiando la forma en que estos se gestionan se puede reducir el número o la complejidad de los requisitos de aplicación.

Se evaluará la publicación de una matriz de cumplimiento para ayudar a identificar qué factores podrían poner una aplicación en el ámbito de un requisito de cumplimiento específico. Esta matriz indicará además qué requisitos de cumplimiento son aplicables a nivel de la organización y no son específicos de las aplicaciones individuales. Disponer de una matriz de cumplimiento proporciona, al menos, una visión básica de los requisitos de cumplimiento para revisar las obligaciones que afectan a las diferentes aplicaciones.

Muchos de los requisitos de cumplimiento están derivados de las guías de mejores prácticas, por lo que es posible que muchos de ellos ya formen parte del conjunto de políticas y estándares internos de la organización. Por ello, una vez analizados los requisitos de cumplimiento, se realizará un mapeo de los mismos con las políticas y estándares de la organización. En caso de que haya requisitos de cumplimiento que no estén relacionados en la correspondiente política o estándar de la organización, se actualizarán estas últimas para incluir dichos requisitos. A partir de aquí, se crearán normas o estándares que traten de forma específica los requisitos de cumplimiento individuales.

El objetivo final es que la organización disponga de una matriz de cumplimiento que indique qué políticas y estándares contienen la información más detallada sobre los requisitos de cumplimiento, así como garantizar que las políticas y estándares de la organización hagan referencia a los requisitos de cumplimiento aplicables.

4.2.1.2.2 Política y Cumplimiento PC2

El objetivo de la práctica en este nivel es establecer una línea de base en materia de requisitos de seguridad y cumplimiento normativo, específico para las aplicaciones.

Línea de acción "Política y Estándares" (Stream A): Desarrollar procedimientos de prueba de requisitos de seguridad

El objetivo de esta actividad es proporcionar las directrices oportunas a los equipos de desarrollo sobre cómo llevar a cabo el cumplimiento de las políticas y estándares de seguridad.

Se desarrollarán y definirán los controles de seguridad y los procedimientos o guiones de pruebas apropiados para ayudar a los equipos de trabajo en la aplicación y verificación continua del cumplimiento de las políticas y estándares de seguridad definidos. Estos documentos e información se organizarán y agruparán en bibliotecas y se pondrán a disposición de los equipos de trabajo en los formatos más adecuados para su utilización en el desarrollo de cada aplicación. Los documentos estarán claramente etiquetados, relacionados y vinculados con las políticas y estándares de los que derivan, con el fin de facilitar su actualización y mantenimiento continuo.

Los requisitos de seguridad deberán ser elaborados en un formato coherente con los procesos de gestión de requisitos que lleve a cabo la organización, adaptándolos si es necesario a las diferentes metodologías o tecnologías de desarrollo empleadas. Podrá ser de utilidad la elaboración de *checklists* con los requisitos de seguridad aplicables. El objetivo es facilitar a los distintos equipos de trabajo la incorporación en el ciclo de vida del desarrollo de los requisitos establecidos por políticas y estándares, necesitando únicamente una interpretación mínima de dichos requisitos.

Los procedimientos o guiones de pruebas, por una parte facilitan la comprensión de los requisitos de seguridad estableciendo expectativas claras sobre la funcionalidad esperada, y por otra, sirven de guía para la elaboración de las pruebas manuales o automáticas. Las pruebas desarrolladas ayudan a los equipos de trabajo a determinar el estado actual de cumplimiento de las políticas y estándares y, además, garantizan el cumplimiento en futuras actualizaciones de las aplicaciones.

Línea de acción "Gestión Cumplimiento" (Stream B): Estandarizar la política y los requisitos de cumplimiento

El objetivo de la actividad es proporcionar una visión clara a los equipos de desarrollo sobre cómo lograr la conformidad con los requisitos de cumplimiento aplicables.

Se desarrollará una biblioteca de requisitos y procedimientos o guiones de prueba que sirvan para establecer los requisitos de cumplimiento normativo en las aplicaciones y verificar su conformidad. Algunos de los requisitos estarán asociados a requisitos de cumplimiento particulares como PCI o RGPD, mientras que otros serán de naturaleza más general como puede ser ISO. La biblioteca deberá estar a disposición de todos los equipos de desarrollo de aplicaciones. En la biblioteca se deberá proporcionar orientación para identificar todos los requisitos aplicables, incluyendo consideraciones que puedan tenerse en cuenta para reducir la carga y alcance de cumplimiento por parte de las aplicaciones.

Se deberá instaurar un proceso de revisión periódica de los requisitos de cumplimiento aplicables a cada aplicación. Esta reevaluación incluirá la revisión de toda la funcionalidad de la aplicación y la identificación de oportunidades para reducir la carga y alcance de cumplimiento normativo con el fin de disminuir el coste global asociado.

Los requisitos de cumplimiento deberán incluir suficiente información y nivel de detalle para que los desarrolladores entiendan los requisitos funcionales y no funcionales derivados de las diferentes obligaciones de cumplimiento. La documentación de los requisitos incluirá referencias a las políticas y estándares

de la organización, así como a la normativa de la que derivan los requisitos. De esta forma, si hay dudas sobre la aplicación de un requisito concreto, el texto original de la normativa puede ayudar a interpretar dicho requisito con mayor precisión.

Para cada uno de los requisitos se especificará un conjunto de procedimientos o guiones de prueba que sirvan para verificar su cumplimiento. Esta práctica, además de ayudar al departamento de control de calidad y cumplimiento, sirven para facilitar la comprensión de los requisitos, estableciendo expectativas claras sobre la funcionalidad esperada. Adicionalmente, hacen que el proceso de cumplimiento sea totalmente transparente.

4.2.1.2.3 Política y Cumplimiento PC3

El objetivo de la práctica en este nivel es evaluar y medir el cumplimiento de las políticas, los estándares internos y los requisitos de terceros en los proyectos de desarrollo de la organización.

Línea de acción "Política y Estándares" (Stream A): Medir cumplimiento de las políticas y estándares de seguridad.

El objetivo de la actividad es que la organización alcance una visión clara y precisa del grado de cumplimiento de las políticas y estándares de seguridad por parte de las aplicaciones que se desarrollen en la organización.

Se desarrollará un programa para medir de forma objetiva el cumplimiento de las políticas y estándares de seguridad en las aplicaciones. Para ello, se notificarán de forma sistemática todos los requisitos obligatorios y se motivará a los equipos de desarrollo para su cumplimiento. Siempre que sea posible, se vinculará el estado de conformidad con los requisitos mediante pruebas e informes de cumplimiento automáticos para las diferentes versiones de las aplicaciones. Dichos informes incluirán la versión de las políticas y estándares que se han evaluado así como los valores de cobertura de código correspondientes. Los informes de cumplimiento se comunicarán debidamente a las partes interesadas.

En el caso de que se identifique algún equipo de trabajo que no esté dando conformidad a los requisitos, se le invitará a revisar los recursos disponibles, como por ejemplo, los requisitos de seguridad y los procedimientos o *scripts* de pruebas, para asegurar de que el incumplimiento no viene motivado por la ausencia de conocimiento o la existencia de directrices inadecuadas. Si este fuera el caso, se comunicarán el problema detectado a los responsables de publicar los catálogos de requisitos y procedimientos de prueba, para mejorar la documentación en las futuras versiones. Adicionalmente, se elevará a los equipos que gestionan los riesgos de seguridad de las aplicaciones cualquier problema de cumplimiento que sea detectado, independientemente de la causa que lo haya motivado.

Línea de acción "Gestión Cumplimiento" (Stream B): Medir el cumplimiento de los requisitos de cumplimiento

El objetivo de la actividad es que la organización alcance una visión clara y precisa sobre el grado de conformidad con los requisitos de cumplimiento normativo.

Se desarrollará un programa para medir de forma objetiva e informar sobre el grado de conformidad de las aplicaciones con los requisitos de cumplimiento normativo. En la medida de lo posible, se procurará la automatización de las pruebas para detectar lo antes posible los incumplimientos de las aplicaciones o las regresiones en nuevas versiones. Cuando no sea posible llevar a cabo pruebas totalmente automatizadas, los equipos de control de calidad, auditoría interna o seguridad de la información evaluarán el cumplimiento periódicamente mediante pruebas manuales, entrevistas, y cualquier otro mecanismo apropiado.

En el caso de que se detecten incumplimientos, se dejará constancia de los problemas detectados y se realizará seguimiento de las acciones correctivas pertinentes, con el fin de verificar que estas son eficaces y que los equipos de producto trabajan en su resolución y progresan adecuadamente. Para mejorar este proceso de revisión, se elaborarán y comunicarán una serie de informes estándar y cuadros de mando de cumplimiento. Esto facilitará que los equipos de trabajo comprendan el estado actual de cumplimiento, y ayudará a la organización a gestionar eficazmente las desviaciones y a priorizar las acciones correctivas.

En caso de desviaciones significativas sobre los requisitos de cumplimento que requieran de esfuerzos económicos importantes o desarrollos significativos, se consultará con el personal expertos para evaluar el coste se solución frente a otras alternativas como minimizar alcance de cumplimiento, eliminar algún requisito o reducir la funcionalidad de la aplicación. Aquellos incumplimientos que se vayan a mantener a largo plazo deberán ser aprobados por la dirección con la correspondiente aceptación formal del riesgo de cumplimiento.

4.2.1.3 Práctica Formación y Orientación (EG)

La práctica de Formación y Orientación se centra en dotar al personal implicado en el ciclo de vida del software de los conocimientos y recursos necesarios para diseñar, desarrollar e implantar software seguro.

Con las mejoras de conocimientos y en el acceso a la información, los equipos de proyecto aumentarán su eficacia a la hora de identificar y establecer mitigaciones de forma preventiva para hacer frente a los riesgos de seguridad específicos a los que se expone la organización.

Los principales objetivos son la formación de los empleados y el aumento de su concienciación en materia de seguridad, ya sea mediante sesiones dirigidas por un instructor o de módulos informatizados. A medida que la organización madura, se establece una base de conocimientos sólida, primeramente en los desarrolladores y posteriormente se extiende a otros roles. Finalmente, se evoluciona a un modelo formativo basado en roles que optimiza el proceso de formación y garantiza la aplicabilidad y eficacia del mismo.

Además de la formación del personal, esta práctica también promueve la realización de inversiones en la promover una cultura de la seguridad en la organización a través de la colaboración entre equipos. Las herramientas de colaboración y el aumento de la transparencia del conocimiento sobre tecnologías y herramientas son elementos fundamentales para apoyar la mejorar la seguridad de las aplicaciones.

En la siguiente figura, se muestran las actividades asociadas a los distintos niveles de madurez comprendidos en la práctica de Formación y Orientación.

Maturity level		Stream A Training and Awareness	Stream B Organization and Culture
1	Offer staff access to resources around the topics of secure development and deployment.	Provide security awareness training for all personnel involved in software development	Identify a "Security Champion" within each development team.
2	Educate all personnel in the software lifecycle with technology and role-specific guidance on secure development.	Offer technology and role-specific guidance, including security nuances of each language and platform	Develop a secure software center of excellence promoting thought leadership among developers and architects.
3	Develop in-house training programs facilitated by developers across different teams.	Standardized in-house guidance around the organization's secure software development standards.	Build a secure software community including all organization people involved in software security.

Figura 9. Actividades de la práctica de Formación y Orientación. Fuente OWASP SAMM

Como se puede observar en la Figura 9, se definen las dos líneas acción para esta práctica: Training and Awareness (Stream A) y Organization and Culture (Stream B).

Las actividades de la línea asociada a la Formación y Concienciación tienen como finalidad la de aumentar los conocimientos generales en materia de seguridad del software entre las diferentes partes interesadas de la organización.

Por su parte, las actividades asociadas a la línea de actuación Organización y Cultura se centran en la promoción dentro de la organización de una cultura de seguridad de las aplicaciones como factor de éxito del proyecto de gestión del ciclo de desarrollo seguro de software.

A continuación, se describirá en detalle cada una de las seis actividades (dos por cada nivel de madurez) comprendidas en la práctica de Formación y Orientación.

4.2.1.3.1 Formación y Orientación EG1

El objetivo de la práctica en este nivel de madurez inicial es ofrecer al personal de la organización acceso a los recursos de formación relacionados con el desarrollo y despliegue seguros

Línea de acción de "Formación y Concienciación" (Stream A): Formar y concienciar a involucrados

Se promoverán actividades formativas para concienciar sobre aspectos y principios básicos de seguridad a los diversos involucrados (desarrolladores, gestores, *testers*, etc.) en el ciclo de vida del software.

La formación cubrirá los temas más importantes relacionados con la seguridad y privacidad de las aplicaciones y se orientarán para que sean comprensibles tanto por personal técnico como no técnico. Por ejemplo, el contenido de la formación podría ser el siguiente:

- Principios de seguridad: diseño seguro, privilegios mínimos, defensa en profundidad por capas, seguridad en caso de fallo, gestión de sesiones, etc.
- Estándares, políticas y procedimientos de la organización definidos para la seguridad de las aplicaciones
- Vulnerabilidades OWASP Top 10 [18] a alto nivel.

La formación incluirá tanto al personal interno de la organización como a personal externo subcontratado y de proveedores habituales. También se impartirá como parte del proceso de incorporación de nuevo personal a los equipos de desarrollo. Los asistentes a la formación deberán firmar su asistencia o se expedirá un certificado de aprovechamiento con el fin de realizar seguimiento de la acción y demostrar el cumplimiento de esta actividad.

El contenido de esta formación básica deberá ser actual. Por ello, se revisará y, si procede se actualizará como mínimo cada 12 meses.

Línea de actuación "Organización y Cultura" (Stream B): Identificar a los "security champions"

Se promoverá la figura de un experto en seguridad por cada equipo de desarrollo: el "Security Champion". Este se encargará de establecer un enlace entre Seguridad Informática y el Dpto. de Desarrollo.

Los "security champions" dedicarán número de horas semanales a tareas relacionadas con el desarrollo seguro de aplicaciones: revisión de defectos y vulnerabilidades, ayuda en priorización de las tareas de mitigación, revisión de riesgos, arquitectura de seguridad, etc. Para llevar a cabo esta labor, los "security champions" recibirán la formación específica necesaria.

Además de las actividades mencionadas, los "security champions" proporcionarán información de forma periódica al equipo de desarrollo sobre los problemas de seguridad en los proyectos, de forma que todo el equipo esté al tanto de los problemas existentes y de las acciones de corrección o mitigación que se deberán de acometer.

Las labores desempeñadas por este personal con conocimientos de seguridad avanzados, retornará un importante beneficio para la organización en forma de sistemas más seguros y resilientes.

4.2.1.3.2 Formación y Orientación EG2

El objetivo de la práctica en este nivel es formar a todo el personal involucrado en el ciclo de desarrollo del software. La formación se orientará de forma específica según las funciones o roles desempeñados.

Línea de actuación "Formación y Concienciación" (Stream A). Personalizar la formación.

Se personalizará la formación para que se adapte las necesidades concretas de cada uno de los roles que intervienen en el ciclo de vida de desarrollo. Además de las materias descritas en la actividad EG1 (nivel de madurez anterior), se incluirá el contenido específico necesario para desempeñar las actividades propias de cada rol. Por ejemplo:

Formación específica por rol:

- Roles de gestión: modelo de SAMM, análisis de riesgos, modelos de amenazas y gestión de defectos.
- Desarrolladores: OWASP Top 10, codificación segura, arquitecturas de seguridad, herramientas y técnicas de testing, etc.
- *Testers*: Herramientas para pruebas de seguridad y gestión de defectos, técnicas y tecnologías específicas de pruebas, etc.

- Security Champions: formación específica en diferentes fases y herramientas del ciclo de vida del software seguro, además de la formación que reciben los desarrolladores.

La formación será obligatoria para personal interno de la organización como a personal externo subcontratado y los asistentes deberán firmar su asistencia o se expedirá un certificado de aprovechamiento con el fin de demostrar el cumplimiento de esta actividad. Siempre que sea posible, al terminar la formación se realizarán pruebas de evaluación para garantizar la comprensión y aprovechamiento por parte de los asistentes.

Adicionalmente, se realizarán encuestas sobre la formación para evaluar la calidad y relevancia de la misma, y se recopilarán las sugerencias de los equipos sobre la inclusión de otros contenidos relevantes para el desempeño del trabajo.

Se actualizará e impartirá la formación anualmente con el fin de incluir o actualizar el contenido de la misma con los cambios organizacionales, tecnológicos y últimas tendencias del sector.

Línea de actuación "Organización y Cultura" (Stream B). Desarrollar centros de excelencia

Se creará dentro de la organización un centro de excelencia en materia de codificación segura, con arquitectos y desarrolladores senior de las diferentes unidades de negocio y *stacks* tecnológicos empleados en la organización. Dicho centro de excelencia deberá ser constituido oficialmente con su correspondiente organigrama.

Este grupo apoyará al resto de equipos de desarrollo de la organización, definiendo estándares y guías de buenas prácticas para mejorar los procesos de desarrollo de la organización. El objetivo final es mitigar los riesgos provocados por la cada vez más rápida velocidad de los cambios tecnológicos tales como la adopción de nuevos lenguajes de programación, nuevos *frameworks* de desarrollo, librerías, etc.

Los equipos de desarrollo que deseen realizar cambios arquitectónicos significativos en su software consultarán con el grupo de trabajo para evitar un impacto adverso en el la seguridad de las aplicaciones o en los controles de seguridad establecidos.

El grupo ayuda a identificar, formar y apoyar a "Product Champions", responsables de ayudar a los diferentes equipos a implementar herramientas que automaticen, agilicen o mejoren varios aspectos del ciclo de vida del desarrollo seguro.

El centro de excelencia tendrá la misión de identifica a los equipos de desarrollo con mayores niveles de madurez y éxito, e identifican las prácticas y herramientas que permiten estos logros, con el objetivo de reproducirlos en otros equipos.

El grupo aporta además su experiencia y conocimiento ayudando a los equipos de seguridad de la información a evaluar herramientas y soluciones para mejorar la seguridad de las aplicaciones, asegurándose de que estas herramientas no

solo son útiles, sino que también son compatibles con la forma de trabajar de los equipos de desarrollo de la organización.

4.2.1.3.3 Formación y Orientación EG3

El objetivo de la práctica en este nivel es el desarrollo de programas de formación internos elaborados por los desarrolladores de los diferentes equipos técnicos de la organización.

Línea de actuación "Formación y Concienciación" (Stream A). Normalizar y formalizar la formación de seguridad.

Se desarrollará un programa interno de formación en materia de seguridad en el desarrollo de software. Para ello, se diseñarán itinerarios formativos con las competencias y conocimientos mínimos que deberían tener los diferentes miembros (formación específica por roles) que se incorporen a los equipos de desarrollo, adecuando dicha formación a las necesidades de la organización y las mejores prácticas de la industria.

Se podría incluso valorar que las personas que se incorporaran a determinados proyectos, áreas o desarrollos críticos no pudieran hacerlo si no han completado satisfactoriamente un itinerario formativo determinado.

El programa se gestiona de forma interna en la organización (aunque se pueden adquirir algunos módulos de forma externa), se revisará la participación del personal involucrado en los proyectos de desarrollo y se realizarán pruebas para evaluar el aprovechamiento y competencias adquiridas por parte de los asistentes.

Idealmente, se utilizará un sistema de gestión del aprendizaje (LMS) para hacer un seguimiento de las formaciones y certificaciones.

Línea de actuación Formación y Concienciación (Stream B). Establecer una comunidad de seguridad.

La seguridad es responsabilidad de toda la organización y no únicamente del Departamento de Seguridad. Para promover esta conciencia en toda la organización se implementarán plataformas comunicación e intercambio de conocimientos que ayuden a fomentar comunidades en torno a diferentes tecnologías, herramientas y lenguajes de programación. En estas comunidades, los empleados comparten información, discuten los retos con otros desarrolladores y buscan en la base de conocimientos respuestas a problemas previamente discutidos.

Se promoverá la formación de comunidades específicas por funciones o roles que permitan que los diferentes equipos se comuniquen libremente y la información se disemine en las diversas unidades de negocio de la organización.

Estas comunidades servirán también para localizar/seleccionar talento interno de la organización con el fin de reclutar miembros para el centro de excelencia de seguridad o "*security champions*" para los equipos de trabajo.

Se analizarán mecanismos dentro de la organización para favorecer o recompensar su trabajo a los participantes de las comunidades.

Para favorecer la comunicación y el trabajo en equipo de las comunidades, se habilitará un portal donde se recopile toda la información generada por las

comunidades. Dicho portal se utilizará además para obtener información relevante, como la notificación de incidentes de seguridad, actualizaciones de herramientas cambios en las arquitecturas, etc.

4.2.2 Función de negocio Diseño

Esta función del modelo se centra en los procesos y actividades relacionados con la forma en que una organización crea el software dentro de los proyectos de desarrollo. En general, esto incluye las etapas la recopilación de requisitos, la especificación de la arquitectura de alto nivel y el diseño detallado.

Las tres prácticas comprendidas en el ámbito de la función de Diseño son:

- **Evaluación de amenazas**. Esta práctica se centra en la identificación de posibles amenazas en las aplicaciones.
- **Requisitos de seguridad**. Esta práctica se centra en la definición de los requisitos de seguridad.
- **Arquitectura de seguridad**. La práctica de la arquitectura de seguridad se centra en la gestión de los riesgos arquitectónicos de las aplicaciones.

A continuación se describirá cada una de las tres prácticas mencionadas.

4.2.2.1 Práctica Evaluación de Amenazas (TA)

La práctica de Evaluación de Amenazas (TA) se centra en la identificación y entendimiento de los riesgos a nivel de proyecto, basándose en la funcionalidad del software que se está desarrollando y las características del entorno de ejecución. A partir de la información sobre las amenazas y los probables ataques contra cada aplicación, la organización será más eficaz en la toma decisiones sobre la priorización de las iniciativas en materia de seguridad. Además, el hecho de que las decisiones se tomen conforme a criterios de aceptación de riesgos hace que estén mejor alineadas con el negocio.

Se comenzará con el desarrollo de modelos simples de amenazas y perfiles básicos de riesgo de las aplicaciones, para ir mejorando a medida que la organización vaya madurando. Esto proporcionará una mayor comprensión de los posibles impactos derivados de problemas de seguridad, a la vez que se mantiene una estrecha vigilancia sobre las capacidades de la organización frente a las amenazas conocidas.

En la siguiente figura, se muestran las actividades asociadas a los distintos niveles de madurez comprendidos en la práctica.

Maturity level		Stream A Application Risk Profile	Stream B Threat Modeling
1	Best-effort identification of high-level threats to the organization and individual projects.	A basic assessment of the application risk is performed to understand likelihood and impact of an attack.	Perform best-effort, risk-based threat modeling using brainstorming and existing diagrams with simple threat checklists.
2	Standardization and enterprise-wide analysis of software-related threats within the organization.	Understand the risk for all applications in the organization by centralizing the risk profile inventory for stakeholders.	Standardize threat modeling training, processes, and tools to scale across the organization.
3	Proactive improvement of threat coverage throughout the organization.	Periodically review application risk profiles at regular intervals to ensure accuracy and reflect current state.	Continuously optimization and automation of your threat modeling methodology.

Figura 10. Actividades de la práctica Evaluación de Amenazas. Fuente: OWASP SAMM

Como se puede observar en la Figura 10, se definen las dos líneas acción para esta práctica: Perfil de Riesgo de las Aplicaciones (Stream A) y Modelado de Amenazas (Stream B).

Las actividades de la línea correspondiente al Perfil de Riesgo de las Aplicaciones se centran en la utilización de los perfiles de riesgo para ayudar a identificar qué aplicaciones suponen una grave amenaza para la organización en caso de que fueran atacadas o comprometidas.

Por su parte, las actividades de la línea de Modelado de Amenazas tienen como finalidad ayudar a los equipos de desarrollo a entender los riesgos que existen en el software que se está construyendo y cómo se pueden mitigar o eliminar dichos riesgos.

A continuación, se describirá en detalle cada una de las seis actividades (dos por cada nivel de madurez) comprendidas en la práctica.

4.2.2.1.1 Evaluación de Amenazas (TA1)

El objetivo de la práctica en este nivel de madurez inicial es tener en cuenta de forma explícita la seguridad durante el proceso de creación requisitos del software.

Línea de acción "Perfil de Riesgo de las Aplicaciones" (Stream A): Evaluación de riesgo de las aplicaciones

Se comenzará utilizando un método sencillo para valorarlos riesgos de las aplicaciones. Los riesgos se evaluarán, estimando el impacto potencial que supone para el negocio de la organización un ataque:

- Violación de la confidencialidad
- Violación de la integridad
- Disponibilidad de la aplicación.

Se valorará utilizar un conjunto reducido de preguntas (5 a 10) para comprender las características más importantes de la aplicación: p.ej. si la aplicación procesa datos financieros, si está expuesta a Internet o si se trata datos personales, etc.

Se clasificará el nivel de riesgo de las aplicaciones siguiendo un método cualitativo simple del tipo: alto/medio/bajo.

Es importante utilizar estos valores para representar y comparar el riesgo de las diferentes aplicaciones entre sí. Las organizaciones maduras, muy orientadas al riesgo, pueden hacer uso de esquemas de riesgo más cuantitativos.

No es preciso reinventar la rueda: si la organización ya utiliza un método o esquema para valorar riesgos que funciona, se empleará dicho método.

Línea de acción "Modelado de Amenazas" (Stream B): Realizar modelado básico de amenazas

El modelado de amenazas consiste en identificar, evaluar y gestionar las amenazas del sistema, los defectos de diseño en la arquitectura del mismo y las mitigaciones de seguridad recomendadas.

El modelado de amenazas es un ejercicio de equipo, que debe incluir a propietarios de productos, arquitectos, “security champions” y *testers*. En este nivel de madurez inicial, es importante implicar a equipos de trabajo y resto de partes interesadas en esta actividad con el fin de incrementar la concienciación en materia de seguridad y crear una visión compartida sobre la seguridad del sistema.

En esta práctica de nivel de madurez inicial, se analizarán de forma específica las amenazas de aquellas aplicaciones de mayor riesgo. Se utilizarán listas de comprobación de amenazas sencillas y relevantes (p.e. STRIDE de Microsoft o OWASP Threat Dragon). Se evitará el uso de modelos complejos que pueden dar lugar a reuniones o discusiones interminables y, por tanto, a afectar al rendimiento de la organización o a que se vea como un proceso engorroso, complejo y de difícil cumplimiento.

En el caso de metodologías de desarrollo iterativas, este proceso se deberá alinear con la metodología empleada y se llevará a cabo de forma iterativa.

En caso de incorporación de nuevas funcionalidades o cambios en los sistemas, el modelado se realizará sobre las nuevas características, no siendo necesario repetir todo el proceso desde cero. Por ello es importante conservar la documentación, diagramas, hilos de correo, etc., con el fin de reutilizarlos en posteriores cambios.

4.2.2.1.2 Evaluación de Amenazas (TA2)

El objetivo de la práctica en el segundo nivel de madurez es obtener una mayor granularidad de requisitos de seguridad de las aplicaciones a partir de la lógica de las aplicaciones y los riesgos de las mismas.

Línea de acción “Perfil de Riesgo de las Aplicaciones” (Stream A): Inventariar los perfiles de riesgo de las aplicaciones.

Se elaborará un inventario de todas las aplicaciones que componen el portfolio de la organización (incluidas aplicaciones “legacy”), incluyendo la información asociada al nivel de riesgo de cada aplicación. De esta forma, la organización podrá enfocar los esfuerzos en la resolución de problemas o vulnerabilidades en aquellos sistemas que más lo precisen, atendiendo a las prioridades establecidas.

A este nivel no es suficiente disponer la realización de un análisis cualitativo informal basado en categorizar las aplicaciones como riesgo alto, medio y bajo. Ello dificulta la comparación y la clasificación de las aplicaciones según el nivel de riesgo. Por este motivo, se utilizará o desarrollara un método estandarizado para la evaluación del riesgo, evaluando, entre otras cosas, el impacto en las dimensiones de seguridad (confidencialidad, integridad y disponibilidad de los datos). Adicionalmente se evaluará el riesgo de la privacidad, teniendo en cuenta los datos que gestiona la aplicación y las consecuencias de una potencial violación de los mismos. Un método que cumpliría con estos requisitos es, por ejemplo, OWASP Risk Rating Methodology [19].

Una vez obtenido el nivel de riesgo de todas las aplicaciones, la organización podrá clasificarlas, asignar la responsabilidad y priorizar las acciones que correspondan para gestionar el riesgo.

Los perfiles de riesgo deberían ser almacenados en un repositorio o inventario centralizado que permita la consulta por parte de todos los involucrados, realizar revisiones posteriores, etc.

Línea de acción "Modelado de Amenazas" (Stream B): Estandarizar y ampliar la modelización de las amenazas.

El principal objetivo de esta actividad es la estandarización del modelado de amenazas de las aplicaciones y promover su utilización en la organización.

Para ello se deberá emplear una metodología estándar para el modelado y se definirá una estrategia para promocionar el uso de la misma en toda la organización, con el fin de que se convierta en práctica habitual.

El modelado de amenazas es una actividad que requiere cualificación adecuada. Por ello, se deberá formar específicamente en esta materia a arquitectos, responsables de seguridad, "Security Champions" y, en general a todos los involucrados en esta actividad. También ayuda la elaboración de plantillas, escenarios y modelos de ejemplo que facilite a los equipos la realización de esta actividad.

La metodología elegida incluirá, como mínimo, la elaboración de diagramas, la identificación de amenazas, la mitigación de defectos de diseño y la forma en que se validarán los artefactos resultantes.

La ventaja de elaborar diagramas del modelo, es que permite una comprensión detallada del contexto y del funcionamiento principal de la aplicación.

En lo que respecta al descubrimiento de amenazas, se utilizarán *check* lists como las proporcionadas por la metodología STRIDE u otras similares adaptadas o específicas del contexto de la organización.

Para cada amenaza o defecto detectado durante el análisis, se clasificará según el riesgo que supone para la organización y se propondrán los controles de mitigación correspondientes para hacer frente a las amenazas. El resultado de la modelización alimentará el proceso de Gestión de Defectos para realizar una gestión adecuada de los mismos por parte de los equipos de desarrollo.

Por último, se definirán "disparadores" que provoquen la revisión y actualización de los modelos existentes. Por ejemplo, cambios de tecnología, despliegue de las aplicaciones en un contexto diferente, etc. Así, cuando suceda algún evento "disparador", el modelo será revisado convenientemente para detectar potenciales nuevos defectos.

4.2.2.1.3 Evaluación de Amenazas (TA3)

El objetivo de esta práctica en su mayor nivel de madurez es requerir la requerir que se lleve a cabo un proceso de identificación de amenazas de seguridad para todos los proyectos de software.

Línea de acción "Perfil de Riesgo de las Aplicaciones" (Stream A): Revisión periódica de los perfiles de riesgo.

La organización revisará de forma periódica el inventario de riesgos de las aplicaciones.

Tanto la cartera de aplicaciones de cualquier organización como las condiciones de entorno de las mismas cambia a lo largo del tiempo. Por ejemplo, habrá

aplicaciones que dejan de utilizarse, aplicaciones que pasan de un uso marginal a uno más amplio, aplicaciones de uso interno que pasan a ser expuestas a Internet, etc.

Por todo lo anterior, es necesario realizar una revisión periódica del inventario de riesgos para garantizar la coherencia, rigor y vigencia de las evaluaciones de riesgo de la cartera de aplicaciones. Adicionalmente, se definirán “disparadores” que provoquen la revisión y actualización del perfil de riesgo ante cambios en las condiciones de contexto de las aplicaciones que pudieran afectar a su nivel de riesgo, y a actuar en consecuencia.

Como parte del proceso de maduración de la organización, se deberá estimular a los equipos a analizar y cuestionar continuamente qué cambios en las condiciones de contexto de las aplicaciones son susceptibles de afectar al perfil de riesgo de las mismas.

Por último, con el fin de aumentar el nivel de madurez en la ejecución de esta actividad, se debería formar de forma continua a los equipos sobre las mejores prácticas y lecciones aprendidas en las evaluaciones de riesgo. De esta forma, se mejorará el rendimiento en la ejecución de la actividad, tanto en la velocidad del proceso de evaluación como en la calidad del resultado.

Línea de acción “Modelado de Amenazas” (Stream B): Optimizar el proceso de modelado de amenazas

Una vez que el modelado de amenazas está integrado como práctica habitual en el ciclo de vida de desarrollo y forma parte de la cultura de seguridad de los equipos, el siguiente paso es optimizar el proceso. Para ello, se elaborarán plantillas y patrones de riesgos, librerías de amenazas, defectos de diseño y mitigaciones con el fin de reutilizarlas y agilizar el proceso de modelado.

De forma periódica (por ejemplo anualmente) se revisaran los modelos de amenazas empleados para garantizar la vigencia de las mismas y verificar que no hay nuevas amenazas relevantes.

Se realizarán reuniones de retrospectiva para reflexionar sobre el trabajo realizado, recopilar las lecciones aprendidas y cómo aplicar dichas enseñanzas en la optimización del proceso.

Cada cierto tiempo se evaluará por personal independiente la calidad de los modelos de amenazas con el fin de garantizar la calidad de los mismos.

Se perseguirá la automatización del proceso mediante el uso de herramientas de modelado y se integrarán con otras herramientas del SLDC tales como las de verificación de la seguridad, seguimiento de riesgos, gestión de vulnerabilidades o defectos.

Adicionalmente, se podrán considerar prácticas de "modelado de amenazas como código" [20] para integrar la actividad de modelado de amenazas con el código de las aplicaciones.

4.2.2.2 Práctica Requisitos de Seguridad (SR)

Esta práctica se centra en los requisitos de seguridad que son importantes en el contexto del desarrollo de software seguro.

Se consideran dos tipos de requisitos de seguridad. El primer tipo se ocupa de los requisitos típicos relacionados con el software y la especificación de los objetivos de seguridad para proteger la aplicación/servicio y sus datos. El segundo tipo trata de los requisitos relativos a proveedores involucrados en desarrollo de las aplicaciones, particularmente para los desarrollos subcontratados, que pueden tener un impacto significativo en la seguridad de una aplicación.

La seguridad de las librerías y componentes de terceros no está incluida en lo tratado en esta práctica, pues se trata en la práctica de Construcción Segura (cadena de suministro de software)

En la siguiente figura, se muestran las actividades asociadas a los distintos niveles de madurez comprendidos en la práctica.

Maturity level		Stream A Software Requirements	Stream B Supplier Security
1	Consider security explicitly during the software requirements process.	High-level application security objectives are mapped to functional requirements.	Evaluate the supplier based on organizational security requirements.
2	Increase granularity of security requirements derived from business logic and known risks.	Structured security requirements are available and utilized by developer teams.	Build security into supplier agreements in order to ensure compliance with organizational requirements.
3	Mandate security requirements process for all software projects and third-party dependencies.	Build a requirements framework for product teams to utilize.	Ensure proper security coverage for external suppliers by providing clear objectives.

Figura 11. Actividades de la práctica Requisitos de Seguridad. Fuente: OWASP SAMM

Como se puede observar en la figura anterior, se definen las dos líneas acción para esta práctica: Requisitos de Software (Stream A) y Seguridad de los proveedores (Stream B).

Las actividades de la línea correspondiente a “Requisitos de Software” trata sobre cómo los requisitos deben especificar los objetivos y las expectativas de seguridad para proteger las aplicaciones y sus datos.

Por otro lado, la línea asociada a la seguridad de los proveedores se ocupa de los requisitos de seguridad relativos a los proveedores, en particular para el desarrollo externalizado.

A continuación, se describirá en detalle cada una de las seis actividades (dos por cada nivel de madurez) comprendidas en la práctica.

4.2.2.2.1 Requisitos de Seguridad (SR1)

El objetivo de esta práctica en el nivel de madurez inicial es conseguir que se consideren de forma específica los requisitos de seguridad durante la fase de captura de requisitos del sistema.

Línea de acción “Requisitos de Seguridad” (Stream A): Identificación de los requisitos de seguridad

Se identificará de forma específica los requisitos de seguridad de la aplicación a partir de los requisitos funcionales del proyecto. Se analizarán las dimensiones de confidencialidad, integridad o disponibilidad deseadas para la aplicación o los datos tratados por la misma y se establecerán los objetivos de seguridad

(requisitos). Un ejemplo de requisito de seguridad sería: "*los datos personales para el proceso de registro deben transferirse y almacenarse de forma segura*".

Además de lo anterior, se analizará la funcionalidad de la aplicación desde la perspectiva de un atacante, con el fin de comprender cómo podría utilizar la aplicación un usuario malintencionado para intentar realizar un ataque. Este análisis facilitará la identificación de requisitos de protección adicionales que darán lugar a nuevos requisitos de seguridad. Un ejemplo de esto sería, en el caso de una "aplicación web", el siguiente requisito: "*los datos del formulario de registro deberán validarse para proteger a la aplicación de ataques de tipo SQL Injection*".

Los requisitos de seguridad no dejan de ser requisitos, aunque tradicionalmente han formado parte de los denominados requisitos no funcionales. Independientemente de cómo sean clasificados los requisitos de seguridad, se deberán seguir las buenas prácticas habituales para redactar requisitos. Es decir, deberán ser específicos, medibles, alcanzables, razonables y limitados en el tiempo (SMART).

Adicionalmente, para la identificación de los requisitos de seguridad, será preciso tener en cuenta lo establecido por las políticas y estándares fijados para dar cumplimiento a la normativa de aplicación y obligaciones de la organización. Recordemos que estas políticas y estándares establecen una línea base de requerimientos de seguridad para todas las aplicaciones y, por tanto, constituye otra fuente de requisitos de seguridad que deberán ser atendidos.

Línea de acción "Seguridad de Proveedores" (Stream B): Evaluación de proveedores

El objeto de esta actividad es comunicar la postura de seguridad de la organización con los proveedores y tratar con estos las necesidades y requerimientos de la organización de forma transaparente.

Los proveedores que colaboran en el desarrollo de software de la organización pueden suponer una fuente de riesgo y tener un impacto significativo en la seguridad de las aplicaciones. Es imprescindible conocer sus competencias y capacidades en lo que al desarrollo seguro se refiere.

Por lo comentando anteriormente, se evaluará a los proveedores sobre sus competencias y prácticas en materia de seguridad. De esta forma, se conocerán sus fortalezas y debilidades y se dispondrá de elementos de valoración para evaluar la necesidad de disponer medidas o controles adicionales para mitigar los riesgos potenciales.

Para llevar a cabo la valoración, se podrá utilizar una lista de comprobaciones básicas, realizar entrevistas con diferentes actores de la organización para conocer sus prácticas de seguridad o, idealmente, realizar una pequeña evaluación de madurez del proveedor utilizando las herramientas de SAMM.

En caso de que se detecte que un proveedor tiene competencias débiles, se debería discutir con dicho proveedor sobre la necesidad de establecer un plan de mejora de la seguridad que satisfaga las necesidades de nuestra organización. Llegado el caso, se evaluará la conveniencia de que dicho proveedor colabore con nuestra entidad, incluso en proyectos con un perfil de bajo riesgo. Hay que tener en cuenta que el nivel de riesgo de una aplicación

puede cambiar en el tiempo, y lo que hoy es un servicio con baja exposición puede evolucionar a una aplicación crítica más adelante.

Con el fin de evitar situaciones o conflictos no deseables con los proveedores, es muy importante trasmitirles de forma previa nuestras necesidades y que alineen sus prácticas con las de nuestra organización. Lo ideal es establecer de forma explícita lo que se espera de cada proveedor, discutirlo claramente y acordarlo formalmente (contrato) con ellos antes de empezar la colaboración.

4.2.2.2.2 Requisitos de Seguridad (SR2)

El objetivo de este nivel de práctica es aumentar la granularidad y detalle de los requisitos de seguridad, derivándolos a partir de la lógica de negocio de las aplicaciones y de los riesgos conocidos.

Línea de Acción "Requisitos de Seguridad" (Stream A): Estandarizar e integrar en el proceso de desarrollo los requisitos de seguridad

El objeto de esta actividad es alinear los requisitos de seguridad con otros requisitos de diversa tipología.

En general, los requisitos de seguridad pueden provenir de diversas fuentes, como las políticas y estándares de la organización, la legislación, problemas conocidos de las aplicaciones, métricas de la aplicación, etc. En este nivel de madurez, la captura de requisitos se realizará de forma sistemática, asegurando que para la elaboración de requisitos de seguridad se consultan todas las fuentes establecidas. Por ejemplo, se analizan las políticas definidas, la legislación aplicable, los sistemas de gestión de vulnerabilidades o la información histórica de defectos de las aplicaciones.

Tal y como se comentó anteriormente, los requisitos de seguridad no dejan de ser requisitos y se deberán gestionar de la misma forma que se hace, por ejemplo, con los requisitos funcionales. Por ello, se utilizará una notación estructurada de los requisitos de seguridad que permita su integración con la forma de especificar y gestionar otros requisitos (como los funcionales) del proyecto. Esto incluiría la elaboración de documentos de análisis de requisitos, creación de historias de usuario (*agile*), etc. Es decir, proceder de manera similar a como se haría con los requisitos funcionales de forma tradicional.

Al igual que sucede con los requisitos funcionales de las aplicaciones, los requisitos de seguridad deberán ser tenidos en cuenta durante el desarrollo del producto. Por tanto, deberán ser analizados, documentados, asignados, priorizados, probados, y validados. Los requisitos de seguridad, en definitiva, especifican características del producto final y deben ser tratados y gestionados como tal por parte de los equipos de desarrollo. La única particularidad es que los requisitos de seguridad deberán ser revisados por los "security champions" del equipo, que también evaluará el resultado de los tests asociados y pruebas de validación.

Línea de acción "Seguridad de Proveedores" (Stream B): Comunicar a proveedores sus responsabilidades en materia de seguridad

El objeto de esta actividad es definir y establecer claramente las responsabilidades en materia de seguridad con los proveedores de software.

Una buena forma de mitigar los riesgos de seguridad derivados del trabajo con proveedores es comunicarles de forma previa las expectativas de la organización en lo que respecta a la seguridad del software, establecer claramente las responsabilidades del proveedor y formalizarlo en el contrato de colaboración. Esto evitará situaciones desagradables o conflictos durante la ejecución de los trabajos y aportará transparencia en la relación.

En la fase de contratación de proveedor se establecerá en el contrato las responsabilidades y requisitos concretos que han de satisfacerse. Éstos pueden ser requisitos de calidad y seguridad específicos o tareas concretas que el proveedor deberá realizar como parte del trabajo. Ejemplos de requisitos a establecer:

- El software entregado estará libre de vulnerabilidades del OWASP Top 10.
- Durante el desarrollo el proveedor realizará análisis estático del código. No se aceptarán productos finales cuyas métricas no cumplan los umbrales mínimos establecidos.
- La cobertura de test automáticos deberá ser superior a un X%.
- En caso de desviaciones sobre los umbrales mínimos establecidos, los defectos serán subsanados por el proveedor en un plazo X.
- Antes de la puesta en producción de un producto, se realizará un test de penetración por parte de un tercero independiente. En caso de que se detecten vulnerabilidades críticas o de nivel alto, el proveedor las resolverá en un plazo X. Las vulnerabilidades de nivel medio se resolverán en un plazo Y.

Idealmente, una vez que se haya puesto en práctica este modo de proceder con varios proveedores, se elaborará una plantilla estándar que será la base para la elaboración de todos los contratos con nuestros proveedores de servicios de desarrollo de software. De esta forma, se evitará pasar por alto los aspectos esenciales. Se podrán realizar modificaciones sobre este “acuerdo tipo” para adaptarlo a ciertos casos, en función del proveedor, circunstancias del contrato, etc.

La organización realizará seguimiento de los indicadores clave de rendimiento para asegurar el cumplimiento de los requisitos de calidad y seguridad acordados.

4.2.2.2.3 Requisitos de Seguridad (SR3)

El objetivo de esta práctica en su mayor nivel de madurez es requerir que se lleve a cabo el proceso de identificación de requisitos de seguridad para todos los proyectos de software de la organización y el software de terceros.

Línea de Acción “Requisitos de Seguridad” (Stream A): Desarrollar un marco de trabajo para los requisitos de seguridad

El objeto de esta actividad es el de desarrollar un marco de trabajo para la identificación de requisitos de seguridad de las aplicaciones, con el fin que se lleve a cabo un proceso de definición de requisitos de seguridad efectivo y eficiente.

Se establecerá un marco de trabajo común para facilitar que en todos los proyectos se capture un conjunto de requisitos de seguridad apropiado y completo y asegure la calidad de los mismos.

Disponer de un marco de trabajo, ayuda además a los equipos a mejorar la eficiencia del proceso, definiendo conjuntos de requisitos comunes a cierto tipo de aplicaciones, requisitos basados en normas, permitiendo la reutilización de requisitos, estableciendo una categorización común de los mismos, etc. Así, la organización analizará las diversas fuentes de requisitos y fomentará la elaboración de un inventario de requisitos comunes derivados de normas, estándares, políticas, funcionalidades de proyectos o familias de productos similares, etc. Esto no solo agilizará la definición de los requisitos de seguridad, sino que permitirá además reutilizar los casos de pruebas o incluso funcionalidades completas que ya han sido implementadas anteriormente.

El marco de trabajo ofrecerá también directrices claras sobre cómo especificar los requisitos y formalizará la manera de definirlos: qué es necesario describir y cómo, los requerimientos para considerar un requisito como totalmente definido o los criterios para considerar que un requisito se ha implementado.

Línea de acción "Seguridad de Proveedores" (Stream B): Alinear la metodología de trabajo con los proveedores

El objeto de esta actividad es el de alinear las prácticas de desarrollo de software seguro de la organización con sus proveedores para controlar y limitar los riesgos de seguridad de las aplicaciones.

Para minimizar el riesgo de problemas de seguridad en el software entregado, se deberá alinear al máximo el proceso de desarrollo de los proveedores con las prácticas establecidas por la organización. Los procesos deberían ser similares y se deberán utilizar las mismas herramientas, entornos de construcción, verificación y despliegue, así como compartir las herramientas de requisitos, repositorio de código, etc.

Adicionalmente, se establecerán puntos o hitos de control regulares para asegurar el alineamiento de los procesos y el progreso cualitativo de los trabajos realizados por el proveedor. De esta forma se evitarán desviaciones significativas que, al final, suponen la realización de esfuerzos adicionales para su compensación.

En el caso de que los proveedores no puedan satisfacer estos requerimientos de la organización, se planificarán acciones adicionales antes de la implantación tales como realizar un modelado de amenazas, análisis estático y de dependencias de la solución, etc. Es decir, se establecerán controles compensatorios para garantizar que los productos entregados satisfacen las necesidades de la organización en materia de seguridad.

4.2.2.3 Práctica Arquitectura de Seguridad (SA)

La práctica de "Arquitectura de Seguridad" (SA) se centra en la seguridad de los componentes y tecnologías empleadas para el diseño de la arquitectura del software.

En el diseño de la arquitectura de seguridad se evaluará el conjunto de los componentes que forman la arquitectura base de su solución, enfocándose en las características de seguridad de dichos elementos.

Por otro lado, se evaluarán, desde la perspectiva de la seguridad, las tecnologías y marcos de trabajo (*frameworks)* empleados para el propio proceso de desarrollo del software.

En la siguiente figura, se muestran las actividades asociadas a los distintos niveles de madurez comprendidos en la práctica.

Maturity level		Stream A Architecture Design	Stream B Technology Management
1	Insert consideration of proactive security guidance into the software design process.	Teams are trained on the use of basic security principles during design	Elicit technologies, frameworks and integrations within the overall solution to identify risk.
2	Direct the software design process toward known secure services and secure-by-default designs.	Establish common design patterns and security solutions for adoption.	Standardize technologies and frameworks to be used throughout the different applications
3	Formally control the software design process and validate utilization of secure components.	Reference architectures are utilized and continuously evaluated for adoption and appropriateness.	Impose the use of standard technologies on all software development.

Figura 12. Actividades de la práctica Arquitectura de Seguridad. Fuente OWASP SAMM

Como se puede observar en la figura anterior, se definen las dos líneas acción para esta práctica: Diseño de la Arquitectura (Stream A) y Gestión de la Tecnología (Stream B).

La primera línea de acción se centra en la evaluación de los diversos componentes que formarán la arquitectura del producto final y en el empleo de buenas prácticas en el diseño.

Por su parte, la línea asociada a la gestión de la tecnología se enfoca en las tecnologías, soluciones y *frameworks* empleados en el proceso de desarrollo del software, con el fin de garantizar el nivel de seguridad apropiado en el producto final.

A continuación, se describirá en detalle cada una de las seis actividades (dos por cada nivel de madurez) comprendidas en la práctica.

4.2.2.3.1 Arquitectura de Seguridad (SA1)

El objetivo de la práctica en este nivel de madurez inicial es insertar en el proceso de desarrollo de software el hábito de tener en cuenta un la seguridad en la fase de diseño del software.

Línea de acción "Diseño de la Arquitectura" (Stream A): Respetar los principios básicos de seguridad

El objeto de la actividad es desarrollar y poner a disposición de los equipos de desarrollo un conjunto de principios básicos de seguridad que deberán ser tenidos en cuenta en el diseño de las aplicaciones.

El primer paso es elaborar una lista reducida con los principios de seguridad que deberán ser tomados en consideración. Por ejemplo:

- Ejecución con privilegios mínimos.
- Control de acceso cerrado por defecto.
- Segregación de funciones.
- Defensa en profundidad (por capas)
- Simplicidad en la funcionalidad de seguridad
- Transparencia (evitar seguridad por oscuridad)
- Protección del eslabón más débil.

En la fase de diseño de la arquitectura, el equipo de desarrollo realizará una comprobación de los principios de seguridad e identificará las características que deberán ser tenidas en cuenta para fortalecer la seguridad y satisfacer los requisitos de seguridad. Inicialmente, se priorizará un conjunto limitado de requisitos de seguridad (los más básicos) para que estos solo supongan un pequeño esfuerzo extra más allá del coste normal de implementación de los requisitos funcionales. El resto de requisitos quedarán como pendientes y se podrán planificar de forma incremental en futuras versiones.

Para mejorar el resultado del desempeño de esta actividad, se formará a los equipos en materia de seguridad antes del proceso, y se procurará la incorporación de más personal con conocimientos de seguridad para ayudar a tomar decisiones de diseño.

Línea de acción "Gestión de la Tecnología" (Stream B): Identificar herramientas y tecnologías

El objeto de esta actividad es tener una visión clara de las herramientas y tecnologías empleadas en el desarrollo y ser consciente de cómo estas pueden introducir riesgos de seguridad en el producto final.

En la actualidad, en el desarrollo de software es bastante frecuente hacer uso de nuevas tecnologías, frameworks, utilidades y herramientas con la finalidad de facilitar y/o acelerar el proceso de desarrollo o mejorar aspectos como la versatilidad o escalabilidad del producto final. No hay nada malo en utilizar nuevas tecnologías para mejorar las capacidades de procesos y productos, pero es preciso tener en cuenta que se pueden introducir nuevos riesgos para la organización que hay que gestionar adecuadamente.

Para evitar asumir nuevos riesgos de forma inconsciente, se identificarán las tecnologías, frameworks y herramientas más importantes que se utilizan para cada aplicación, incluyendo los entornos de desarrollo y operativo. Se realizará un análisis de las características y del estado de la seguridad de estos componentes y se pondrá sobre la mesa los hallazgos o defectos de seguridad relevantes para que puedan ser gestionados adecuadamente.

4.2.2.3.2 Arquitectura de Seguridad (SA2)

El objetivo de la práctica en el segundo nivel de madurez es orientar el diseño del software hacia un modelo basado en servicios seguros conocidos y diseñar el software bajo el principio de "seguro por defecto" (security-by-default).

Línea de acción "Diseño de la Arquitectura" (Stream A): Proveer catálogo de soluciones de seguridad de preferencia

El objeto de esta actividad es crear un catálogo de soluciones y servicios relacionados con aspectos de seguridad (identificación, control de acceso, etc.) que, de forma preferente, serán reutilizados por los equipos de desarrollo.

La organización definirá un catálogo de servicios relacionados con la seguridad y promoverá la utilización de los mismos por parte de las aplicaciones, en lugar de desarrollar soluciones ad-hoc para cada una de ellas.

Para elaborar el catálogo de servicios y soluciones, la organización identificará infraestructura o servicios compartidos por las diversas aplicaciones, tales como servicios de Single Sign On (SSO), control de accesos, registro de eventos (logging), monitorización, firewalls de nivel de aplicación, etc. Estos se recopilaran, evaluarán y clasificarán según la funcionalidad de seguridad que satisfagan. Para cada uno de ellos, se documentará el beneficio que supone para los equipos de desarrollo reutilizar dichos recursos. En el caso de que existan distintos recursos de una misma categoría, se podrá seleccionar más de un servicio compartido si así se requiere.

Los servicios del "catálogo" definirán en gran medida la seguridad futura de las aplicaciones de la organización. Por ello, se deberá realizar una revisión cuidadosa del estado de seguridad para conocer en detalle las fortalezas y posibles debilidades de cada uno de ellos.

De forma complementaria al catálogo, para cada uno de los servicios seleccionados se elaborará una guía de diseño para que los equipos de desarrollo dispongan de información suficiente para integrar las aplicaciones con dichos servicios. Esta información se pondrá a disposición de los equipos en forma de actividades de formación, talleres de trabajo, tutorización, directrices de diseño, documentos de mejores prácticas, etc. El objetivo es fomentar el uso de estas tecnologías clave en la organización prestando el apoyo necesario a los equipos de desarrollo para su adopción.

Si es necesario, se dará un tratamiento de proyecto a cada uno de los servicios del catálogo y se implementarán como soluciones horizontales fácilmente integrables en los futuros desarrollos.

En definitiva, se elaborará una especie de estándar de servicios relacionados con funciones de seguridad que serán utilizados por las aplicaciones de la organización para satisfacer los requisitos de seguridad más comunes.

Línea de acción "Gestión de la Tecnología" (Stream B): Promover el uso de herramientas y tecnologías de preferencia.

El objeto de esta actividad es proveer un catálogo de tecnologías y herramientas utilizadas para el desarrollo de productos, que dispongan del nivel adecuado de seguridad.

De igual forma que se propone en la actividad anterior, se identificarán tecnologías, *frameworks* y herramientas de uso común en los proyectos y se elaborará un inventario de tecnologías de referencia recomendadas para su uso en la organización.

Al igual que sucede con los servicios de seguridad, el "catálogo" de tecnologías recomendadas afectará en gran medida la seguridad y desempeño futuro de la organización. Por ello, se deberá realizar una revisión cuidadosa del estado de seguridad de cada una de estas tecnologías para conocer en detalle las

fortalezas y posibles debilidades de cada uno de ellas. Para ello se tendrá en consideración el historial de incidentes, el historial de respuesta a las vulnerabilidades, la idoneidad de la funcionalidad para la organización, la complejidad de uso o si existe el conocimiento suficiente dentro de la organización para su utilización.

El inventario de tecnologías será elaborado y aprobado por los arquitectos y desarrolladores senior de la organización. Adicionalmente, para garantizar la vigencia del catálogo este será revisado, al menos, una vez al año.

Con el fin de impulsar la utilización de las tecnologías de referencia, se debería elaborar, para cada una de ellas, una guía de utilización y ponerla a disposición de los equipos de desarrollo. Esta información se pondrá a disposición de los equipos en forma de actividades de formación, talleres de trabajo, tutoriales, directrices de diseño, documentos de mejores prácticas, etc.

4.2.2.3.3 Arquitectura de Seguridad (SA3)

El objetivo de la práctica en este nivel de madurez es controlar formalmente el proceso de diseño del software y validar la utilización de componentes seguros.

Línea de acción "Diseño de la Arquitectura" (Stream A): Construir arquitecturas de referencia

El objeto de esta actividad es proporcionar una total transparencia sobre las características, calidad y facilidad de uso de las soluciones y servicios de seguridad (de preferencia) proporcionados de forma centralizada.

La organización desarrollará un conjunto de arquitecturas de referencia basadas en grupos de componentes seguros para garantizar una implementación adecuada de la seguridad en las aplicaciones. Dichas arquitecturas constituyen el punto de partida para estandarizar un enfoque de la seguridad basado en los principios de "seguridad por defecto" (*security-by-default*) y orientación a la configuración (configuration-driven).

Disponer de arquitecturas de referencia tiene, además, la ventaja de reducir los esfuerzos relacionados con la revisión de la arquitectura de las aplicaciones y auditorías de seguridad, optimiza el proceso de desarrollo y reduce los gastos de mantenimiento de las aplicaciones.

Para facilitar que las arquitecturas de referencia se adapten a las necesidades de la organización y que se promocione su utilización, es importante que estas sean elaboradas por arquitectos, desarrolladores senior y otro personal técnico de la organización.

Las arquitecturas de referencia deberán ser mantenidas y mejoradas de forma continua basándose en los propios conocimientos de la organización y comunidades o grupos expertos externos. Por tanto, se supervisarán continuamente los puntos débiles conocidos o deficiencias del conjunto de soluciones de seguridad disponibles en la organización, aumentando de esta forma la eficacia de las arquitecturas de referencia utilizadas.

Las arquitecturas de referencia pueden materializarse en un conjunto base de bibliotecas y herramientas de software que serán utilizadas por parte de los equipos de proyecto en la construcción del software.

Esta actividad se podría iniciar mediante la selección de un proyecto concreto, trabajando en las primeras fases del ciclo de vida y haciendo que el personal experto construya la arquitectura de seguridad de forma genérica, de modo que pueda utilizarse en un futuro en otros proyectos de la organización.

Línea de acción "Gestión de la Tecnología" (Stream B): Imponer el uso de herramientas y tecnologías recomendadas

El objeto de esta actividad es reducir la superficie de ataque y la posibilidad de introducir riesgos en el producto final mediante el uso obligatorio de tecnologías y herramientas de desarrollo verificadas.

Se establecerán restricciones sobre el uso de tecnologías o herramientas de desarrollo que no formen parte del catálogo de tecnologías de referencia consideradas por la organización.

Para cada desarrollo de la organización, ya sea interno o contratado, se supervisará la utilización de tecnologías del estándar de la organización. Estas restricciones se implementaran, idealmente, en las propias herramientas de construcción de proyectos empleadas por la organización o mediante el análisis estático automatizado de los productos finales (código fuente, dependencias, etc.). A partir de los hallazgos en el uso de tecnologías que no forman parte del estándar, se deberá determinar si existen deficiencias en el inventario de tecnologías recomendadas que hayan motivado el uso de utilidades no recomendadas por la organización.

Adicionalmente, se analizarán los patrones de diseño y los módulos de las arquitecturas de referencia que no se utilicen o se utilicen de forma incorrecta para determinar si es necesario actualizarlos

4.2.3 Función de negocio Implementación

La función de negocio de Implementación se centra en los procesos y actividades relacionados con la forma en que una organización construye y despliega los componentes de software y gestiona sus defectos.

Las actividades de esta función son las que más repercuten en el día a día de los desarrolladores. El objetivo final es entregar un software que cumpla con lo requerido, sea fiable y tenga el menor número de defectos.

Las tres prácticas comprendidas en la función de Implementación son:

- **Construcción segura**. Esta práctica se enfoca en la creación de un proceso de construcción consistente y repetible, así como en tener en cuenta la seguridad de las dependencias del software de la aplicación.
- **Despliegue seguro**. Esta práctica se centra en aumentar la seguridad de los despliegues de software en el entorno de producción y en la gestión de los "secretos" (credenciales de acceso a repositorios, claves necesarias para el funcionamiento del producto, etc.) empleados.
- **Gestión de defectos**. Esta práctica se centra en la gestión de los defectos de seguridad en el software y sus métricas asociadas.

A continuación se describirá cada una de las tres prácticas mencionadas.

4.2.3.1 Práctica “Construcción Segura” (SB)

Esta práctica de la construcción segura (SB) hace hincapié en la importancia de construir el software de una manera estandarizada y repetible, utilizando componentes seguros, incluyendo las dependencias de software de terceros.

Por un lado, se centra en la eliminación de cualquier subjetividad del proceso de compilación mediante la búsqueda de la automatización total. Una cadena de construcción automatizada permite además incluir comprobaciones de seguridad automatizadas adicionales, como SAST y DAST, para detectar fallos de seguridad o vulnerabilidades en las aplicaciones antes de que estas lleguen a desplegarse.

Por otra parte, la práctica también trata los aspectos de seguridad de las dependencias del software desarrollado. El objetivo es identificar dichas dependencias y hacer seguimiento de su estado de seguridad para tener bajo control el impacto de sobre la seguridad de las aplicaciones. En un desarrollo avanzado de esta práctica, se aplicarán a las dependencias de software comprobaciones de seguridad similares a las de la propia aplicación.

En la siguiente figura, se muestran las actividades asociadas a los distintos niveles de madurez comprendidos en la práctica.

Maturity level		Stream A Build Process	Stream B Software Dependencies
1	Build process is repeatable and consistent.	Create a formal definition of the build process so that it becomes consistent and repeatable.	Create records with Bill of Materials of your applications and opportunistically analyze these.
2	Build process is optimized and fully integrated into the workflow.	Automate your build pipeline and secure the used tooling. Add security checks in the build pipeline.	Evaluate used dependencies and ensure timely reaction to situations posing risk to your applications.
3	Build process helps prevent known defects from entering the production environment.	Define mandatory security checks in the build process and ensure that building non-compliant artifacts fails.	Analyze used dependencies for security issues in a comparable way to your own code.

Figura 13. Actividades de la práctica Construcción Segura. Fuente: OWASP SAMM

Como se puede observar en la figura anterior, se definen las dos líneas acción para esta práctica: Proceso de Construcción (Stream A) y Dependencias del Software (Stream B).

La primera línea de acción se centra en el desarrollo de un proceso de construcción fiable, consistente y, en la medida de lo posible, automatizado, que garantice que el software que se despliega es predecible y está directamente vinculado al código fuente.

Por su parte, la línea asociada a la gestión de las dependencias del software, se enfoca en la identificación clara de las librerías externas y en garantizar que estas tienen un nivel de seguridad apropiado.

A continuación, se describirá en detalle cada una de las seis actividades (dos por cada nivel de madurez) comprendidas en la práctica.

4.2.3.1.1 Construcción Segura (SB1)

El objetivo de la práctica en este nivel es conseguir que el proceso de construcción de las aplicaciones sea repetible y consistente.

Línea de acción "Proceso de Construcción" (Stream A): Definir un proceso de construcción consistente

La finalidad de esta actividad es limitar el riesgo debido a errores humanos durante el proceso de construcción, lo que contribuye a reducir posibles riesgos de seguridad en el producto final.

Se definirá y describirá en detalle el proceso de construcción de las aplicaciones, estableciendo un procedimiento o conjunto de instrucciones claras que deberán seguir los equipos de desarrollo para generar los artefactos finales. Idealmente, el proceso deberá ser automatizado con el fin de evitar inconsistencias debidas a errores humanos y lograr que el resultado final sea idéntico (proceso repetible).

El proceso deberá incluir la práctica de generar un valor para cada artefacto generado que pueda utilizarse posteriormente para verificar su integridad. Por ejemplo una firma, CRC o un hash.

El procedimiento de construcción se publicará en un repositorio central de documentación y estará accesible para su utilización por parte de los equipos de desarrollo.

Se deberán revisar periódicamente las herramientas utilizadas durante el proceso de construcción para asegurar que están actualizadas (y parcheadas) debidamente y configuradas de forma segura según las instrucciones del fabricante y lo determinado por las buenas prácticas del sector.

Línea de acción "Dependencias del software" (Stream B): Identificar las dependencias de la aplicación

El objeto de la actividad es conocer la información sobre problemas de seguridad conocidos en las dependencias de las aplicaciones.

Para cada aplicación del catálogo, se identificarán y registrarán sus dependencias. Se tendrá en cuenta las distintas ubicaciones en las que pueden especificarse las dependencias, como por ejemplo archivos de configuración, directorio del proyecto, herramienta de gestión de paquetes o el propio código.

Se recopilará la siguiente información de cada dependencia:

- Dónde se utiliza o se hace referencia a ella
- Versión
- Tipo de licencia
- Información sobre la fuente (enlace al repositorio, nombre del autor, etc.)
- Estado de soporte y mantenimiento

El ejercicio anterior permite conocer de forma rápida las aplicaciones afectadas por un determinado CVE y actuar en consecuencia.

Una vez identificadas las dependencias, se comprobará en las bases de datos de vulnerabilidades y otros repositorios de información la existencia de vulnerabilidades conocidas. Las dependencias con vulnerabilidades conocidas se actualizarán, sustituirán o se realizará la gestión oportuna al respecto (gestión de vulnerabilidades).

La revisión de vulnerabilidades se realizará de forma periódica, al menos cada tres meses.

4.2.3.1.2 Construcción Segura (SB2)

El objetivo de la práctica en el segundo nivel de madurez es conseguir que el proceso de construcción esté optimizado y totalmente integrado en el flujo de trabajo de los equipos.

Línea de acción "Proceso de Construcción" (Stream A): Automatizar el proceso de construcción

El objeto de esta actividad es desarrollar un proceso de construcción eficiente que permita la integración de herramientas de seguridad dentro del mismo.

Se deberá automatizar el proceso de construcción de software de forma que este pueda ser ejecutado de forma consistente en cualquier instante. Para realizar la construcción de los artefactos no se debería requerir ninguna intervención manual. De esta forma, se reducen enormemente las probabilidades de que se produzcan errores humanos durante el proceso. El uso de herramientas de automatización incrementará la fiabilidad del proceso de construcción.

Al utilizar sistemas automáticos, el estado de seguridad de los productos finales depende en gran medida de la seguridad de las herramientas utilizadas para la construcción. Por ello, es crítico realizar el correspondiente *hardening* y garantizar el adecuado mantenimiento del conjunto de herramientas empleadas para la construcción. Por ejemplo, hay que prestar especial atención a los *interfaces* de dichas herramientas, como aquellos basados en aplicaciones web, con el fin de protegerlos contra accesos ilegítimos o no controlados. La exposición de nuestro sistema de construcción, por ejemplo a Internet, podría permitir a un atacante manipular la integridad del proceso o incorporar código malicioso en los productos finales.

Durante el proceso de construcción, puede ser preciso el acceso a credenciales y otro tipo de información secreta, como credenciales de acceso a otros sistemas de información o certificados de firma de código. Esta información sensible deberá ser gestionada extremando los cuidados para evitar que sea relevada a sujetos no autorizados, ya sea personal interno o externo a la organización.

Para garantizar la integridad de los artefactos generados, es buena práctica firmarlos utilizando un certificado que identifique a la organización o unidad de negocio que los ha generado.

Adicionalmente, se añadirá a la secuencia de operaciones del proceso de construcción (pipeline) verificaciones de seguridad automatizadas, tales como análisis estático de código mediante herramientas SAST, ejecución de pruebas automáticas de seguridad, análisis dinámico con herramientas DAST, etc. De esta forma, se aprovecha la automatización de la cadena de producción para aumentar la seguridad del producto final.

Línea de acción "Dependencias del software" (Stream B): Revisar las la seguridad de las dependencias de la aplicación

La finalidad es tener una visión clara y precisa de los problemas de seguridad conocidos en las dependencias de la aplicación y elaborar una lista de dependencias permitidas para el desarrollo de software dentro de la organización

Se deberán evaluar las dependencias utilizadas por las diferentes aplicaciones y se establecerá la relación de aquellas que son aceptadas y aprobadas para su uso en los proyectos.

Las dependencias aceptadas se almacenarán en un repositorio central de dependencias a partir del cual se construirá el software de la organización. De esta forma, se garantiza que cualquier artefacto construido solo hace uso de componentes autorizados.

Se realizará una revisión periódica de las dependencias aprobadas para garantizar que:

- Siguen estando correctamente licenciadas
- No hay vulnerabilidades conocidas y significativas que afecten a sus aplicaciones
- La dependencia sigue recibiendo soporte y mantenimiento activo
- Se está utilizando una versión actual
- Existe una razón válida o justificada para incluir la dependencia

Se establecerán mecanismos que permitan reaccionar adecuadamente ante no conformidades derivadas del uso de dependencias, tales como cambios en el tipo de licencia o aparición de nuevas vulnerabilidades significativas. Idealmente, se deberá automatizar la búsqueda de dependencias, generar las alertas correspondientes y asignar las tareas de corrección asociadas a los equipos de desarrollo.

4.2.3.1.3 Construcción Segura (SB3)

El objetivo de la práctica en este nivel de madurez es que el proceso de construcción ayude a evitar que los defectos conocidos se acaben propagando al entorno de producción.

Línea de acción "Proceso de Construcción" (Stream A): Imponer línea de base de seguridad durante la construcción

Esta actividad tiene como objetivo establecer y hacer que se aplique durante la construcción una línea base o conjunto de requisitos mínimos de seguridad.

Para forzar el cumplimiento de los requisitos de seguridad definidos en la línea base, se establecerán las comprobaciones de seguridad pertinentes que deben realizarse durante el proceso de compilación y construcción (*pipeline* de construcción). También será necesario establecer los criterios mínimos que deberán satisfacerse para que el proceso de compilación y construcción pueda llegar a buen término. Dichos criterios podrían variar en función de los perfiles de riesgo de las distintas aplicaciones. Por ejemplo, se puede establecer que aplicaciones con un perfil de riesgo bajo puedan incorporar dependencias con algunas vulnerabilidades de nivel medio, mientras que para las aplicaciones con un perfil de riesgo mayor no se permita la incorporación de dependencias externas que contengan vulnerabilidades conocidas de nivel medio o alto.

Para que esta medida sea realmente efectiva, en el proceso automático de construcción se llevarán a cabo las comprobaciones necesarias para garantizar el cumplimiento de los criterios de seguridad previamente establecidos y se

obligará a interrumpir el proceso de compilación y construcción en caso de que dichos criterios no se cumplan. Es decir, los criterios establecidos en la línea base, constituyen un umbral mínimo de seguridad que las aplicaciones deben satisfacer. Si una aplicación no supera dicho umbral, simplemente no se podrá construir.

En las comprobaciones de seguridad llevadas a cabo en el proceso de construcción se detectan los problemas que están por debajo del umbral mínimo, se realizarán las advertencias y se registrarán en un sistema centralizado para gestionar su tratamiento.

En caso de que se considere necesario o sensato, se podrá implementar un mecanismo de excepción para permitir que el proceso de construcción no sea interrumpido si el riesgo de una vulnerabilidad concreta puede ser mitigado, aceptado por la organización debido a circunstancias excepcionales como pueden ser ciertas necesidades de negocio o emergencias. No obstante, se deberá asegurar que estos casos son aprobados de forma explícita y se registrará el hecho y justificación de la excepción realizada.

Para facilitar la ejecución efectiva de esta actividad, es casi imprescindible contar con un sistema y flujo de construcción automático basado en la ejecución secuencial de operaciones de construcción. En el caso de no disponer de un sistema con dichas características o con limitaciones para poder abortar el proceso de construcción de forma automática, se deberá asegurar el mismo proceder mediante otras medidas, como una política clara, revisiones manuales posteriores a la construcción y auditorías periódicas para detectar desviaciones.

En lo que respecta a la integridad de código, la firma de los artefactos resultantes de la construcción se gestionará en un servidor centralizado independiente con el fin de evitar la exposición de la calve privada y certificado de firma en el sistema que ejecuta la compilación y construcción. Adicionalmente, siempre que la tecnología empleada para la construcción lo permita, se deberá aplicar un método determinista que produzca artefactos reproducibles byte a byte. Es decir, que dado un mismo código fuente de una determinada versión de una aplicación, cuando sea compilado, siempre generará el mismo binario o artefacto de salida. Esto también se conoce como compilación o construcción reproducible.

Línea de acción "Dependencias del software" (Stream B): Verificar las dependencias de las aplicaciones

El objetivo de esta actividad es gestionar los problemas de seguridad de las dependencias utilizadas de la misma forma que se gestionan los defectos introducidos en el código propio y controlar

Se establecerá una lista blanca de dependencias y versiones aprobadas, y se comprobará durante el proceso de construcción que no se utilizan dependencias no permitidas. El proceso de construcción fallará ante la presencia de una dependencia que no figure en la lista blanca. De igual forma que en el caso anterior, en caso de que se considere necesario o sensato, se podrá implementar un mecanismo de excepción para permitir que el proceso de construcción no se interrumpa ante determinadas circunstancias excepcionales como pueden ser ciertas necesidades de negocio, emergencias, aceptación explícita del riesgo o mecanismo de mitigación.

Adicionalmente, se llevará a cabo una verificación de la seguridad de las dependencias utilizadas de la misma forma que se realiza con el código propio, utilizando las mismas herramientas SAST y analizando las dependencias transitivas.

Al igual que las vulnerabilidades del código propio, se gestionarán los problemas de seguridad con los proveedores de las dependencias, incluyendo el establecimiento de acuerdos de nivel de servicio para su resolución de problemas. Hay que tener en cuenta que no siempre es posible establecer acuerdos de nivel de servicio, como por ejemplo en el caso de dependencias de código abierto. En este caso, se analizarán los problemas más relevantes y la posibilidad de aplicar medidas compensatorias a tiempo.

De forma similar a como se procede con el código propio, se realizará seguimiento de todos los problemas identificados y de su estado utilizando el mismo sistema de seguimiento de defectos. Igualmente, la construcción de los artefactos fallará siempre que las dependencias incluidas contengan vulnerabilidades que estén por encima del umbral definido.

4.2.3.2 Práctica “Despliegue Seguro” (SD)

La práctica de Despliegue seguro se centra en garantizar que la seguridad y la integridad de las aplicaciones desarrolladas no se vean comprometidas durante el despliegue.

Al igual que el resto de prácticas de SAMM, esta práctica establece dos líneas de actuación. La primera, se enfoca en la automatización del despliegue con el fin de eliminar los posibles errores asociados a la intervención manual. Por tanto, promueve la automatización de las actividades del proceso de despliegue en la medida de lo posible y la evaluación del éxito de las operaciones a partir de los resultados de comprobaciones automáticas integradas. Adicionalmente fomenta la separación de funciones (Separation of Duties), pues propone hacer responsables del despliegue a personal con formación específica que no sean los propios desarrolladores de las aplicaciones.

Por otro parte, la segunda línea de actuación se centra en la privacidad e integridad de datos sensibles, tales como contraseñas, direcciones IP, claves de cifrado y en general, cualquier secreto necesario para que las aplicaciones puedan funcionar en los entornos de producción. Esta práctica, en su versión menos sofisticada, aboga por que estos secretos de producción se trasladen de repositorios y archivos de configuración a *vaults* digitales debidamente protegidos. En una versión más sofisticada de la práctica, se impulsa que los secretos se generen dinámicamente en el momento del despliegue, así como la existencia de controles de seguridad que detecten y mitiguen la presencia de cualquier secreto no protegido en el entorno de operación.

En la siguiente figura, se muestran las actividades asociadas a los distintos niveles de madurez comprendidos en la práctica.

Maturity level		Stream A Deployment Process	Stream B Secret Management
1	Deployment processes are fully documented.	Formalize the deployment process and secure the used tooling and processes.	Introduce basic protection measures to limit access to your production secrets.
2	Deployment processes include security verification milestones.	Automate the deployment process over all stages and introduce sensible security verification tests.	Inject secrets dynamically during deployment process from hardened storages and audit all human access to them.
3	Deployment process is fully automated and incorporates automated verification of all critical milestones.	Automatically verify integrity of all deployed software, indenendently on whether it's internally or externally developed.	Improve the lifecycle of application secrets by regularly generating them and by ensuring proper use.

Figura 14. Actividades de la práctica Despliegue Seguro. Fuente OWASP SAMM

Como se puede observar en la figura anterior, se definen las dos líneas acción para esta práctica: Proceso de Despliegue (Stream A) y Gestión de Secretos (Stream B).

La primera línea de acción se centra en el desarrollo de un proceso de despliegue repetible y consistente que garantiza que sólo se desplieguen artefactos de software correctos en la producción.

La segunda línea de acción pone el foco en la protección de la información sensible (credenciales, claves, direcciones IP, etc.) necesaria tanto para el despliegue de las aplicaciones como para el funcionamiento de las mismas.

A continuación, se describirá en detalle cada una de las seis actividades (dos por cada nivel de madurez) comprendidas en la práctica.

4.2.3.2.1 Despliegue Seguro (SD1)

El objetivo de la práctica en el nivel inicial de madurez es conseguir que el proceso de despliegue de las aplicaciones esté completamente documentado.

Línea de acción "Proceso de Despliegue" (Stream A): Proceso de despliegue repetible

Esta actividad tiene como objetivo establecer un proceso de despliegue repetible y coherente que garantice que solo se desplieguen en producción los artefactos de software correctos, reduciendo el riesgo provocado por errores humanos.

Para lograr el objetivo mencionado, la organización deberá definir el proceso de despliegue en todas sus fases, desglosándolo en un conjunto de instrucciones claras que deberá seguir una persona o una herramienta automatizada. Esta definición debe describir todo el proceso de principio a fin, de modo que pueda seguirse de forma sistemática para cada despliegue y obtener los mismos resultados.

La definición del proceso de despliegue deberá ser almacenada y gestionada de forma centralizada y será accesible para todo el personal pertinente. Se evitará almacenar o distribuir diferentes copias de la documentación para evitar el uso de múltiples versiones, algunas de las cuales pueden ser obsoletas o incompletas.

El despliegue de las aplicaciones en producción se llevará a cabo (manual[4] o automáticamente) por personal específico distinto de los desarrolladores y se deberá asegurar que el equipo de desarrollo no precisa de acceso directo al entorno de producción para llevar a cabo su trabajo.

Todas las herramientas utilizadas para el despliegue en producción serán revisadas, asegurándose de que los proveedores las mantienen de forma activa y de que están debidamente actualizadas con sus correspondientes parches de seguridad. Hay que tener en consideración que la mayoría de estas herramientas requieren acceso al entorno de producción, por lo que su seguridad es extremadamente crítica. Por ello, éstas se configurarán debidamente para que se ajuste a las directrices del proveedor, a las mejores prácticas de seguridad (*hardening)*, y se configurarán los controles de acceso a dichas herramientas de acuerdo con el principio de mínimo privilegio.

Adicionalmente, se procurará que el personal con acceso al entorno de producción encargado de las labores de despliegue tenga un nivel mínimo de formación, cualificación o certificación para garantizar su competencia en esta materia.

Línea de acción "Gestión de Secretos" (Stream B): Proteger los secretos en la configuración y el código fuente

El objetivo de esta actividad es definir y limitar el acceso a la información sensible del entorno de producción.

Los desarrolladores no deben tener acceso a los secretos o credenciales de los entornos de producción. Por ello, no se utilizarán los secretos de producción en los archivos de configuración de los entornos de desarrollo, preproducción o de pruebas. Del mismo modo, no se almacenarán secretos sin protección en los archivos de configuración almacenados en los repositorios de código.

En general, siempre que se guarden secretos o datos sensibles de producción, se deberán cifrar previamente antes de almacenar los ficheros en el repositorio o almacenamiento correspondiente. Se deberá evaluar la utilización de herramientas específicas diseñadas para ello.

Se establecerá un mecanismo apropiado para proteger debidamente los secretos de producción. Por ejemplo:

- Haciendo que personas específicas añadan la información de secretos a los archivos de configuración relevantes en el momento del despliegue (principio de separación de funciones)
- Cifrando los secretos de producción contenidos en los archivos de configuración por adelantado.

4.2.3.2.2 Despliegue Seguro (SD2)

El objetivo de la práctica en este nivel de madurez es automatizar el despliegue e incluir en el proceso hitos o puntos de control sobre la seguridad.

[4] En el nivel de madurez inicial no sería obligatorio que el proceso de despliegue se lleve a cabo con herramientas automáticas.

Línea de acción "Proceso de Despliegue" (Stream A): Automatizar el despliegue e integrar controles de seguridad en el proceso.

El objetivo de esta actividad es conseguir un proceso de implantación eficiente en el que estén integradas herramientas para establecer controles de seguridad durante el proceso.

Se automatizará el proceso de despliegue, de modo que no se necesiten acciones de configuración manual y se elimine el riesgo de error humano.

En el proceso automático de despliegue se integrarán comprobaciones de seguridad automatizadas, por ejemplo, utilizando herramientas de análisis dinámico de seguridad (DAST), exploración de vulnerabilidades y finalmente, si tiene sentido, verificando la integridad de los artefactos desplegados. El resultado de estas pruebas se almacenará de forma centralizada. En caso de que se detecte algún defecto, se realizarán las correspondientes notificaciones automáticas al personal competente y en caso de que se identifique algún problema que supere la criticidad predefinida, se detendrá el despliegue o se revertirá automáticamente. Si se considera necesario o sensato, se podrá implementar un mecanismo manual de excepción para permitir que el despliegue siga adelante si el riesgo puede ser mitigado o es aceptado por la organización debido a circunstancias excepcionales como pueden ser ciertas necesidades de negocio o emergencias. En estos casos, se deberá asegurar la aprobación explícita y se registrará el hecho y justificación de la excepción realizada.

Se contabilizarán y registrarán todos los despliegues, registrando la información relevante asociada, como quién lo ha realizado, la versión de software que se ha desplegado y cualquier otra información de interés.

Línea de acción "Gestión de Secretos" (Stream B): Añadir los secretos de la aplicación durante el proceso despliegue

El objetivo de esta actividad es detectar la presencia de secretos en los ficheros de configuración o en el propio código fuente y evitar de posibles fugas de información sensible.

Se establecerá un mecanismo automático para añadir "al vuelo" las credenciales y secretos a los archivos de configuración durante el proceso de despliegue. De esta manera, ni los desarrolladores ni personal encargado del despliegue tienen acceso a la información sensible.

Haz que el sistema utilizado para almacenar la información de secretos e inyectarla en la fase de despliegue deberá ser robusto desde el punto de vista de la seguridad. La información sensible se cifrará tanto en estado de reposo como en tránsito y se gestionarán conforme al principio de mínimo privilegio. Por ejemplo, un desarrollador podría tener acceso a ciertos secretos del entorno de desarrollo, pero no del entorno de producción y desde luego, nunca del entorno de producción.

Una de los errores más comunes que suelen darse es el almacenamiento de secretos en los repositorios de código. Por ello, se realizarán comprobaciones y búsquedas automáticas periódicas para que detectar la presencia de secretos en los repositorios de código y archivos de configuración. Cuando se detecte un secreto almacenado indebidamente se marcarán para su eliminación posterior.

4.2.3.2.3 Despliegue Seguro (SD3)

El objetivo de la práctica en este nivel de madurez es lograr que el proceso de despliegue esté totalmente automatizado e incorpore la verificación automática de todos los controles de seguridad críticos.

Línea de acción "Proceso de Despliegue" (Stream A): Verificar la integridad de los artefactos de despliegue

La finalidad de esta actividad es asegurar la integridad de los artefactos que se despliegan a la producción.

Los artefactos a desplegar, serán firmados en el momento de la compilación/construcción. Durante el proceso de despliegue se realizará la verificación automática de la integridad del software que se despliega, comprobando las firmas con certificados de confianza. Esta acción se llevará a cabo tanto para los componentes desarrollados y construidos por la propia organización como con otros componentes de terceros. En caso de que las firmas no puedan ser verificadas o falle el proceso de verificación, no se realizará el despliegue en producción.

Se deberá definir y mantener actualizada una lista de certificados de confianza. Si en dicha lista se incluyen certificados de terceras partes como proveedores o desarrolladores externos, se comprobará periódicamente la vigencia de los certificados así como el estado de confianza de los mismos, en línea con la gobernanza más amplia de la organización en relación a los proveedores de confianza.

Durante el proceso de despliegue se realizará una aprobación manual al menos una vez. Esto se llevará a cabo siempre que dicha comprobación y aprobación humana sea significativamente más precisa o útil que una automatizada.

Línea de acción "Gestión de Secretos" (Stream B): Gestionar el ciclo de vida de los secretos de las aplicaciones

El objetivo es reducir el riesgo y detectar usos indebidos de los secretos del entorno de producción.

Los secretos del entorno de producción deberán ser gestionados adecuadamente. Se implantará un proceso de gestión del ciclo de vida que garantice la renovación periódica de los mismos.

Cada aplicación e incluso cada instancia de una aplicación, utilizará un conjunto de secretos diferente. Esta práctica garantiza que el comportamiento inesperado de la aplicación pueda ser rastreado y analizado adecuadamente.

La gestión del ciclo de vida de los secretos puede ser gestionada de forma eficiente mediante herramientas que, tras un cambio, ayuden a la actualización automática de los mismos en todos los sitios necesarios.

Se deberá asegurar que todos los accesos a los secretos, tanto de lectura como de escritura, quedan registrados debidamente en un repositorio central. Estos registros deberán ser revisados con cierta regularidad para detectar posibles usos anormales. En caso de que se detecte algún problema, se analizarán las

causas del mismo y se incorporará a la práctica de gestión de defectos para asegurar la resolución de cualquier situación que se considere inaceptable.

4.2.3.3 Práctica “Gestión de Defectos” (DM)

La práctica de gestión de defectos (DM) se enfoca en la recopilación, registro y análisis de los defectos de seguridad del software y la documentación detallada de los mismos para promover la toma de decisiones basadas en métricas.

Esta práctica se ocupa, por un lado del proceso de tratamiento y gestión de defectos del software para garantizar que los productos entregados tengan un nivel de seguridad adecuado. Por otra parte, también pone el foco en el enriquecimiento de la información sobre defectos y en la extracción de métricas para establecer mejoras a medio plazo y orientar las acciones de mejora de la seguridad de las aplicaciones y del programa global de garantía de seguridad de la organización.

En su versión más avanzada, esta práctica requiere una gestión de defectos totalmente formalizada e independiente, así como información correlacionada para detectar tendencias en malas prácticas de seguridad e influir en la estrategia global de la organización.

En la siguiente figura, se muestran las actividades asociadas a los distintos niveles de madurez comprendidos en la práctica.

Maturity level		Stream A Defect Tracking	Stream B Metrics and Feedback
1	All defects are tracked within each project.	Introduce a structured tracking of security defects and make knowledgeable decisions based on this information.	Regularly go over previously recorded security defects and derive quick wins from basic metrics.
2	Defect tracking used to influence the deployment process.	Rate all security defects over the whole organization consistently and define SLAs for particular severity classes.	Collect standardized defect management metrics and use these also for prioritization of centrally driven initiatives.
3	Defect tracking across multiple components is used to help reduce the number of new defects.	Enforce the predefined SLAs and integrate your defect management system with other relevant tooling.	Continuously improve your security defect management metrics and correlate it with other sources.

Figura 15. Actividades de la práctica Gestión de Defectos. Fuente OWASP SAMM

Como se puede observar en la figura anterior, se definen las dos líneas acción para esta práctica: Seguimiento de Defectos (Stream A) y Métricas y Retroalimentación (Stream B).

La primera línea de acción se ocupa del seguimiento de defectos a través de la recopilación y el seguimiento de todos los problemas potenciales del software, desde los defectos de arquitectura hasta los problemas de codificación y las vulnerabilidades en tiempo de ejecución.

Por su parte, la línea de “Métricas y Retroalimentación” se enfoca en cómo el seguimiento de defectos puede impulsar la mejora de las actividades de seguridad dentro de la organización a través de métricas y la correspondiente retroalimentación sobre otros procesos.

A continuación, se describirá en detalle cada una de las seis actividades (dos por cada nivel de madurez) comprendidas en la práctica.

4.2.3.3.1 Gestión de Defectos (DM1)

El objetivo de la práctica en este nivel es asegurar el seguimiento de todos los defectos de seguridad asociados a cada proyecto, desde los defectos de arquitectura hasta los problemas de codificación y las vulnerabilidades en tiempo de ejecución.

Línea de acción Seguimiento de Defectos (Stream A): Seguimiento centralizado de defectos

La finalidad de esta actividad es conseguir una visión clara de los defectos de seguridad conocidos que afectan a determinadas aplicaciones.

En primer lugar, se deberá establecer la definición e interpretación común de lo que es un defecto de seguridad y los mecanismos más comunes para identificarlos. Por ejemplo: evaluaciones de modelos de amenazas, pruebas de penetración, resultado de herramientas SAST o DAST, etc.

Adicionalmente, se deberá fomentar una cultura de transparencia y de evitar la culpabilización del personal de los equipos de desarrollo por introducir o identificar defectos de seguridad.

Los defectos de seguridad se registrarán en una ubicación o repositorio perfectamente definido. La ubicación podría no ser la misma para toda la organización, pero si deberá garantizar la obtención de una visión general de todos los defectos que afectan a una aplicación o proyecto concreto en un momento dado. El acceso a la información de defectos de seguridad deberá disponer de las correspondientes reglas de acceso para mitigar el riesgo de fuga de esta información sensible.

Los defectos de seguridad deberán ser clasificados de forma cualitativa, aunque sea mediante una clasificación simple, que permita priorizar los esfuerzos de resolución.

Por último, se realizarán esfuerzos para evitar la duplicidad de información y la existencia de falsos positivos, pues esto desvirtúa en parte el proceso de gestión.

Línea de acción Métricas y retroalimentación (Stream B): Definir las unas métricas básicas

Uno de los beneficios de esta actividad es la obtención de información general de los defectos registrados para identificar acciones de mejora (*quick wins*) del programa de garantía de la seguridad.

Periódicamente (típicamente una vez al año), se revisarán en cada uno de los equipos de desarrollo los defectos de seguridad registrados, tanto los resueltos como los que aún estén abiertos, con el fin de extraer unas métricas básicas a partir de la información disponibles. Por ejemplo:

- El número total de defectos frente al número total de actividades de verificación. Esto podría dar una idea de si se están buscando defectos con la intensidad y calidad adecuadas.
- Los componentes de software en los que residen los defectos. Esto puede indicar dónde sería más necesario poner el foco de atención, y dónde es más probable que los defectos de seguridad vuelvan a aparecer en el futuro.

- El tipo o categoría del defecto, que podría indicar las áreas de conocimiento en las que los equipos de desarrollo necesitan más formación.
- La severidad de los defectos, que puede ayudar a la organización y los equipos a comprender la exposición al riesgo del software.

En función de la información obtenida a través de estas métricas, se identificarán las actividades que nos pueden conducir a identificar los *quick wins* en materia de mejora de la seguridad del software. Por ejemplo, sesiones de intercambio de conocimientos sobre un tipo de vulnerabilidad concreta, acciones de reciclaje de conocimientos, ejecución de análisis de seguridad, etc.

4.2.3.3.2 Gestión de Defectos (DM2)

El objetivo principal de la práctica en este nivel de madurez es que la actividad de seguimiento de defectos tenga influencia en la priorización de las actividades de los procesos de desarrollo e implantación.

Línea de acción Seguimiento de Defectos (Stream A): Calificación y seguimiento de defectos

El objetivo de esta actividad es clasificar los defectos de seguridad y definir unas expectativas claras sobre su tratamiento.

En primer lugar, se establecerá en toda la toda la organización una metodología de clasificación de defectos bien definida y coherente, basada en factores como la probabilidad y el impacto estimado en caso de que el defecto (vulnerabilidad) sea explotado. Esta clasificación permitirá identificar las aplicaciones que necesitan mayor atención e inversión en esfuerzo para corregir los problemas.

Se establecerán compromisos en forma de acuerdos de nivel de servicio (SLA) para la corrección de defectos de seguridad en función de su grado de criticidad, supervisando e informar regularmente sobre cualquier incumplimiento de dichos compromisos. Adicionalmente, se definirá un procedimiento de actuación para los casos en los que no sea factible o económicamente viable arreglar un defecto dentro del tiempo definido por los SLA. Como mínimo, este procedimiento deberá garantizar que todas las partes interesadas tengan una sólida comprensión del riesgo existente. También podrá contemplar el establecimiento de controles de seguridad compensatorios para estos casos.

En el caso de que no exista un SLA formal para los defectos de menos gravedad, se deberá articular algún mecanismo que asegure que los equipos responsables reciban una visión general periódica sobre los problemas que afectan a las aplicaciones y que entiendan cómo afectan a la seguridad los problemas existentes o como la conjunción de múltiples problemas amplifican el riesgo global de las aplicaciones.

Línea de acción Métricas y retroalimentación (Stream B): Definir métricas avanzadas

El objetivo de esta actividad el aprendizaje de la organización en materia de seguridad del software a partir de los defectos de seguridad encontrados.

Se definirán, recopilarán y calcularán nuevas métricas unificadas en toda la organización. Por ejemplo:

- Cantidad total de actividades de verificación y defectos identificados.
- Tipos y gravedad de los defectos identificados.
- Tiempo de detección y tiempo de resolución de los defectos.
- Ventanas de exposición de los defectos presentes en los sistemas vivos.
- Número de vulnerabilidades reabiertas.
- Cobertura de las actividades de verificación para determinados componentes de software.
- Cantidad de riesgo aceptado.
- Ratio de incidentes de seguridad causados por defectos de seguridad desconocidos o no documentados.

Se generará un informe periódico (por ejemplo, mensual) para las partes interesadas correspondientes. Por lo general, el informe irá dirigido a directores, responsables de seguridad e ingenieros de proyecto. El contenido de este informe se utilizará para realizar acciones en el contexto de la estrategia global de seguridad. Por ejemplo, para mejorar o ampliar la formación de los equipos, invertir en actividades de verificación de la seguridad, etc.

Periódicamente se compartirán con todos los equipos de la organización los detalles técnicos más destacados o interesantes de los defectos, incluida las estrategias de corrección. Este intercambio de conocimientos contribuirá a una amplificación del efecto del aprendizaje a toda la organización y a limitar la aparición de defectos similares en el futuro.

4.2.3.3.3 Gestión de Defectos (DM3)

El objetivo principal de la práctica en este nivel de madurez es conseguir que el seguimiento de defectos de los diversos componentes se emplee para ayudar a reducir de forma global aparición de nuevos defectos.

Línea de acción Seguimiento de Defectos (Stream A): Imponer SLAs para la gestión de defectos

El objeto de esta actividad es asegurar que los defectos de seguridad se tratan dentro de los SLA predefinidos.

Se implantará un sistema de emisión de alertas automatizadas sobre los defectos de seguridad si el tiempo de resolución incumple los SLA definidos. Dichos defectos deberán transferirse de forma automáticamente al proceso de gestión de riesgos con el fin de que se califiquen mediante una metodología cuantitativa rigurosa y coherente con la organización. Se evaluará además, el efecto global de los diferentes defectos, es decir, cómo los defectos particulares influyen en otros o cómo se amplifican unos a otros al nivel de toda la organización.

El conocimiento adquirido en todo el proceso de gestión de defectos se utilizará para priorizar, introducir y hacer seguimiento de controles compensatorios que mitigan los riesgos de negocio.

Adicionalmente, el sistema de gestión de defectos se integrará con los sistemas automatizados derivados de otras prácticas del modelo. Por ejemplo:

- Construcción y despliegue: El proceso de construcción/despliegue fallará y se interrumpirá si los defectos de seguridad de un artefacto final superan el umbral de severidad, a menos que se aplique explícitamente un criterio de excepción debidamente aprobado por el responsable correspondiente.
- Monitorización: se vigilará en los entornos de producción las posibles explotaciones de los defectos conocidos y se alertará en caso de que dicha explotación llegue a materializarse.

Línea de acción Métricas y retroalimentación (Stream B): Utilizar las métricas para mejorar la estrategia de seguridad

El objetivo de la actividad es utilizar las métricas para mejorar la estrategia de seguridad de la organización.

La idea es utilizar la información recopilada en el proceso de gestión de defectos para optimizar la estrategia de seguridad de la organización. Para ello, se revisarán periódicamente (al menos una vez al año) las métricas de gestión de defectos recopiladas y se evaluara la relación entre el esfuerzo realizado para su obtención y el valor que aporta dicha información para la organización. Se determinará, con conocimiento de causa, si se descartan aquellas métricas que no aportan el valor esperado. Adicionalmente, si es posible, se establecerán y automatizarán acciones de verificación de la calidad de los datos recogidos y se asegurará la mejora sostenible de los mismos en caso de que se detecte alguna anomalía.

Se agregará y relacionará la información proporcionada por las métricas de gestión de defectos con las métricas de gestión de incidentes de seguridad o inteligencia de amenazas (Cyber Threat Intellingence). Los resultados obtenidos se emplearán como como entrada para otras iniciativas en toda la organización. Por ejemplo:

- Planificación de la formación en materia de seguridad para el personal
- Mejora de las actividades de verificación de la seguridad de desarrollo de software tanto interno como externo
- Gestión de la cadena de suministro: realización de auditorías de seguridad de las organizaciones asociadas
- Seguimiento de los ataques contra su infraestructura y aplicaciones
- Inversión en infraestructura de seguridad o controles compensatorios
- Dotación de personal de su equipo de seguridad y establecimiento del presupuesto de seguridad

4.2.4 Función de negocio Verificación

Esta función de negocio pone el foco sobre los procesos y actividades relacionados con la forma en que la organización verifica y prueba los artefactos producidos a lo largo del desarrollo del software. Esto suele incluir actividades relacionadas con el aseguramiento de la calidad, como por ejemplo las pruebas

de requisitos funcionales y de seguridad, pero también incluye otras actividades de revisión y evaluación.

Esta función se desarrolla mediante tres prácticas:

- **Evaluación de la arquitectura**: validación de la seguridad y el cumplimiento de la arquitectura del software y la infraestructura de apoyo.
- **Pruebas orientadas a los requisitos**: pruebas de seguridad tanto positivas (verificación de controles) como negativas (pruebas de mal uso/abuso) basadas en los requisitos o “historias de usuario”.
- **Pruebas de seguridad**: detección y resolución de problemas básicos de seguridad mediante automatización, dejando que las pruebas manuales se centren en vectores de ataque más complejos.

A continuación se describirá cada una de las tres prácticas mencionadas.

4.2.4.1 Práctica de Evaluación de la Arquitectura (AA)

La práctica de Evaluación de la Arquitectura (AA) garantiza que la arquitectura de la aplicación y de la infraestructura cumple adecuadamente todos los requisitos de seguridad y de conformidad pertinentes, y mitiga suficientemente las amenazas a la seguridad identificadas.

Al igual que en el resto de prácticas de SAMM, se establecen dos líneas o flujos de actuación. La primera línea de acción está orientada a verificar que se satisfacen los requisitos de seguridad y cumplimiento identificados en las prácticas de Política y Cumplimiento y Requisitos de Seguridad. La segunda línea de acción se centra en la arquitectura, inicialmente asegurando las mitigaciones contra las amenazas típicas, y posteriormente tratando las amenazas específicas identificadas en la Práctica Evaluación de Amenazas (TA).

En el nivel de madurez más elevado, se formaliza el proceso de revisión de la arquitectura de seguridad, evaluando continuamente la eficacia de los controles de seguridad de la arquitectura, su escalabilidad y su alineación estratégica. Los puntos débiles identificados y las posibles mejoras realimentan la práctica de la Práctica Arquitectura de Seguridad (SA) para mejorar las arquitecturas de referencia establecidas.

En la siguiente figura, se muestran las actividades asociadas a los distintos niveles de madurez comprendidos en la práctica.

Maturity level		Stream A Architecture Validation	Stream B Architecture Mitigation
1	Review the architecture to ensure baseline mitigations are in place for typical risks.	Identify application and infrastructure architecture components and review for basic security provisioning	Ad-hoc review of the architecture for unmitigated security threats.
2	Review the complete provision of security mechanisms in the architecture.	Validate the architecture security mechanisms	Analyze the architecture for known threats.
3	Review the architecture effectiveness and feedback results to improve the security architecture.	Review of the architecture components' effectiveness	Feed the architecture review results back into the enterprise architecture, organization design principles & patterns, security solutions and reference architectures.

Figura 16. Actividades de la práctica Verificación. Fuente: OWASP SAMM

Como se puede observar en la figura anterior, se definen las dos líneas acción para esta práctica: Validación de la Arquitectura (Stream A) y Mitigaciones de la Arquitectura (Stream B).

La primera línea de acción se centra en la validación de la arquitectura de la aplicación y de la infraestructura que la soporta para confirmar que estas cumplen con los objetivos y requisitos de seguridad establecidos.

Por su parte, la línea relacionada con la mitigación se centra en garantizar que todas las amenazas identificadas durante la fase de evaluación de la amenazas se mitigan adecuadamente, así como que los modelos de arquitecturas de referencia utilizadas por la organización se actualicen convenientemente para abordar cualquier amenaza no tratada.

A continuación, se describirá en detalle cada una de las seis actividades (dos por cada nivel de madurez) comprendidas en la práctica.

4.2.4.1.1 Evaluación de la Arquitectura (AA1)

El objetivo de la práctica en el nivel inicial de madurez es revisar la arquitectura de las aplicaciones para asegurar que se han implementado mitigaciones de base para los riesgos de seguridad típicos.

Línea de acción "Validación de la Arquitectura" (Stream A): Evaluar la arquitectura de la aplicación

El objeto de esta actividad es comprender la arquitectura de alto nivel de las aplicaciones y valorar las medidas de seguridad básicas establecidas.

Para ello, se creará una visión de la arquitectura global de cada aplicación y se examinará para evaluar la provisión de mecanismos de seguridad generales como la autenticación, la autorización, la gestión de usuarios y derechos, la comunicación segura, la protección de datos, la privacidad, la gestión de claves, la gestión de registros, etc.

De forma general, en la arquitectura se identificará cualquier funcionalidad relacionada con la seguridad y se evaluará la correcta provisión de los mecanismos de control de forma *ad hoc*, desde las perspectivas de los usuarios anónimos, los usuarios autorizados y los roles específicos de la aplicación.

El examen se basará en los artefactos del proyecto, documentos de arquitectura o diseño, o entrevistas con responsables y el personal técnico. Se deberán considerar todos los elementos de la infraestructura, incluyendo aquellos sistemas, componentes y bibliotecas (incluidos los SDK) que no son específicos de la aplicación, pero que proporcionan apoyo directo para utilizar o gestionar las aplicaciones en la organización.

En caso de que durante la evaluación se detecte la ausencia de algún control de seguridad que sea necesario, este hecho se registrará como defecto siguiendo lo establecido por la Práctica "Gestión de Defectos" (DM).

Línea de acción "Mitigaciones de la Arquitectura" (Stream B): Evaluar la arquitectura para las amenazas típicas

Esta actividad persigue el objetivo de verificar y asegurar que la arquitectura definida ofrece suficiente protección contra las amenazas de seguridad típicas.

Las amenazas típicas de una arquitectura pueden estar relacionadas con suposiciones incorrectas y/o con una confianza excesiva en el suministro de mecanismos de seguridad como la autenticación, la autorización, la gestión de usuarios y derechos, la comunicación segura, etc. Adicionalmente, las amenazas típicas también pueden estar relacionadas con limitaciones conocidas o problemas de componentes o *frameworks* tecnológicos que forman parte de la solución y para los que no se ha establecido una mitigación suficiente.

Se revisará la arquitectura para detectar posibles amenazas de seguridad típicas. Esta revisión la deberá llevar a cabo personal técnico con conocimientos específicos sobre seguridad, contando con la aportación de arquitectos, desarrolladores, gestores y responsables, según sea necesario. De esta forma, se garantizará que la arquitectura hace frente a todas las amenazas comunes que los equipos de desarrollo, con conocimientos menos especializados en materia de seguridad, pueden haber pasado por alto.

4.2.4.1.2 Evaluación de la Arquitectura (AA2)

El objetivo de la práctica en este nivel de madurez es revisar la arquitectura de las aplicaciones para asegurar que esta realiza una provisión completa de mecanismos de seguridad.

Línea de acción "Validación de la Arquitectura" (Stream A): Verificar de forma metódica la arquitectura de la aplicación

La finalidad de esta actividad es la de establecer un proceso consistente de revisión de la arquitectura.

Se verificará que la arquitectura de la solución aborda todos los requisitos de seguridad y cumplimiento identificados. Para cada interfaz de la aplicación, repasará la lista de requisitos de seguridad y conformidad y se analizará la arquitectura para comprobar que se proveen los controles de seguridad pertinentes. Adicionalmente, se realizará análisis de las interacciones o flujos de datos para asegurar que los requisitos de seguridad se satisfacen adecuadamente en los distintos componentes de la aplicación. El documento de análisis resultante deberá mostrar que características específicas de nivel de diseño satisfacen cada uno de los requisitos de seguridad.

Los análisis mencionados se realizarán tanto en las interfaces internas, por ejemplo entre los diferentes niveles o capas, como en las externas, es decir, aquellas que constituyen la superficie de ataque.

Adicionalmente se identificarán y validarán de forma específica aquellas decisiones de diseño relevantes relacionadas con la arquitectura, especialmente cuando estas se desvíen del catálogo de soluciones y arquitecturas de seguridad establecidas por la organización.

Finalmente, como resultado del análisis y conclusiones de la evaluación, se expondrán los hallazgos detectados y se destacará cualquier requisito de seguridad que no esté contemplado de forma clara en el diseño de la aplicación. Estos hallazgos deberán registrarse como defectos siguiendo lo establecido por la Práctica "Gestión de Defectos" (DM).

Línea de acción "Mitigaciones de la Arquitectura" (Stream B): Verificar la arquitectura de la aplicación para las amenazas identificadas

El objetivo de esta actividad es asegurar que todas las amenazas identificadas a partir del modelo de amenazas sean tratadas adecuadamente.

Se revisará de forma sistemática todas y cada una de las amenazas identificadas durante las actividades del proceso de Práctica Evaluación de Amenazas (TA) y se analizará cómo la arquitectura diseñada mitiga dichas amenazas. Se deberá llevar a cabo un proceso estandarizado para analizar las arquitecturas de los sistemas y los flujos de datos existentes, identificando los objetivos de seguridad correspondientes que hacen frente a cada tipo de amenaza. Adicionalmente, para cada amenaza se identificarán las características de diseño de la arquitectura que la contrarrestan y se evaluará su eficacia.

Si están disponibles, se revisarán las decisiones arquitectónicas tomadas para comprender las limitaciones de la arquitectura y los controles compensatorios determinados durante la fase de diseño. Se analizará el impacto de estos elementos y el de cualquier suposición o hipótesis de seguridad en la que se base la seguridad del sistema diseñado y se realizará una nueva evaluación de los mismos con el fin de detectar incoherencias o debilidades.

Finalmente, tras las actividades de verificación, se enriquecerá el modelo de amenazas evaluado con la información obtenida, de forma que se relacione cada amenaza y su correspondiente impacto estimado con las contramedidas diseñadas. Se elaborará un documento que muestre dicho mapeo para que esté disponible y accesible para las partes interesadas.

Las actividades de revisión de amenazas deberás ser llevadas a cabo por personal con experiencia y conocimientos específicos sobre seguridad, contando con la aportación de arquitectos, desarrolladores, gestores y responsables, según sea necesario.

4.2.4.1.3 Evaluación de la Arquitectura (AA3)

El objetivo de la práctica en este nivel de madurez es revisar la efectividad de la arquitectura y retroalimentar el resultado para la mejora globalmente la arquitectura de seguridad de las aplicaciones.

Línea de acción "Validación de la Arquitectura" (Stream A): Verificar la eficacia de los componentes de seguridad

El objetivo de esta actividad es verificar la eficacia de los componentes de seguridad de los sistemas o aplicaciones.

Se revisará de forma periódica (por ejemplo, una vez al año) la efectividad de los mecanismos de seguridad proporcionados por los diferentes componentes de la arquitectura evaluando las capacidades de prevención, detección y respuesta de los mismos. Adicionalmente, se analizarán dichos componentes desde la perspectiva de su alineamiento con la estrategia general de seguridad. Se evaluará el grado de disponibilidad, escalabilidad y madurez de las soluciones de seguridad elegidas, con el fin de determinar si cumplen con lo requerido para ser utilizadas en un ámbito empresarial.

Si en alguna aplicación concreta tiene sentido utilizar determinadas soluciones de arquitectura específicas, es importante que estas se acaben incorporando al catálogo de modelos de arquitectura para asegurar la disponibilidad de la solución diseñada para futuros casos.

Finalmente, como resultado de la revisión, se expondrán los hallazgos detectados y se registrarán como defectos siguiendo lo establecido por la Práctica “Gestión de Defectos” (DM), con el fin de que se gestionen las mejoras de arquitectura correspondientes.

Línea de acción “Mitigaciones de la Arquitectura” (Stream B): Utilizar resultados de revisión para mejorar las arquitecturas de referencia

El objetivo de esta actividad es la mejora continua de la arquitectura empresarial a partir del las revisiones de arquitectura.

La organización aprovechará para mejorar la seguridad de su software. Para ello identificará las amenazas que quedan sin resolver en las arquitecturas de las aplicaciones y utilizará dicha información para realizar las adaptaciones que correspondan en las arquitecturas. Así, se formalizará un proceso de recopilación de información que permita identificar los hallazgos recurrentes en la arquitectura de las aplicaciones. Esta información actuará como desencadenante de proceso de identificación de las causas de las debilidades y tratamiento de las mismas. El resultado de este proceso realimentará a las actividades de la Práctica Arquitectura de Seguridad (SA) con el fin de actualizar el catálogo de arquitecturas de referencia establecidas por la organización.

4.2.4.2 Práctica de Pruebas Orientadas a Requisitos (RT)

El objetivo de la práctica de las pruebas basadas en los requisitos (RT) es garantizar que los controles de seguridad implementados funcionan según lo previsto y satisfacen los requisitos de seguridad establecidos en el proyecto. Para ello, se deberá desarrollar un conjunto de pruebas y tests de regresión de los requisitos de seguridad que se ejecutarán regularmente.

La práctica pone el foco sobre la importancia de realizar pruebas positivas y negativas. Las primeras verifican que los controles de seguridad de la aplicación satisfacen los requisitos de seguridad establecidos y validan su correcto funcionamiento. Estos requisitos suelen ser de naturaleza funcional. Por su parte, las pruebas negativas se ocupan de la calidad de la implementación de los controles de seguridad y tienen como objetivo detectar fallos de diseño y de implementación inesperados mediante pruebas de uso y abuso.

En su versión más avanzada, se fomenta la ejecución de pruebas de estrés de seguridad, como por ejemplo la denegación de servicio, y se promueve la mejora continua de la seguridad mediante la automatización de pruebas unitarias y pruebas de regresión de seguridad para todos los errores identificados y corregidos.

En la siguiente figura, se muestran las actividades asociadas a los distintos niveles de madurez comprendidos en la práctica.

Maturity level		Stream A Control Verification	Stream B Misuse/Abuse Testing
1	Opportunistically find basic vulnerabilities and other security issues.	Test for software security controls	Perform security fuzzing testing
2	Perform implementation review to discover application-specific risks against the security requirements.	Derive test cases from known security requirements	Create and test abuse cases and business logic flaw test
3	Maintain the application security level after bug fixes, changes or during maintenance.	Perform regression testing (with security unit tests)	Denial of service and security stress testing

Figura 17. Actividades de la práctica Pruebas Orientadas a los Requisitos. Fuente: OWASP SAMM

Como se puede observar en la figura anterior, se definen las dos líneas acción para esta práctica: Verificación de Controles (Stream A) y Pruebas de Mal uso/Abuso (Stream B).

La primera línea de acción valida que se cumplen los controles y requisitos de seguridad definidos mediante la ejecución de pruebas derivadas de los requisitos. Además, esta línea de acción se ocupa de evitar que, errores que han sido resueltos en versiones anteriores, se vuelvan a introducir en versiones posteriores de la aplicación mediante las correspondientes pruebas de regresión.

La segunda línea se centra en la realización de pruebas de uso indebido/abuso sobre funcionalidades o recursos del software que pueden ser objeto de abuso, con el fin de identificar los puntos débiles de la aplicación.

A continuación, se describirá en detalle cada una de las seis actividades (dos por cada nivel de madurez) comprendidas en la práctica.

4.2.4.2.1 Pruebas Orientadas a Requisitos (RT1)

El objetivo de la práctica en este nivel de madurez inicial es encontrar vulnerabilidades básicas y otros problemas de seguridad de forma oportunista.

Línea de acción "Verificación de Controles" (Stream A): Probar eficacia de los controles de seguridad

La finalidad es verificar la eficacia de los controles de seguridad típicos. Por tanto, se desarrollarán y ejecutarán pruebas de seguridad para verificar que los controles de seguridad básicos funcionan tal y como es esperado. A grandes rasgos, esto supone verificar el correcto funcionamiento de los controles de confidencialidad, integridad y disponibilidad de los datos y el servicio.

Las pruebas básicas de seguridad incluirán, como mínimo, pruebas de autenticación, control de acceso, validación de datos de entrada, codificación y "escapado" de datos [21] y controles de cifrado. El objetivo de estas pruebas es validar que dichos controles de seguridad se aplican correctamente.

Se realizarán pruebas sobre los controles de seguridad relevantes del software. Estas se realizarán manualmente o, idealmente, de forma automática cada vez que se realice algún cambio en los controles de seguridad. La verificación de los controles del software será obligatoria para todo el software que forme parte del programa SAMM.

Línea de acción "Pruebas de Mal Uso/Abuso" (Stream B): Realizar pruebas de fuzzing

El propósito de esta actividad es conocer y verificar el comportamiento de las aplicaciones cuando estas reciben datos de entrada inesperados.

El *fuzz testing* o *fuzzing* es una técnica de pruebas de software de caja negra que consiste en encontrar errores de implementación mediante la inyección automatizada de datos malformados o semimalformados.

Se realizarán pruebas enviando datos aleatorios o malformados (*fuzzing*) al componente software objeto de la prueba para intentar provocar su bloqueo o detectar comportamientos anómalos. Al menos, se buscarán vulnerabilidades en los componentes utilizando técnicas de fuzzing contra los principales parámetros de entrada. Los fallos o comportamientos anómalos detectados con estas pruebas se registrarán y analizarán para determinar su impacto en la seguridad de las aplicaciones.

La ventaja de las pruebas fuzz es la simplicidad del diseño de la prueba, y su falta de preconceptos sobre el comportamiento del sistema. Este enfoque tiene la ventaja de que se pueden detectar errores y vulnerabilidades que pruebas manuales realizadas por un humano o pruebas estructuradas a pasarían por alto.

Se promoverá la disponibilidad y uso de herramientas o *frameworks* de fuzzing para ampliar las capacidades de las herramientas de test.

4.2.4.2.2 Pruebas Orientadas a Requisitos (RT2)

El objetivo de la práctica en este nivel de madurez es la revisión de la implementación para descubrir los riesgos específicos de la aplicación.

Línea de acción "Verificación de Controles" (Stream A): Pruebas de seguridad a partir de los requisitos definidos

Esta actividad tiene como finalidad incluir o integrar los requisitos de seguridad en los escenarios de prueba. Es decir, se procederá de la misma forma que se haría con cualquier requisito funcional, creando los tests correspondientes para asegurar el cumplimiento de los requisitos.

A partir de los requisitos de seguridad, se identificarán e implementarán un conjunto de casos de prueba de seguridad para comprobar la correcta funcionalidad del software. Para ello, será necesario previamente conocer los objetivos de las pruebas, que deberán ser especificados en la definición de los requisitos de seguridad. Es decir, se derivarán los casos de pruebas de seguridad a partir de los requisitos de seguridad creados en las actividades de la Práctica Requisitos de Seguridad (SR).

Los requisitos de seguridad tienen naturaleza funcional y establecen la funcionalidad esperada (el qué) e, implícitamente, la implementación (el cómo). Estos requisitos también se denominan "requisitos positivos", ya que indican la funcionalidad esperada que puede validarse mediante pruebas de seguridad. Algunos ejemplos de requisitos positivos son:

- "La aplicación bloqueará al usuario tras seis intentos fallidos de inicio de sesión"
- "Las contraseñas deben tener un mínimo de seis caracteres alfanuméricos"
- "Un usuario no identificado no podrá acceder a la aplicación"

Así, la validación de los requisitos positivos consiste en evidenciar la funcionalidad esperada y los resultados se mostrarán forma de condición de fallo o éxito.

Los diferentes casos de prueba se identificarán y definirán durante la fase de análisis de requisitos y/o la fase de diseño. Se considerará las pruebas de los requisitos de seguridad como parte de las pruebas funcionales del software y, por tanto, deberán recibir el mismo tratamiento.

Línea de acción "Pruebas de Mal Uso/Abuso" (Stream B): Definir Casos de abuso de seguridad a partir de los requisitos

El objetivo es detectar fallos en la lógica de negocio de la aplicación.

Los casos de mal uso y abuso describen escenarios de mal uso y uso malintencionado sobre la aplicación, detallando cómo podría proceder un atacante para obtener algún beneficio o causar daños a la organización. Se definirán casos de uso y abuso para utilizar o explotar las debilidades de los controles con el fin de atacar a una aplicación. A partir de los modelos de casos de uso y abuso definidos, se identificarán las pruebas de seguridad específicas que exploten directa o indirectamente los escenarios de abuso.

Los casos de abuso de la funcionalidad, a veces denominados "ataques de lógica de negocio", dependen del diseño y la implementación de las funciones y características de la aplicación. Por ello, es necesario un conocimiento amplio de la aplicación y su lógica de negocio para identificar y definir los posibles casos de abuso. Como parte de las pruebas de lógica de negocio, se identificarán las reglas de negocio relevantes de la aplicación y se analizarán posibles maneras de comprometer su aplicación. Por ejemplo, en una aplicación de comercio de acciones, ¿se permite al atacante iniciar una operación al principio del día y bloquear un precio, mantener la transacción abierta hasta el final del día, y luego completar la venta si el precio de las acciones ha subido o cancelar si el precio ha bajado?

A la hora de definir los casos de abuso, un buen enfoque es construir dichos casos en torno a personas concretas con motivaciones y características bien definidas que tendrían interés en atacar la aplicación y obtener un beneficio de la explotación de debilidades potenciales. Las debilidades identificadas serán analizadas y se establecerán los correspondientes requisitos de seguridad para evitar los posibles ataques.

4.2.4.2.3 Pruebas Orientadas a Requisitos (RT3)

El objetivo de la práctica en este nivel de madurez es mantener el nivel de seguridad de la aplicación tras la corrección de errores, cambios evolutivos o durante el mantenimiento.

Línea de acción "Verificación de Controles" (Stream A): Automatizar las pruebas de requisitos de seguridad

La finalidad es lograr una detección rápida, eficaz y fiable de violaciones de los requisitos de seguridad.

Se desarrollarán y automatizarán pruebas de regresión de todos los defectos identificados y corregidos como medida de protección que garantice que no se introducen problemas similares en versiones posteriores de la aplicación. Estas

pruebas de seguridad verificarán dinámicamente, es decir, en tiempo de ejecución, que las nuevas versiones de la aplicación funcionan según lo esperado y que los cambios de código se han implementado correctamente.

Una buena práctica sería crear casos de prueba de seguridad como un conjunto de pruebas de seguridad genérico que forme parte del marco de pruebas unitarias global existente. El conjunto de pruebas de seguridad genérico puede incluir casos de prueba de seguridad para validar los requisitos positivos y negativos de los controles de seguridad esenciales, como por ejemplo, la identificación, autenticación y control de acceso, la validación, codificación y *escapado* de los datos de entrada, la gestión de usuarios y sesiones, la gestión de errores y excepciones, el cifrado y la auditoría y el registro de eventos.

Se verificará la correcta ejecución de las pruebas de seguridad lo antes posible. Si es factible, se considerará la superación de las pruebas de seguridad como un requisito previo al *merge* del nuevo código con el código del repositorio principal. Otra alternativa es considerar la superación de las pruebas de seguridad como un requisito previo al despliegue de la aplicación. Es decir, como requisito para validar la construcción.

Para las pruebas funcionales de seguridad, se deberían utilizar pruebas unitarias de la funcionalidad de los controles de seguridad, probando las funciones, métodos o clases de los componentes software involucrados.

Línea de acción "Pruebas de Mal Uso/Abuso" (Stream B): Realización de pruebas de estrés de seguridad

Mediante esta actividad se obtendrá información acerca de la resiliencia de las aplicaciones contra ataques de denegación de servicio (DoS).

Las aplicaciones son especialmente susceptibles a los ataques de denegación de servicio. Por ello, se llevarán a cabo pruebas de denegación de servicio y de estrés en condiciones controladas, preferiblemente en entornos de preproducción, QA o específicos para dicho fin.

Se utilizarán herramientas de pruebas de carga para generar tráfico sintético y verificar el rendimiento de la aplicación bajo una gran carga. Uno de los datos más importantes a obtener es el número de solicitudes por segundo que una aplicación puede manejar sin dejar de cumplir sus requisitos de rendimiento. Esto dará una idea de cuántas solicitudes concurrentes debe generar un atacante para afectar a la aplicación.

Uno de los efectos principales de los ataques de denegación de servicio es el agotamiento de los recursos de la aplicación. Para determinar la capacidad de la aplicación para hacer frente a un ataque de denegación de servicio, se deberán analizar los diferentes recursos de la aplicación (memoria, almacenamiento, etc.) con el fin de conocer cómo se pueden agotar.

A la hora de desarrollar pruebas de carga, se dará prioridad a aquellas funcionalidades o acciones que puede ser accedidas o realizadas por usuarios no autenticados, ya que este escenario es el que es más sencillo de explotar por parte de un atacante.

Los resultados obtenidos de las pruebas de carga deberán ser utilizadas como datos de entrada de los procesos de diseño del software.

4.2.4.3 Práctica de Pruebas de Seguridad (ST)

A diferencia de la práctica anterior de Pruebas Basadas en Requisitos (RT), que se centra en verificar que las aplicaciones implementan correctamente los requisitos de seguridad y la lógica de negocio, el objetivo de esta práctica es descubrir los puntos débiles técnicos y de la lógica de negocio de la aplicación, y hacerlos visibles a la dirección y a las partes interesadas de la empresa.

Esta práctica de Pruebas de Seguridad (ST) establece dos líneas de acción diferenciadas. La primera está orientada a fomentar y aprovechar el hecho de que las pruebas de seguridad automatizadas son rápidas de ejecutar y se pueden adaptar a un gran número de aplicaciones de la organización. La segunda línea de acción, se centra en la denominadas “pruebas en profundidad”, que requieren tener buenos conocimientos de cada una de las aplicaciones y su lógica de negocio, precisan de la participación de personal experto, son lentas y difíciles de automatizar.

En la siguiente figura, se muestran las actividades asociadas a los distintos niveles de madurez comprendidos en la práctica.

Maturity level		Stream A Scalable Baseline	Stream B Deep Understanding
1	Perform security testing (both manual and tool based) to discover security defects.	Utilize automated security testing tools	Perform manual security testing of high-risk components
2	Make security testing during development more complete and efficient through automation complemented with regular manual security penetration tests.	Employ application-specific security testing automation	Conduct manual penetration testing
3	Embed security testing as part of the development and deployment processes.	Integrate automated security testing into the build and deploy process	Integrate security testing into development process

Figura 18. Actividades de la práctica Pruebas de Seguridad. Fuente OWASP SAMM

Como se puede observar en la figura anterior, se definen las dos líneas acción para esta práctica: Línea de Base Escalable (Stream A) y Conocimiento Profundo de Aplicación (Stream B)

La primera línea de acción de la práctica promueve el establecimiento de una línea de base escalable de seguridad común para detectar automáticamente las debilidades más inmediatas o sencillas de detectar (lo que no quiere decir que sean las menos importantes o de menor severidad). Se irán personalizando y aumentando progresivamente las pruebas automatizadas para cada aplicación y se incrementará su frecuencia de ejecución. El objetivo es detectar cada vez más errores y problemas de regresión y, además, descubrirlos lo antes posible dentro del proceso de desarrollo del software. La idea es que cuantos más errores se puedan detectar de forma automática, más tiempo tendrán los expertos para utilizar sus conocimientos y creatividad en descubrir vectores de ataque más complejos y dedicar esfuerzos en las pruebas en profundidad de la segunda línea de acción de esta práctica.

La segunda línea de acción, como se ha comentado, se centra en realizar pruebas en profundidad de las aplicaciones, principalmente manuales, utilizando vectores de ataque complejos. El objetivo último de esta segunda línea de acción es convertir las pruebas de seguridad avanzadas en una parte integral del proceso de desarrollo. Como estas revisiones son lentas y difíciles de escalar,

se deberá priorizar las pruebas sobre los diversos componentes en función de su riesgo, cambios relevantes recientes o próximas versiones importantes.

4.2.4.3.1 Pruebas de Seguridad (ST1)

El objetivo de la práctica en este nivel inicial de madurez es realizar pruebas de seguridad, tanto manuales como automatizadas, para descubrir defectos de seguridad.

Línea de acción "Línea de Base Escalable" (Stream A): Pruebas de seguridad automatizadas

El objetivo de la actividad es encontrar vulnerabilidades comunes fáciles de detectar.

Las pruebas de seguridad de las aplicaciones pueden llevarse a cabo mediante la inspección del código fuente de una aplicación sin ejecutarla (análisis estático), u observando el comportamiento de la aplicación en respuesta a diversas condiciones de entrada (análisis dinámico). El primer enfoque suele denominarse Prueba de Seguridad de Aplicaciones Estática (SAST), y el segundo, Prueba de Seguridad de Aplicaciones Dinámica (DAST). Existe también un enfoque híbrido, conocido como Pruebas Interactivas de Seguridad de Aplicaciones (IAST), que combina los puntos fuertes de ambos enfoques (a costa de una sobrecarga adicional), pues se prueban de forma dinámica y automática las aplicaciones instrumentadas. Esto permite un seguimiento preciso del estado interno de la aplicación en respuesta a la entrada externa.

Muchas vulnerabilidades de seguridad son muy difíciles de detectar sin inspeccionar cuidadosamente el código fuente. Aunque lo ideal es que esto se realice mediante una revisión por expertos o por pares, es una tarea lenta y costosa.

La herramientas SAST automáticas, aunque son "ruidosas" (arrojan un gran número de falsos positivos) y a menudo menos precisas que las revisiones dirigidas por expertos, son más baratas, mucho más rápidas y más consistentes que las humanas.

Por su parte, las pruebas dinámicas no requieren disponer del código fuente de la aplicación, por lo que son ideales para aquellos casos en que el código fuente no está disponible. También permiten identificar casos concretos de vulnerabilidades. Debido a su enfoque de "caja negra", sin instrumentación, las herramientas DAST suelen tener la limitación de descubrir únicamente fallos en la superficie de las aplicaciones.

Para dar cumplimiento a esta práctica de SAMM, se utilizarán herramientas automatizadas de pruebas de seguridad estáticas y dinámicas (SAST y DAST). Esto permitirá realizar las pruebas de seguridad de forma más eficiente y obtener resultados más consistentes y de mayor calidad. Aprovechando esta automatización, se incrementará progresivamente la frecuencia de las pruebas y se trabajará en ampliar la cobertura del código.

La organización seleccionará las herramientas adecuadas en función de varios factores, como la profundidad y la precisión de la inspección, la solidez y la precisión de los casos de prueba de seguridad, las integraciones disponibles con otras herramientas, el modelo de uso, el coste, etc. Para seleccionar dichas

herramientas la organización debería contar con la opinión del personal técnico experto en seguridad, desarrolladores y responsables de desarrollo.

Línea de acción "Conocimiento Profundo de Aplicación" (Stream B): Probar manualmente los componentes de alto riesgo

La finalidad de esta actividad es la detección manual de vulnerabilidades en los componentes críticos de las aplicaciones.

Se realizarán pruebas de seguridad manuales selectivas (se pueden emplear previamente una combinación de herramientas de análisis estático y dinámico para guiar o enfocar la revisión manual), con el fin de analizar más a fondo ciertas partes de la aplicación, procediendo con lo haría un atacante. Las herramientas automatizadas son eficaces para encontrar varios tipos de vulnerabilidades, pero nunca pueden sustituir a la revisión manual experta.

Las vulnerabilidades a nivel de código en partes de software críticas para la seguridad pueden tener un impacto dramáticamente mayor, por lo que los equipos de proyecto revisan los módulos de alto riesgo en busca de vulnerabilidades comunes. Ejemplos comunes de funcionalidad de alto riesgo son los módulos de autenticación, los puntos de control de acceso, la gestión de sesiones, interfaces externas y validación de datos de entrada. Los equipos pueden combinar métricas a nivel de código y escaneos automatizados enfocados para determinar dónde es mejor enfocar sus esfuerzos. En la práctica, la actividad puede adoptar muchas formas, como la revisión por pares, revisiones independientes espontáneas de los miembros de un grupo de seguridad especializado, etc.

Los hallazgos resultantes de las pruebas de seguridad deberán registrarse como defectos siguiendo lo establecido por la Práctica "Gestión de Defectos" (DM).

4.2.4.3.2 Pruebas de Seguridad (ST2)

El objetivo de esta actividad es hacer que las pruebas de seguridad, durante las fases de desarrollo del software, sean más completas y eficientes, tanto mediante técnicas de automatización como complementándolas con tests de penetración manuales regulares.

Línea de acción "Línea de Base Escalable" (Stream A): Desarrollar pruebas de seguridad específicas de la aplicación

El objetivo de esta actividad es encontrar las vulnerabilidades que se dan de forma típica en las aplicaciones de la organización y que son fácilmente detectables.

Se trata de dar un paso más para aumentar la eficacia de las herramientas automáticas que se emplean en las pruebas de seguridad. Para ello, se configurarán y personalizarán las herramientas de pruebas de seguridad automatizadas adaptándolas a la naturaleza de las aplicaciones y *stacks* tecnológicos de la organización.

Las herramientas de pruebas de seguridad automatizadas tienen dos características importantes a tener en cuenta:

- La tasa de falsos positivos, es decir, los errores y vulnerabilidades inexistentes que se reportan incorrectamente
- La tasa de falsos negativos, es decir, los errores y vulnerabilidades reales que no detectan.

A medida que se madura en el uso de herramientas de pruebas automatizadas, se deberán intentar minimizar las tasas de falsos positivos y falsos negativos. Esto maximizará el tiempo que los equipos de desarrollo tienen que dedicar a revisar y abordar los problemas reales de seguridad en sus aplicaciones, y reduce la fricción asociada con el uso de herramientas de análisis de seguridad automatizadas.

Para evitar el "ruido" de las herramientas automáticas, se desactivará la compatibilidad de la herramienta con tecnologías y *frameworks* que no se utilizan, y se configurará la herramienta para las versiones específicas de las tecnologías empleadas siempre que sea posible. Esto aumentará la velocidad de ejecución y la fiabilidad de los resultados generados.

Muchas de las herramientas de análisis estático automatizado permiten a los equipos escribir sus propias reglas o personalizar las reglas de análisis por defecto para mejorar la precisión y la profundidad de la cobertura de código. Por ejemplo, entradas de datos potencialmente peligrosas (por ejemplo, no validadas) puede marcarse como seguras después de que pasen por un método de desinfección y de esta forma se evitarán falsos positivos en los resultados.

La organización se deberá apoyar en los "security champions" o en equipos de seguridad compartidos para realizar pruebas de concepto o pilotos con las herramientas de pruebas de seguridad e identificar los falsos positivos que deben ignorarse o eliminarse de los resultados. También se podrán identificarán falsos negativos y *antipatrones* que se suelen dar en los diseños de las aplicaciones de la organización. Una vez que las herramientas hayan sido suficientemente ajustadas, podrán ponerse a disposición de un mayor número de equipos de desarrollo.

Se considerará establecer una estrategia incremental en el uso de las herramientas automáticas. Es mucho más conveniente detectar de forma fiable un subconjunto de problemas de seguridad y ampliar la cobertura de forma gradual, que intentar detectar todos los problemas conocidos inmediatamente.

Línea de acción "Conocimiento Profundo de Aplicación" (Stream B): Establecer un proceso de pruebas de penetración

La finalidad de esta actividad es obtener información acerca del grado de resistencia de las aplicaciones antes ataques, desde una perspectiva de "caja negra".

Se realizarán pruebas de penetración manuales a partir del conjunto de casos de prueba de seguridad identificados para cada proyecto y se observará el comportamiento del sistema o aplicación para cada caso. Normalmente, esto se realizará durante la fase de pruebas previa a la publicación de la aplicación.

En los casos en los que el software no puede probarse de forma realista fuera del entorno de producción, se valorará el uso de técnicas como los despliegues

blue-green[5] para permitir la ejecución de las pruebas de seguridad en el entorno de producción.

Las pruebas de penetración incluyen, tanto pruebas específicas para comprobar la solidez de la lógica de negocio, como pruebas de vulnerabilidad comunes para comprobar el diseño y la implementación de la aplicación. Una vez definidas las pruebas a realizar, el personal responsable de verificar la calidad o el personal de desarrollo con conocimientos de seguridad procederá a ejecutar las pruebas. El equipo de seguridad informática de la organización supervisará las primeras ejecuciones de los casos de prueba de seguridad de un proyecto para ayudar y entrenar a los “security champions” del equipo de proyecto.

Otra alternativa utilizada por algunas organizaciones es ofrecer programas "Bug Bunty”, que invitan a analistas de seguridad a encontrar vulnerabilidades en las aplicaciones y a informar de ellas responsablemente a cambio de recompensas.

Antes de la publicación o el despliegue efectivo de las aplicaciones, las partes interesadas revisan los resultados de las pruebas de seguridad y, en su caso, aceptan los riesgos asociados a las pruebas fallidas en el momento de la publicación. En dicho caso, se establecerá un calendario concreto para subsanar las deficiencias a lo largo del tiempo.

Se deberá comunicar a los equipos de desarrollo la ejecución de las pruebas de penetración y se difundirán sus resultados para mejorar el conocimiento y la conciencia de la seguridad dentro de la organización.

4.2.4.3.3 Pruebas de Seguridad (ST3)

El objetivo de la práctica en este nivel máximo de madurez es incorporar las pruebas de seguridad como parte de los procesos de desarrollo, despliegue e implantación de las aplicaciones.

Línea de acción “Línea de Base Escalable” (Stream A): Integrar las herramientas de pruebas en la cadena de suministro (pipeline)

Esta acción tiene como objetivo la identificación de las vulnerabilidades detectables de forma automáticamente en las fases más tempranas posibles del ciclo de vida del software.

Para ello, se integrarán las pruebas de seguridad automatizadas en la cadena de producción del software (pipeline). Es decir, se configurarán las herramientas de pruebas para que se ejecuten automáticamente como parte del proceso de construcción y despliegue y se inspeccionarán los resultados de las pruebas según sean obtenidos. Con esto se consigue el que sistema global de producción de software seguro sea escalable a toda la organización con poco esfuerzo.

Realizar pruebas de seguridad en fases tempranas es beneficioso. Se considerará adoptar un enfoque del desarrollo orientado a las pruebas (test-

[5] https://martinfowler.com/bliki/BlueGreenDeployment.html

driven development o TDD[6]), lo que implica la identificación y ejecución de casos de pruebas de seguridad relevantes en las primeras fases del ciclo de desarrollo (análisis de requisitos y diseño). Así, los proyectos llegan a la fase de implementación con un conjunto de pruebas fallidas para funcionalidad todavía inexistente y será el trabajo de los desarrolladores implementar la funcionalidad requerida para que dichas pruebas sean pasadas. La implementación se completa cuando se superan todas las pruebas. En general, la metodología TDD proporciona una visión clara y anticipada a los desarrolladores, lo que contribuye a reducir el riesgo de retrasos debido a problemas de seguridad o a evitar la aceptación forzosa del riesgo por parte de la organización para cumplir los plazos de entrega.

Independientemente de la metodología aplicada, los resultados de las pruebas de seguridad automatizadas (y manuales) integradas en el proceso de construcción deberán ser accesibles a través de cuadros de mando o informes, y se presentarán de forma rutinaria a la dirección y a las partes interesadas del negocio (por ejemplo, antes de cada lanzamiento) para su revisión. Si existieran hallazgos no resueltos cuyo riesgo es aceptado por la organización, las partes interesadas y los responsables de desarrollo establecerán un calendario concreto para abordarlos.

El conocimiento adquirido a partir de la ejecución de las pruebas de seguridad creadas y los resultados de las mismas se deberá difundir entre los equipos de desarrollo para mejorar los conocimientos y la concienciación en materia de seguridad del software dentro de la organización.

Línea de acción "Conocimiento Profundo de Aplicación" (Stream B): Verificación de seguridad continua y escalable

Esta actividad tiene como objetivo principal la identificación de problemas de seguridad que solo son detectables mediante pruebas manuales, lo antes posible en el proceso de ciclo de vida del software.

Para lograr el objetivo de esta actividad, se integrarán las pruebas de seguridad en paralelo a todas las demás actividades de desarrollo, incluidos el análisis de requisitos, el diseño y la construcción del software.

Dada la gran cantidad de herramientas de seguridad que se emplean en las diversas fases del desarrollo, ya no tiene sentido esperar a solucionar los problemas de seguridad en una fase final del proceso, como por ejemplo las pruebas previas al despliegue o a la publicación. Los problemas de seguridad deben tratarse y clasificarse lo antes posible y la corrección de defectos debe ser planificada en función del riesgo y esfuerzo de reparación.

Se mejorará de forma proactiva la integración de las pruebas de seguridad en el proceso de desarrollo, propagando adecuadamente los resultados de otras actividades de pruebas de seguridad como pueden ser las pruebas manuales de penetración. Por ejemplo, si una prueba de penetración de seguridad identifica

[6] TDD es una práctica de programación que consiste en escribir primero las pruebas, después escribir el código que hace que la prueba pase satisfactoriamente y, finalmente, se refactoriza el código escrito.

problemas con la gestión de sesiones, cualquier cambio en la gestión de sesiones debería desencadenar pruebas de seguridad explícitas antes de pasar los cambios a producción.

Los “security champions” y los expertos en desarrollo de software seguro de la organización revisarán de forma continua los resultados de las pruebas de seguridad automatizadas y manuales que se llevan a cabo durante el desarrollo. Los resultados de dichas pruebas se incluirán como parte de las actividades formativas en materia de concienciación sobre seguridad para los equipos de desarrollo. Adicionalmente, como parte del desarrollo evolutivo y mejora continua de la la organización, las lecciones aprendidas en el proceso se incorporarán a las guías y manuales generales para mejorar el desempeño de las pruebas de seguridad.

En el caso de que existan hallazgos no resueltos que permanecen como riesgos aceptados para la versión, las partes interesadas y los directores de desarrollo deben trabajar juntos para establecer un calendario concreto para resolverlos.

Las partes interesadas revisan los resultados de las pruebas y los manejan de acuerdo con la gestión de riesgos de la organización

4.2.5 Función de negocio Operación

Esta función de negocio de SAMM comprende las actividades necesarias para garantizar que la confidencialidad, la integridad y la disponibilidad se mantengan durante toda la vida operativa de una aplicación y sus datos asociados. Una mayor madurez en las prácticas de esta función de negocio, proporciona una mayor garantía de la resiliencia de la organización frente a las interrupciones operativas y una mayor capacidad de respuesta ante cambios en los escenarios de operación.

Esta función se desarrolla mediante tres prácticas:

- Gestión de Incidentes: actividades llevadas a cabo para mejorar la detección y la respuesta de la organización a los incidentes de seguridad.
- Gestión del entorno: actividades proactivas llevadas a cabo para mejorar y mantener la seguridad de los entornos en los que operan las aplicaciones de la organización.
- Gestión operativa: actividades de apoyo operativo necesarias para mantener la seguridad durante todo el ciclo de vida del producto.

A continuación se describirá cada una de las tres prácticas mencionadas.

4.2.5.1 Práctica Gestión de Incidentes (IM)

Una vez que la organización tiene aplicaciones en funcionamiento, es probable que se enfrente a incidentes de seguridad[7]. En SAMM, se define un incidente de seguridad como una brecha, o la amenaza de una brecha inminente, sobre alguna de las dimensiones u objetivos de seguridad de, al menos, uno de los activos. Algunos ejemplos de incidentes de seguridad pueden ser

[7] Como se suele decir: La cuestión no es si voy a tener incidentes o no, sino cúando.

- un ataque exitoso de denegación de servicio (DoS) contra una aplicación en la nube,
- que un usuario de una aplicación acceda a datos privados de otro abusando de una vulnerabilidad de seguridad, o
- un atacante que modifique el código fuente de la aplicación.

La práctica de la gestión de incidentes (IM) se centra en el tratamiento de estos en su organización.

Históricamente, muchos incidentes de seguridad se han detectado meses, o incluso años, después de la violación inicial. Durante el "tiempo de permanencia" antes de que se detecte un incidente, pueden producirse daños importantes, lo que aumenta la dificultad de la recuperación. La primera de las dos líneas de acción de esta práctica, la detección de incidentes, se centra en reducir ese tiempo de espera.

Una vez que haya identificado que está sufriendo un incidente de seguridad, es esencial responder de manera disciplinada y minuciosa para limitar el daño y volver al estado de normalidad de la manera más eficiente posible. La segunda línea de acción de esta práctica se centra precisamente en esto.

En la siguiente figura, se muestran las actividades asociadas a los distintos niveles de madurez comprendidos en la práctica.

Maturity level		Stream A Incident Detection	Stream B Incident Response
1	Best-effort incident detection and handling	Use available log data to perform best-effort detection of possible security incidents.	Identify roles and responsibilities for incident response.
2	Formal incident management process in place	Follow an established, well-documented process for incident detection, with emphasis on automated log evaluation.	Establish a formal incident response process and ensure staff are properly trained in performing their roles.
3	Mature incident management	Use a proactively managed process for detection of incidents.	Employ a dedicated, well-trained incident response team.

Figura 19. Actividades de la práctica de Gestión de Incidentes. Fuente OWASP SAMM

Como se puede observar en la figura anterior, se definen las dos líneas acción para esta práctica: Detección de Incidentes (Stream A) y Respuesta ante Incidentes (Stream B).

La primera línea de acción se enfoca en la capacidad de la organización para detectar incidentes de seguridad e iniciar las actividades de respuesta apropiadas.

La segunda línea se centra en la capacidad de respuesta de la organización. La respuesta ante incidentes comienza en el momento en que se reconoce y confirma la existencia de un incidente de seguridad. El objetivo es actuar de forma coordinada y eficiente para limitar al máximo los daños. en la medida de lo posible. Las actividades de esta corriente se centran en la capacidad de la organización para responder de forma apropiada y eficaz a los incidentes de seguridad reportados.

A continuación, se describirá en detalle cada una de las seis actividades (dos por cada nivel de madurez) comprendidas en la práctica.

4.2.5.1.1 Gestión de Incidentes (IM1)

El objetivo de la práctica de en el nivel de madurez inicial es llevar a cabo actividades para la detección y el tratamiento de incidentes en la medida de las posibilidades de la organización.

Línea de acción "Gestión de incidentes" (Stream A): Detección básica de incidentes.

La finalidad de esta línea de acción es dotar a la organización de cierta capacidad para detectar los incidentes de seguridad más básicos.

Se analizarán la información de log disponibles, como los logs de acceso, eventos de las aplicaciones, logs de infraestructura, etc., para detectar posibles incidentes de seguridad. La frecuencia con la que se realizan los análisis de la información será proporcional a la criticidad de las aplicaciones.

En instalaciones y servicios sencillos o de pequeño tamaño, el análisis se puede llevar a cabo mediante las herramientas comunes de línea de comandos. Con volúmenes de logs más grandes se deberá recurrir a técnicas de automatización. Incluso un cron que ejecute un simple script que busque eventos sospechosos ya es un paso adelante.

En caso de que se disponga de un sistema centralizado que permita la agregación de la información de logs, el análisis de la información se llevará a cabo en dicho sistema y se deberían emplear los principios básicos de correlación de registros.

Se deberá asegurar que una posible indisponibilidad del responsable de la detección de incidentes no afecte significativamente a capacidad de detección de la organización.

La organización definirá un punto de contacto común, responsable del proceso de creación de aquellos incidentes de seguridad que sean detectados.

Línea de acción "Respuesta ante Incidentes" (Stream B): Crear un plan de respuesta a incidentes.

El objeto es que la organización se dote de cierta capacidad para resolver eficazmente los incidentes de seguridad más comunes.

En primer lugar, la organización deberá reconocer la competencia de respuesta a incidentes y definir un responsable del proceso. El personal involucrado en la gestión de incidentes de seguridad deberá disponer del tiempo y recursos necesarios para mantenerse mínimamente al día sobre el estado actual de las mejores prácticas de gestión de incidentes y las herramientas forenses.

En este nivel de madurez inicial, es posible que no se establezca un equipo dedicado a la respuesta ante incidentes, pero sí será preciso definir claramente a los participantes del proceso. Se asignará un punto de contacto único que deberá ser conocido por todas las partes interesadas. Se deberá asegurar además que dicho punto de contacto conoce cómo ponerse en contacto con cada uno de los participantes en el proceso. En caso de que sea necesario, también se definirán las responsabilidades y calendarios de guardia.

En el momento que se produzca algún incidente de seguridad, se deberán documentar debidamente todas las decisiones y medidas adoptadas para la gestión del incidente. Esta información, dada su sensibilidad, se deberá proteger adecuadamente de accesos no autorizados.

4.2.5.1.2 Gestion de Incidentes (IM2)

El objetivo de la práctica en este nivel de madurez es establecer en la organización un proceso formal de gestión de incidentes de seguridad.

Línea de acción "Gestión de incidentes" (Stream A): Definir un proceso de detección de incidentes

Esta actividad tiene como objeto que la organización sea capaz de llevar a cabo una detección oportuna y coherente de incidentes de seguridad esperados o previstos.

La organización nombrará formalmente al propietario del proceso de detección de incidentes, documentará el proceso, habilitará que la documentación sea clara y accesible para todas las partes interesadas, y garantizará su revisión y actualización periódica según sea necesario. Además, se asegurará que el personal responsable de la detección de incidentes conoce y actúa conforme al proceso definido (por ejemplo, mediante talleres formativos).

El proceso de detección de incidentes suele apoyarse en un alto grado de automatización, recogiendo y correlacionando datos de registros de diferentes fuentes, incluidos los logs de las aplicaciones. Los registros se pueden recopilar en un sistema centralizado para facilitar la revisión y análisis de los mismos.

Se verificará periódicamente la integridad de los datos de registro recopilados y, en caso de que se añadan nuevas aplicaciones, se asegurará que el proceso de recopilación cubre la información asociada en un periodo de tiempo razonable.

Se desarrollará una lista de comprobación que cubra los vectores de ataque previstos y su correspondiente "intrusion kill chain"[8] conocida o esperada. Esta lista se evaluará y actualizará regularmente. El proceso de detección utilizará la lista de comprobación establecida para identificar posibles incidentes de seguridad.

En el momento que se determine con un cierto grado de confianza que un evento analizado se trata efectivamente de un incidente de seguridad, este se notificará inmediatamente al personal responsable, incluso fuera del horario de trabajo. En caso necesario, se realizará un análisis más profundo del evento y se iniciará el proceso de escalado para su gestión.

Línea de acción "Respuesta ante Incidentes" (Stream B): Definir un proceso de respuesta a incidentes

[8] La Intrusion Kill Chain es una secuencia de siete pasos que caracterizan las diferentes etapas de un ataque. Esta cadena facilita a la potencial víctima el proceso de identificar cada fase del ataque y determinar si las medidas de protección son adecuadas para mitigar el ataque. Fuente: INCIBE (https://www.incibe-cert.es/blog/cyber-kill-chain-sistemas-control-industrial)

Esta actividad tiene como finalidad la comprensión y gestión eficiente de la mayoría de los incidentes de seguridad.

Se establecerá formalmente y se documentará el proceso de respuesta a incidentes de seguridad. La documentación deberá incluir información del tipo:

- Escenarios más probables/comunes de incidentes de seguridad e instrucciones de alto nivel para su gestión. Para documentar estos escenarios se utilizará también información de dominio público sobre incidentes de terceros relevantes.
- Reglas de clasificación de cada incidente
- Reglas sobre la participación de las diferentes partes involucradas, como la alta dirección, relaciones públicas, departamento jurídico, delegado de protección de datos, recursos humanos, autoridades (Policía, Agencia de Protección de Datos, CSIRT nacional, etc.) y clientes. En caso de que existan plazos de cumplimiento obligatorio establecidos por la normativa de aplicación, se especificarán dichas restricciones temporales.
- El proceso para realizar el análisis de la causa del incidente y la documentación de sus resultados

Se garantizará la disponibilidad de un equipo de respuesta a incidentes, así como una adecuada formación del mismo, tanto en horario laboral como fuera del mismo.

Se establecerán claramente los plazos de actuación ante la aparición de un incidente de seguridad, así como la composición y ubicación del centro de operaciones o sala de crisis desde donde se gestionarán los incidentes de seguridad.

La organización se ocupará de que las herramientas hardware y software necesarias para la gestión de incidentes estén debidamente actualizadas y listas para su uso en cualquier momento.

4.2.5.1.3 Gestion de Incidentes (IM3)

El objeto de la práctica en este nivel de madurez es la mejora e incremento de la madurez del proceso de gestión de incidentes de seguridad.

Línea de acción "Gestión de incidentes" (Stream A): Mejorar el proceso de detección de incidentes

La finalidad de esta actividad es mejorar el proceso de detección de incidentes ya implantado.

Se garantizará que en la definición del proceso de gestión de incidentes se incluyan medidas para la mejora continua del mismo. Por ejemplo estableciendo un proceso de tipo PDCA (Ciclo de Demming) que revise y actualice anualmente el proceso. Se verificará que se ejecuta el ciclo de mejora continua (por ejemplo, haciendo seguimiento de cambios en el proceso).

Se asegurará que la lista de comprobación para la detección de actividad sospechosa se actualice debidamente y esté correlacionada, al menos, con:

- Fuentes y bases de conocimiento externas a la empresa. Por ejemplo, anuncios de nuevas vulnerabilidades que afecten a las tecnologías utilizadas
- Incidentes de seguridad sufridos anteriormente
- Resultados de modelos de amenazas.

Utilizar la correlación de los registros para la detección de incidentes para todos los escenarios de incidentes razonables. Si los datos de los registros para la detección de incidentes no están disponibles, se tratará su ausencia como un defecto y se gestionará conforme a lo establecido en el proceso de Práctica "Gestión de Defectos" (DM).

La velocidad con la que se detectan los incidentes y la calidad de la detección no debe depender de a hora o día en que se producen los eventos. Si los eventos de seguridad no se reconocen y resuelven en el tiempo establecido (por ejemplo, 30 minutos), se generarán las notificaciones correspondientes para seguir la ruta de escalado establecida.

Línea de acción "Respuesta ante Incidentes" (Stream B): Establecer un equipo de respuesta a incidentes

El objetivo de esta actividad es lograr una respuesta eficiente a los incidentes, independientemente del momento en el que ocurran, la ubicación o el tipo de incidente que se haya producido.

Se establecerá un equipo de respuesta a incidentes dedicado, disponible continuamente y responsable de la mejora continua del proceso a través de Análisis de Causa Raíz (RCA) regulares. En el caso de las organizaciones distribuidas, defina y documente las normas de actuación, responsabilidades y logística para todas las ubicaciones relevantes.

Se documentarán exhaustivamente y se actualizarán (al menos una vez al año) los procedimientos detallados de respuesta a incidentes y se automatizarán en la medida que sea posible.

Se garantizará que están listos todos los recursos necesarios para la ejecución de los procedimientos, como por ejemplo, infraestructuras de comunicación independientes, ubicación externa fiable, etc. Se detectará y corregirá a tiempo la no disponibilidad de cualquiera de estos recursos.

Se llevarán a cabo ejercicios o simulacros de incidentes y emergencias para garantizar que todo está debidamente preparado y los recursos entrenados para el momento en que surja un incidente de seguridad. Se analizará el resultado de estos ejercicios y se realizarán las correcciones y mejoras oportunas.

Se definirán, recopilarán y evaluarán métricas del proceso de respuesta ante incidentes y, sobre la base de la información obtenida, se llevarán a cabo las actuaciones correspondientes para la mejora del proceso.

4.2.5.2 Práctica Gestión del Entorno (EM)

El trabajo de la organización en materia de seguridad de las aplicaciones no termina una vez que estas entran en funcionamiento. De forma periódicamente

se publican nuevas funciones y parches de seguridad para los distintos elementos del *stack* tecnológico empleado por la organización, hasta que finalmente estos dejan de recibir soporte o quedan obsoletos.

Por defecto, la mayoría de las tecnologías de cualquier *stack* de aplicaciones no son seguras por defecto. Por ello, es necesario realizar una configuración específica de los diferentes elementos (*hardening*) para garantizar el funcionamiento seguro de los sistemas. Esta práctica de Gestión del Entorno (EM) se enfoca precisamente en esto, en mantener el entorno limpio y seguro.

Las vulnerabilidades se descubren a lo largo del ciclo de vida de las diversas tecnologías empleadas por la organización, y las nuevas versiones que eliminan dichas vulnerabilidades se publican según el calendario establecido por cada uno de los fabricantes. Esto hace que sea esencial supervisar los informes de vulnerabilidades y realizar una aplicación de parches ordenada y coherente que garantice la compatibilidad de todos los sistemas afectados.

En la siguiente figura, se muestran las actividades asociadas a los distintos niveles de madurez comprendidos en la práctica.

Maturity level		Stream A Configuration Hardening	Stream B Patching and Updating
1	Best-effort patching and hardening	Perform best-effort hardening of configurations, based on readily available information.	Perform best-effort patching of system and application components.
2	Formal process with baselines in place	Perform consistent hardening of configurations, following established baselines and guidance.	Perform regular patching of system and application components, across the full stack. Ensure timely delivery of patches to customers.
3	Conformity with continuously improving process enforced	Actively monitor configurations for non-conformance to baselines, and handle detected occurrences as security defects.	Actively monitor update status and manage missing patches as security defects. Proactively obtain vulnerability and update information for components.

Figura 20. Actividades de la práctica Gestión del Entorno. Fuente OWASP SAMM

Como se puede observar en la figura anterior, se definen las dos líneas acción para esta práctica: *Hardening* de Configuración (Stream A) y Parcheado y Actualización (Stream B).

La primera línea de acción se centra en la gestión de la configuración de seguridad de todos los elementos del *stack* tecnológico. Se hace especial hincapié sobre aquellos elementos de terceros, como sistemas operativos, contenedores, *frameworks*, servicios, librerías, *appliances*, etc., puesto que su arquitectura, diseño e implementación no están bajo el control de la organización.

La segunda línea ocupa de la gestión de los parches y las actualizaciones de todos los elementos del *stack*. En el caso del software desarrollado por la organización para sus clientes, estas actividades hacen referencia a la entrega de parches y actualizaciones a los clientes. En el caso de elementos de terceros, estas actividades se refieren a la aplicación oportuna de las actualizaciones y parches recibidos.

A continuación, se describirá en detalle cada una de las seis actividades (dos por cada nivel de madurez) comprendidas en la práctica.

4.2.5.2.1 Gestión del Entorno (EM1)

El objetivo de la práctica en el nivel de madurez inicial es llevar a cabo actividades básicas de gestión del entorno operativo.

Línea de acción Hardening de Configuración (Stream A): Hardening básico de la configuración

El objetivo de esta actividad es realizar labores de *hardening* básico en la medida de las posibilidades de la organización.

Se aplicará una configuración básica de seguridad sobre los diferentes elementos de la pila basándose en guías públicamente disponibles tales como las publicadas por los proyectos *open source*, documentación de proveedores, artículos de blogs, etc. En el momento que los equipos técnicos de la organización desarrollen las guías de configuración para sus aplicaciones, basándose en métodos de prueba y error y/o en información recopilada por los miembros del equipo, se fomentará que dicha información y aprendizajes sean compartidos con toda la organización.

Se identificarán los elementos clave de los *stacks* tecnológicos comunes y se establecerán unas mínimas normas o estándares de configuración para cada uno de ellos. En este nivel de madurez, todavía no existirá un proceso formal para gestionar las líneas de base de la configuración y es probable que las configuraciones no se apliquen de forma coherente en todos los casos ni se supervise la conformidad.

Línea de acción Parcheado y Actualización (Stream B): Parcheado básico de componentes.

El objetivo es trabajar en la aplicación básica de parches y actualizaciones en la medida de las posibilidades de la organización.

Se identificarán las aplicaciones y los componentes de terceros que necesitan ser actualizados o parcheados, incluidos sistemas operativos, los servidores de aplicaciones y librerías de código de terceros.

En este nivel de madurez, las actividades de identificación y aplicación de parches se llevarán a cabo de forma *ad hoc* y en la medida de las posibilidades de la organización, sin un proceso gestionado y formalizado para el seguimiento de versiones, actualizaciones y estado de los parches. No obstante deberían existir ciertas directrices de alto nivel como, por ejemplo, probar los parches antes de pasarlos a producción, asegurar la compatibilidad entre los componentes. Los equipos de producto harán lo máximo para cumplir con dichos requerimientos.

Salvo en el caso de actualizaciones de seguridad críticas, se aprovecharán para realizar parcheados las ventanas de mantenimiento establecidas para otros fines. En el caso del software desarrollado por la organización, los parches se entregarán a los clientes o se aplicarán a los servicios proporcionados por la organización aprovechando el lanzamiento de nuevas versiones con nuevas funcionalidades.

Se garantizará que los equipos son capaces de identificar las versiones de todos los componentes en uso, para evaluar si sus productos están afectados por una

vulnerabilidad de seguridad publicada o notificada. Es decir, existe una lista actualizada de los componentes de producción que incluye información sobre el versionado y, además, se revisan regularmente las fuentes públicas para detectar vulnerabilidades que afecten a los componentes utilizados.

4.2.5.2.2 Gestión del Entorno (EM2)

El objetivo de la práctica en este nivel de madurez es el establecimiento formal de una línea base de seguridad para la gestión del entorno de operación.

Línea de acción Hardening de Configuración (Stream A): Establecer línea de base de hardening de la configuración

La finalidad es aplicar un proceso de *hardening* consistente en todos los componentes del *stack* tecnológico de la organización.

Se establecerán las líneas base para el *hardening* de configuración para todos los componentes de cada uno de los *stacks* tecnológicos empleado en la organización. Para favorecer la aplicación de estas líneas base se desarrollarán guías de configuración para los diversos componentes. Se exigirá a los equipos técnicos que apliquen dichas líneas base a todos los sistemas nuevos, y, en la medida de lo posible, a los sistemas existentes.

Se realizará gestión de cambios para las líneas base y guías de configuración y se asignará un responsable o propietario a cada una de ellas. Estos responsables tendrán la obligación de mantener la documentación actualizada de forma continua, basándose en la evolución de las mejores prácticas, nuevas instrucciones de fabricante o en los cambios de los componentes relevantes.

En los entornos más grandes o complejos, se centralizará la configuración de los componentes conforme a la línea base establecida y se emplearán herramientas automatizadas para aplicar el *hardening* en los diversos sistemas o servidores de la organización.

Línea de acción Parcheado y Actualización (Stream B): Formalizar proceso de parcheado.

Esta actividad está enfocada en conseguir la aplicación proactiva y consistente de los parches de seguridad.

Se desarrollará, formalizará y se seguirá un proceso bien definido para gestionar los parches de los componentes de las aplicaciones. Los procesos de actualización establecerán calendarios regulares para aplicar las actualizaciones de software, alineados con los calendarios de actualización de los proveedores (como el *Microsoft Patch Tuesday*[9]). Para el software desarrollado por la organización, se entregarán las versiones a los clientes o se aplicarán los parches a los servicios proporcionados de forma regular (por ejemplo, mensualmente), independientemente de que la actualización incluya nuevas funcionalidades.

[9] El 'Patch Tuesday' es una práctica de los grandes fabricantes de la industria del software que consiste en agrupar sus parches para liberarlos el segundo martes de cada mes.

Se elaborará una guía para priorizar los parches de los componentes en función de su criticidad. Se tendrán en cuenta los factores como la criticidad de la aplicación y/o la gravedad de las vulnerabilidades abordadas a la hora de establecer las prioridades para la aplicación de parches. Esta guía reflejará la tolerancia al riesgo de la organización y sus objetivos de gestión.

En el caso de recibir una notificación de una vulnerabilidad crítica en un componente, mientras no esté disponible el parche, se clasificará y gestionará la situación como un problema de gestión de riesgos. Esto dará lugar, por ejemplo, a la implementación de controles compensatorios, aceptación del riesgo o desactivación de las aplicaciones o funciones afectadas.

4.2.5.2.3 Gestión del Entorno (EM3)

El objetivo de este nivel de práctica es establecer un proceso de mejora continua en la gestión del entorno operativo.

Línea de acción Hardening de Configuración (Stream A): Seguimiento continuo de la configuración.

El objetivo perseguido por esta actividad es que la organización disponga de una visión clara y precisa de las configuraciones de los componentes con el fin de evitar incumplimientos sobre lo establecido por las líneas base de configuración segura.

Se supervisará de forma proactiva la configuración de seguridad del *stack* tecnológico empleado por la organización, realizando comprobaciones periódicas que garanticen el cumplimiento de las líneas base establecidas. Los resultados de las comprobaciones se almacenarán y estarán disponibles mediante la publicación de informes y cuadros de mando de cumplimiento.

Cuando se detecten configuraciones no conformes se creará la correspondiente incidencia. Esta se tratará como un hallazgo de seguridad y se gestionarán las acciones correctivas pertinentes conforme a lo establecido por la Práctica “Gestión de Defectos” (DM).

En relación con lo comentado en el párrafo anterior, la práctica más eficiente sería establecer mecanismos automatizados para corregir las desviaciones o gestionar las no conformidades, como pueden ser la utilización de estrategias de "autorreparación" (self-healing)[10] de la configuración o el disparo de alertas de gestión de información y eventos de seguridad (SIEM).

Cada vez que se produzca una actualización de algún componente (nuevas versiones, parches de fabricante, etc.) se revisará la documentación de las líneas de base y guías de configuración para actualizarlas convenientemente, manteniendo así su vigencia y exactitud.

Independientemente de las actualizaciones debidas a cambios de versión de los componentes, se revisarán las líneas de base y guías de configuración al menos una vez al año. Adicionalmente, se revisará periódicamente el propio proceso de

[10] Mecanismo que permite el autodiagnóstico de los sistemas y su reacción, de forma que se restablezca un modo de funcionamiento satisfactorio tras la aparición de un fallo o defecto (https://core.ac.uk/download/pdf/62914417.pdf)

gestión de las líneas base y guías de configuración para asegurar que se tienen en cuenta e incorporan a las mismas los comentarios y las lecciones aprendidas de los equipos encargados de su aplicación y mantenimiento.

Línea de acción Parcheado y Actualización (Stream B): Parcheado diligente de la configuración.

El objetivo perseguido por esta actividad es que la organización disponga de una visión clara del estado de los parches de todos los componentes software para evitar incumplimientos sobre lo establecido por el proceso de gestión del parcheado.

Se elaborarán y utilizarán cuadros de mando/informes sobre la actividad de parcheado para hacer seguimiento del cumplimiento de gestión de parcheado y los SLA establecidos. Esto incluirá la verificación de que la lista con componentes y versiones se mantiene debidamente actualizada.

Si se descubre que no se ha realizado alguna actualización sin la debida justificación, este hecho se tratará como un defecto asociado al producto y se gestionará conforme a lo establecido por la Práctica “Gestión de Defectos” (DM).

La organización no confiará exclusivamente en las notificaciones habituales de los fabricantes para conocer las vulnerabilidades de sus soluciones y los parches asociados, sino que se revisarán fuentes externas de información sobre amenazas para tener conocimiento de vulnerabilidades de día cero y gestionarlas como un problema de gestión de riesgos.

4.2.5.3 Práctica Gestión Operativa

La práctica de Gestión Operativa (OM) pone el foco sobre las actividades que garantizan el mantenimiento de la seguridad en todas las funciones de apoyo operativo. Aunque estas funciones no son realizadas directamente por las aplicaciones, la seguridad general de las mismas y la de sus datos depende de su correcto funcionamiento. El despliegue de una aplicación en un sistema operativo no soportado o con vulnerabilidades, o el no almacenamiento seguro de los soportes de copia de seguridad, puede provocar que las protecciones incorporadas en dicha aplicación sean inútiles.

Las funciones cubiertas por esta práctica incluyen, como mínimo:

- El aprovisionamiento, administración y desmantelamiento del sistema
- El aprovisionamiento y la administración de la base de datos;
- Las copias de seguridad, su restauración y el archivado de datos.

En la siguiente figura, se muestran las actividades asociadas a los distintos niveles de madurez comprendidos en la práctica.

Maturity level		Stream A Data Protection	Stream B System Decomissioning / Legacy Management
1	Foundational Practices	Implement basic data protection practices	Decomission unused applications and services as identified. Manage customer upgrades/migrations individually.
2	Managed, Responsive Processes	Develop data catalog and establish data protection policy.	Develop repeatable decommissioning processes for unused systems/services, and for migration from legacy dependencies. Manage legacy migration roadmaps for customers.
3	Active Monitoring and Response	Automate detection of policy non-compliance, and audit compliance periodically. Regularly review and update to data catalog and data protection policy.	Proactively manage migration roadmaps, for both unsupported end-of-life dependencies, and legacy versions of delivered software.

Figura 21. Actividades de la práctica Gestión Operativa. Fuente: OWASP SAMM

Como se puede observar en la figura anterior, se definen las dos líneas acción para esta práctica: *Protección de los Datos* (Stream A) y Desmantelamiento de Sistemas y Gestión del *Legacy* (Stream B).

Las actividades de gestión de los datos se ocupa de garantizar que la organización proteja adecuadamente los datos en todas sus fases: creación, manipulación, almacenamiento, procesado y destrucción.

La segunda línea de acción se centra en la identificación, gestión y seguimiento de sistemas, aplicaciones, dependencias y servicios que ya no se utilizan, que han llegado al final de su vida útil o que ya no se soportan activamente. La eliminación de los sistemas y servicios no utilizados mejora la capacidad de gestión del entorno operativo y reduce la superficie de ataque de la organización. Además, permite ahorrar costes directos e indirectos como por ejemplo, la reducción del número de licencias, reducción del volumen de almacenamiento o la reducción del esfuerzo mantenimiento.

4.2.5.3.1 Gestión Operativa (OM1)

El objetivo de la práctica en el nivel de madurez inicial es comenzar a realizar actividades básicas de gestión.

Línea de acción Protección de los Datos (Stream A): Protección básica de los datos

La finalidad es que la organización tenga una visión clara de la sensibilidad de los datos que maneja y se establezcan algunas medidas sencillas y de rápida aplicación (*quick wins*) para la protección de los mismos.

La organización recopilará información que le permita conocer los tipos y grados de sensibilidad de los datos que son procesados por las aplicaciones, y sea consciente del destino o destinos finales de los mismos (por ejemplo, copias de seguridad, transferencia a socios externos, etc.).

En este nivel de madurez, se supone que no se dispone de un catálogo de datos en la organización. En estas circunstancias, se protegerán y gestionarán todos los datos asociados a una aplicación conforme a los requisitos de protección aplicables a los datos conocidos de mayor sensibilidad.

Se implementarán controles básicos, para evitar la propagación o compartición de datos sensibles no anonimizados desde el entorno de producción a otros entornos (desarrollo, pre-producción, QA, etc.).

Línea de acción Desmantelamiento de Sistemas y Gestión del Legacy (Stream B): Identificar aplicaciones no utilizadas

El objetivo de esta actividad es que la organización conozca los activos o componentes software instalados en el entorno de producción que ya no son utilizados.

Se identificarán todas las aplicaciones no utilizadas de forma *ad hoc*, ya sea por observación casual o realizando alguna revisión ocasional. Cuando se identifique alguna aplicación que no es utilizada, se gestionará la correspondiente acción correctiva para su desmantelamiento. Si existe un proceso formal para el desmantelamiento de las aplicaciones no utilizadas, se deberá asegurar que es conocido y seguido por el personal técnico correspondiente.

En la medida de lo posible, se fomentará la migración de clientes o usuarios desde versiones anteriores de los productos hacia las nuevas versiones. Cuando una versión de algún producto ya no sea utilizada por ningún grupo de clientes o usuarios, se dejará de dar soporte a dicha versión. De esta forma, además, se disminuirá el esfuerzo de los equipos de desarrollo en el mantenimiento de las aplicaciones.

4.2.5.3.2 Gestión Operativa (OM2)

El objetivo de la práctica en este nivel de madurez es establecer procesos gestionados consistentes que satisfagan las necesidades de la organización.

Línea de acción Protección de los Datos (Stream A): Elaborar el catálogo de datos.

Esta actividad tiene como objetivo estandarizar la gestión de los datos de diversa naturaleza que gestionan las aplicaciones de la organización.

En este nivel de madurez, las actividades de protección de datos se centran en la gestión activa de la administración de los datos.

Se identificarán los datos almacenados, procesados y transmitidos por las aplicaciones, y se clasificará de acuerdo a sus tipos, niveles de sensibilidad y ubicaciones donde se almacenan. Adicionalmente, se identificarán claramente los registros sujetos a algún tipo de regulación. Toda esta información constituirá el catálogo de datos de la organización.

El catálogo de datos actuará como única “fuente de verdad”, lo que permitirá una selección más precisa de los controles que se aplicarán para proteger los datos y conocer con exactitud qué grupos de datos están afectados por regulaciones específicas. Adicionalmente, la recopilación de esta información mejora la precisión, la rapidez y la eficiencia de las respuestas a las consultas relacionadas, como por ejemplo las que realizan los auditores, los equipos de respuesta a incidentes o los clientes, y respalda el modelado de amenazas y las actividades relacionadas con el cumplimiento normativo.

Se establecerán controles técnicos y administrativos para proteger la confidencialidad de los datos sensibles, y la integridad y disponibilidad de todos los datos gestionados, desde su creación hasta la destrucción de las copias de seguridad al final del período de conservación.

Se establecerán procesos y procedimientos basados en la política de protección de datos de la organización, para proteger y preservar los datos durante toda su vida útil, ya sea en reposo, mientras se procesan o en tránsito. Se deberá prestar especial atención al manejo y protección de datos sensibles fuera del sistema de procesamiento activo, lo que incluye, entre otros aspectos, el almacenamiento, conservación y destrucción de las copias de seguridad, así como el etiquetado, cifrado y protección física de los medios de almacenamiento fuera de línea.

Los procesos y procedimientos definidos cubrirán la aplicación de los controles adoptados para cumplir con las restricciones normativas, legales o de cualquier otro tipo, sobre los lugares de almacenamiento, el acceso del personal y otros factores relevantes.

Línea de acción Desmantelamiento de Sistemas y Gestión del Legacy (Stream B): Formalizar el proceso de desmantelamiento de sistemas

El objetivo de esta actividad es la estandarización del proceso de desmantelamiento de aplicaciones y sistemas con el fin de reducir el riesgo de que existan componentes obsoletos no utilizados en el entorno de operación.

Se establecerá un procedimiento para el desmantelamiento de sistemas, aplicaciones o servicios que siga un una serie de pasos bien definidos, eliminando todas las cuentas, reglas de cortafuegos, datos, ficheros de configuración, etc., del entorno operativo.

Se seguirá un proceso consistente para la sustitución o actualización oportuna de aplicaciones de terceros y dependencias de las aplicaciones tales como sistemas operativos, utilidades, librerías, etc., que han llegado al final de su vida útil. Para que la organización sea eficaz en la ejecución de estas tareas, será de gran ayuda contar con una lista actualizada de todos los componentes del entorno de producción con información sobre su versionado y las fechas de fin de soporte/vida útil.

La organización se comprometerá con aquellos clientes/usuarios que utilicen aplicaciones que hayan llegado al final de su vida útil, a migrarlos de forma oportuna a nuevas versiones compatibles de las aplicaciones. Para ello, será de gran ayuda contar con una lista actualizada que muestre el estado y fechas de fin de soporte/vida útil de las versiones liberadas de las aplicaciones de la organización.

4.2.5.3.3 Gestión Operativa (OM3)

El objetivo de la práctica en este nivel de madurez es dotar a la organización de las capacidades de detección y reacción.

Línea de acción Protección de los Datos (Stream A): Respuesta ante violaciones de datos

Esta actividad tiene como finalidad garantizar el cumplimiento técnico de la política de protección de datos de la organización.

Las actividades de esta línea de acción se centran en la automatización de la protección de datos, reduciendo en la medida de lo posible la dependencia del

esfuerzo humano para evaluar y gestionar el cumplimiento de las políticas. Se hace hincapié en los mecanismos de retroalimentación y las revisiones proactivas, para identificar y actuar sobre las oportunidades de mejora del proceso.

Se implantarán controles técnicos para garantizar el cumplimiento de la política de protección de datos, y se realizará monitorización para detectar intentos de infracción o infracciones consumadas. Se utilizarán herramientas para la prevención de la pérdida de datos, el control y el seguimiento del acceso a los datos o la detección de comportamientos anómalos.

Se auditará regularmente el cumplimiento de los controles administrativos establecidos, y se supervisará de cerca el funcionamiento de los mecanismos automatizados, incluyendo la fiabilidad de las copias de seguridad y la eliminación de registros. Las herramientas de supervisión detectarán y alertarán rápidamente de los fallos en la automatización, lo que permitirá tomar a tiempo medidas correctivas.

Por último, se deberá revisar y actualizar periódicamente el catálogo de datos, para que siga reflejando con exactitud la realidad.

Línea de acción Desmantelamiento de Sistemas y Gestión del Legacy (Stream B): Revisar periódicamente el estado de soporte de las aplicaciones

El objetivo de esta actividad es que la organización tenga una visión completa del ciclo de vida de todos los activos de software.

De forma periódica se revisará la información sobre el estado del ciclo de vida y soporte de cada activo de software y componente de la infraestructura, estimando el final de la vida útil de los mismos. Adicionalmente, se definirá un proceso para mitigar los riesgos de seguridad que surgen a medida que los activos/componentes se acercan al final de su vida útil. Este proceso se revisará y actualizará periódicamente (al menos una vez al año), para incluir las lecciones aprendidas.

En el caso de los productos y servicios desarrollados por la propia organización, se establecerá el correspondiente plan de soporte

del producto, proporcionando plazos claros sobre el final de soporte de las versiones más antiguas del producto.

Se limitará el número de versiones activas de los productos/servicios. Por ejemplo, solo se mantendrán activas las versiones N.x.x y N-1.x.x de un producto y se harán públicos los plazos para el final del soporte de versiones anteriores, con el fin de que los usuarios/clientes no sufran interrupciones de servicio inesperadas.

5 Aplicar el modelo: Programa de mejora de la seguridad del software

Quizás lo más difícil en cualquier iniciativa de cambio es precisamente dar los primeros pasos, arrancar. En la mayoría de ocasiones, el camino para la transformación o la mejora se percibe como algo muy largo y la cantidad de acciones a ejecutar o la magnitud de las mismas parece inabordable. Para tener éxito, o estar cerca del mismo, la mejor forma de enfocar este tipo de procesos es dividir el enorme esfuerzo en una serie de pequeños pasos y definir objetivos intermedios que permitan acercarnos progresivamente al objetivo final. Como dijo el famoso filósofo Lao-Tse, "*Un viaje de mil millas comienza con el primer paso*".

Según se recomienda en la guía de inicio rápido de OWASP [22], aquellas organizaciones que se planteen iniciar un programa de seguridad en el software haciendo uso de SAMM, deberían llevar a cabo los siguientes pasos:

1. Preparación inicial
2. Evaluación de la situación
3. Definición de objetivos
4. Planificación de acciones
5. Ejecución de actividades
6. Puesta en marcha del programa

Si se desea realizar una primera aproximación rápida al modelo, los pasos 1 a 4 podrían ser llevados a cabo de forma muy rápida y con un esfuerzo mínimo[11]. Sin embargo, las fases 5 y 6 requerirán, sin duda bastante más esfuerzo, recursos y tiempo.

El programa de seguridad del software que se defina, se debería enfocar como un proceso iterativo de mejora continua en el que, en cada iteración, la organización irá ganando madurez de forma incremental hasta llegar a objetivo final, conforme a sus necesidades de negocio. En ese sentido, la filosofía con la que se construye el modelo es coherente con este enfoque de mejora continua mediante un proceso cíclico que se ejecute con una periodicidad de 3 a 12 meses, según la organización.

En los siguientes subapartados, se describen brevemente los pasos mencionados para poner en marcha el programa de seguridad del software.

5.1 Preparación inicial

Como en cualquier otro tipo de proyecto e iniciativa, es necesario realizar unas mínimas acciones iniciales para asegurar que el proyecto pueda arrancar de forma adecuada.

Definir el alcance preliminar

[11] Según la guía de inicio rápido podrían ser llevadas a cabo incluso por una sola persona en el plazo de 1-2 días.

Al igual que se procedería con cualquier proyecto, sea del tipo que sea, uno de los primeras acciones es definir el alcance del mismo. El alcance determina el objetivo de la iniciativa, qué queda incluido y qué no (límites del proyecto), los resultados o productos finales que se obtendrán, las expectativas, cómo se organizará el proyecto y una estimación preliminar de costes.

Un aspecto muy relevante es la definición del ámbito donde se aplicará el programa de seguridad. Por ejemplo, en el caso que nos aplica se determinará si el programa de seguridad se aplica sobre todas aquellas unidades organizativas que desarrollen software, si solo afectará a un subconjunto de las mismas, o incluso si afectará únicamente a un equipo de desarrollo concreto. Este aspecto es importante pues en función de la extensión de la iniciativa, será más o menos sencillo llevar a cabo la puesta en marcha. Cuanto más limitado sea el alcance de aplicación, más sencillo será poner en marcha el programa de mejora. Posteriormente, en función de los resultados, se podrá ir ampliando al resto de la organización.

Identificar a las partes interesadas

Se deberá identificar a las partes interesadas más relevantes, asegurar que estén debidamente informadas y confirmar su grado de compromiso e interés sobre la iniciativa de mejora. Para la puesta en marcha del programa de seguridad se requerirá una alta participación, esfuerzo y soporte por parte de muchas personas, especialmente de la alta dirección. Sin estos ingredientes, existe un gran riesgo de incumplimiento de los objetivos y, por tanto, de fracaso del proyecto.

La lista de partes interesadas dependerá en gran medida de la estructura de la organización. A modo de ejemplo, a continuación se muestra una posible relación de involucrados:

- Patrocinador ejecutivo (alta dirección)
- Equipo de seguridad
- Responsables de desarrollo
- Arquitectos
- Desarrolladores
- Equipos de QA

Informar sobre la iniciativa

Se realizará un ejercicio de comunicación de la iniciativa para difuminar el conocimiento, transmitir la importancia y beneficios esperados de la misma. Hay que tener en cuenta que la puesta en marcha de un programa de aseguramiento del software requerirá esfuerzos importantes por parte de la organización, asignación de recursos, cambios estructurales, modificación de los procedimientos de trabajo, etc. Por ello es preciso que todas las personas que puedan estar afectadas conozcan la importancia de lo que van a hacer, el enfoque estratégico y los beneficios esperados que la puesta en marcha del programa tendrá para la organización.

5.2 Evaluación (AS-IS)

Una vez que la organización ha tomado la decisión de abordar la puesta en marcha de un programa de seguridad en el software, es necesario tener una idea clara del estado actual o punto de partida.

Para definir la estrategia de implantación, es preciso conocer el punto de partida y determinar la meta final. Únicamente conociendo dónde estamos y dónde queremos llegar, se puede planificar el camino a seguir.

Evaluar prácticas de seguridad actuales

El primer paso es conocer y valorar las prácticas que actualmente se llevan a cabo en los procesos de desarrollo de la organización o unida organizativa sobre la que se aplicará la iniciativa de mejora. Para ello, se entrevistará a las partes interesadas correspondientes y se les preguntará sobre las prácticas de seguridad que realiza la organización.

El propio modelo ofrece herramientas para llevar a cabo la autoevaluación. La herramienta más sencilla consiste en una hoja de cálculo [23] donde figura una lista de preguntas para evaluar el estado de madurez de las distintas actividades del modelo. Cada una de las preguntas tiene un conjunto de criterios de calidad enumerados bajo la pregunta. Adicionalmente, la herramienta contiene una enumeración de posibles respuestas para cada una de las preguntas. Para responder a cada pregunta, hay que tener en cuenta si se cumplen todos los criterios de calidad. Si los criterios de calidad no se cumplen completamente, la respuesta deberá ser "No". En caso de que sí se cumplan todos los criterios de calidad, se elegirá la respuesta correspondiente entre el resto de opciones. Para comprender mejor el funcionamiento de la herramienta, se muestra un ejemplo en la Figura 22.

		Security Requirements	Answer	Interview Notes	Rating
Software Requirements	1	Do project teams specify security requirements during development?	Yes, for at least half of the applications		0,38
		Teams derive security requirements from functional requirements and customer or organization concerns Security requirements are specific, measurable, and reasonable Security requirements are in line with the organizational baseline			
	2	Do you define, structure, and include prioritization in the artifacts of the security requirements gathering process?	No		
		Security requirements take into consideration domain specific knowledge when applying policies and guidance to product development Domain experts are involved in the requirements definition process You have an agreed upon structured notation for security requirements Development teams have a security champion dedicated to reviewing security requirements and outcomes			
	3	Do you use a standard requirements framework to streamline the elicitation of security requirements?	Yes, for some applications		
		A security requirements framework is available for project teams The framework is categorized by common requirements and standards-based requirements The framework gives clear guidance on the quality of requirements and how to describe them The framework is adaptable to specific business requirements			

Figura 22. Ejemplo de autoevaluación con herramienta SAMM

Para lograr un resultado consistente, se deberá asegurar que los participantes en las encuestas y talleres de evaluación entienden bien las preguntas, la materia y las actividades del modelo sobre las que tratan. Además, se evaluará la posibilidad de entrevistar a diversas personas o equipos sobre la misma cuestión, o la realización de las encuesta de forma anónima para asegurar la honestidad de las respuestas.

Determinar el nivel de madurez

Sobre la base de la información recopilada durante las entrevistas/encuestas, se deberá determinar el nivel de madurez de cada una de las actividades, prácticas y funciones de negocio. Finalmente, obtendremos el nivel de madurez global de la organización en materia de desarrollo seguro.

La propia herramienta realiza el cálculo del nivel de madurez, promediando las puntuaciones obtenidas a partir de las respuestas para cada actividad y sumándose para obtener la calificación global (Figura 23-26).

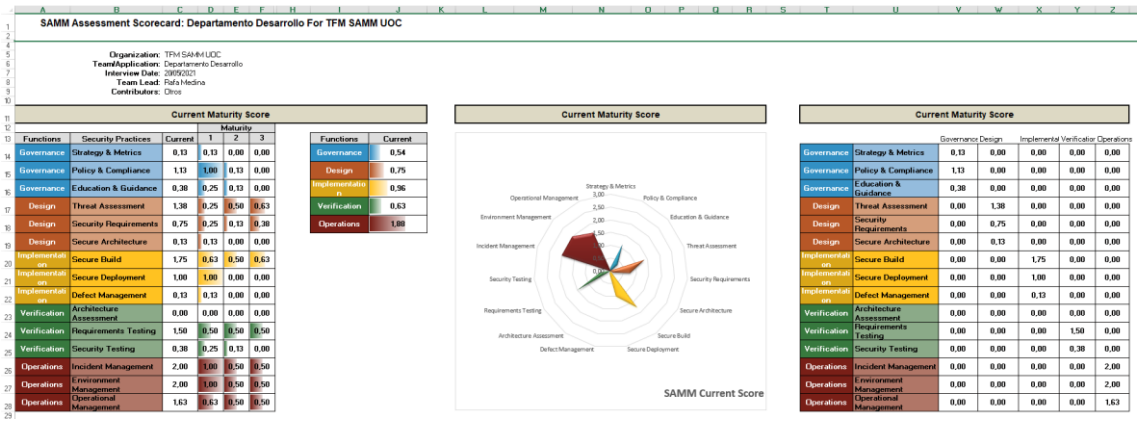


Figura 23. Simulación autoevaluación de organización ficticia con SAMM Assessment Toolbox 2.0

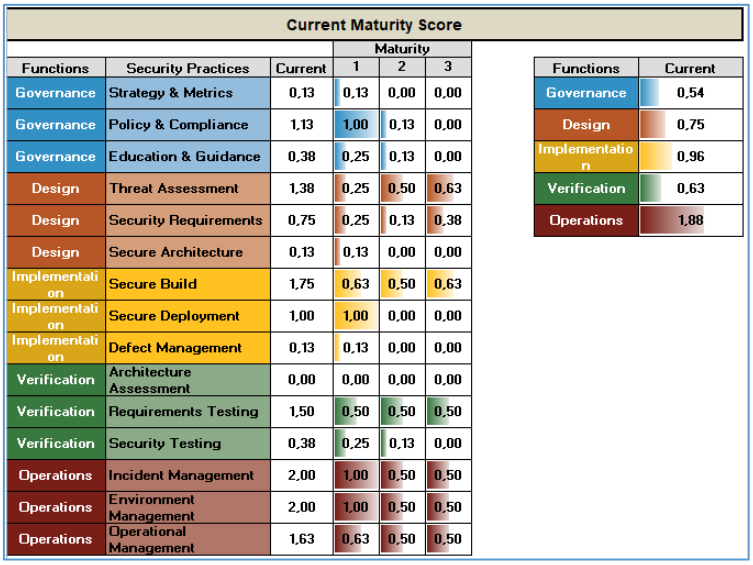


Figura 24. Detalle autoevaluación de organización ficticia con SAMM Assessment Toolbox 2.0

Current Maturity Score						
		Governance	Design	Implementa	Verificatio	Operations
Governance	Strategy & Metrics	0,13	0,00	0,00	0,00	0,00
Governance	Policy & Compliance	1,13	0,00	0,00	0,00	0,00
Governance	Education & Guidance	0,38	0,00	0,00	0,00	0,00
Design	Threat Assessment	0,00	1,38	0,00	0,00	0,00
Design	Security Requirements	0,00	0,75	0,00	0,00	0,00
Design	Secure Architecture	0,00	0,13	0,00	0,00	0,00
Implementation	Secure Build	0,00	0,00	1,75	0,00	0,00
Implementation	Secure Deployment	0,00	0,00	1,00	0,00	0,00
Implementation	Defect Management	0,00	0,00	0,13	0,00	0,00
Verification	Architecture Assessment	0,00	0,00	0,00	0,00	0,00
Verification	Requirements Testing	0,00	0,00	0,00	1,50	0,00
Verification	Security Testing	0,00	0,00	0,00	0,38	0,00
Operations	Incident Management	0,00	0,00	0,00	0,00	2,00
Operations	Environment Management	0,00	0,00	0,00	0,00	2,00
Operations	Operational Management	0,00	0,00	0,00	0,00	1,63

Figura 25. Detalle 2 autoevaluación de organización ficticia con SAMM Assessment Toolbox 2.0

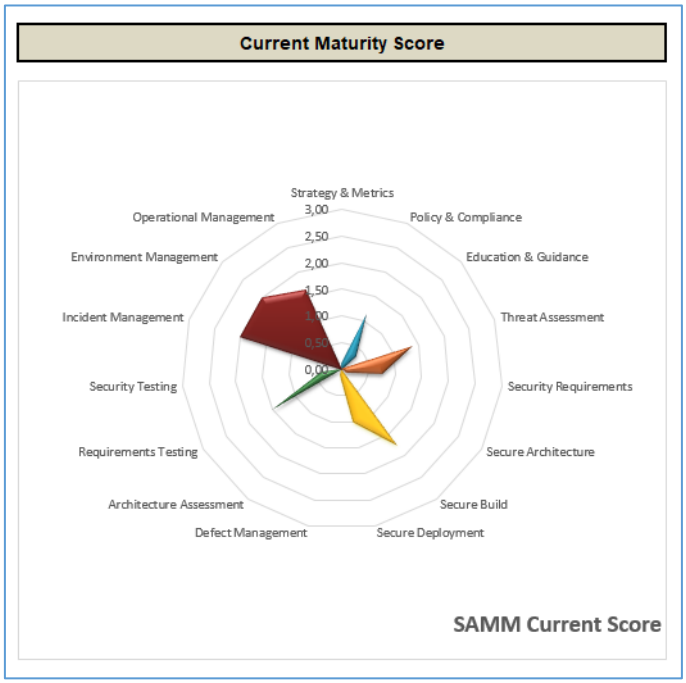


Figura 26. Detalle 3 autoevaluación de organización ficticia con SAMM Assessment Toolbox 2.0

Además de la herramienta mencionada basada en una hoja de cálculo, se han desarrollado otras herramientas de autoevaluación. Por ejemplo, existe una calculadora online [24], así como una aplicación Web mucho más completa [25] que permite no solo realizar la evaluación inicial, sino que proporciona un exhaustivo cuadro de mando (Figura 27) con el que se podrá realizar seguimiento del estado de madurez de la organización y la evolución del programa de seguridad en el software a lo largo del tiempo.

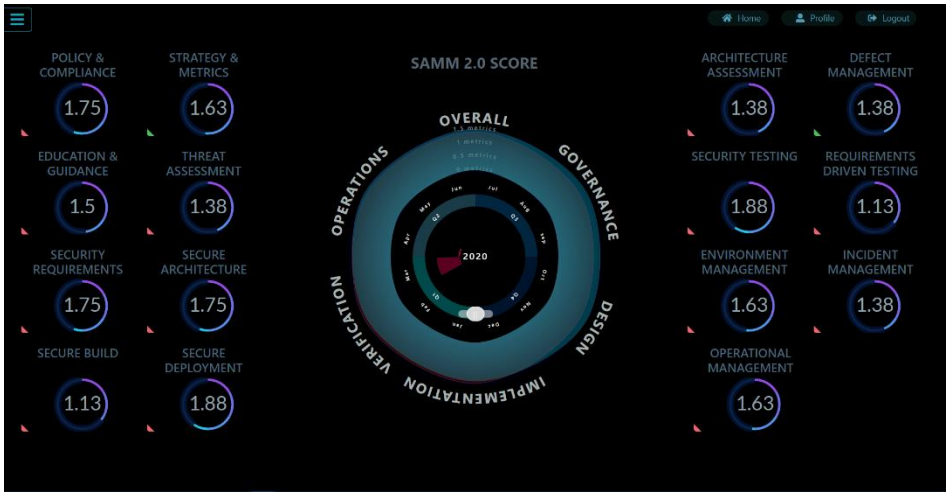


Figura 27. Cuadro de mando SAMM. Fuente OWASP SAMM

Como se ha podido ver, existen diversos recursos y herramientas que SAMM pone a disposición de las organizaciones, que facilitan enormemente la evaluación del nivel de madurez, así como el seguimiento del programa de mejora seguridad y la gobernanza de los procesos de desarrollo seguro.

5.3 Definición de objetivos (TO-BE)

Tras la evaluación de la situación de partida, el siguiente paso es definir el objetivo hacia el que la organización quiere dirigirse.

Definir objetivo

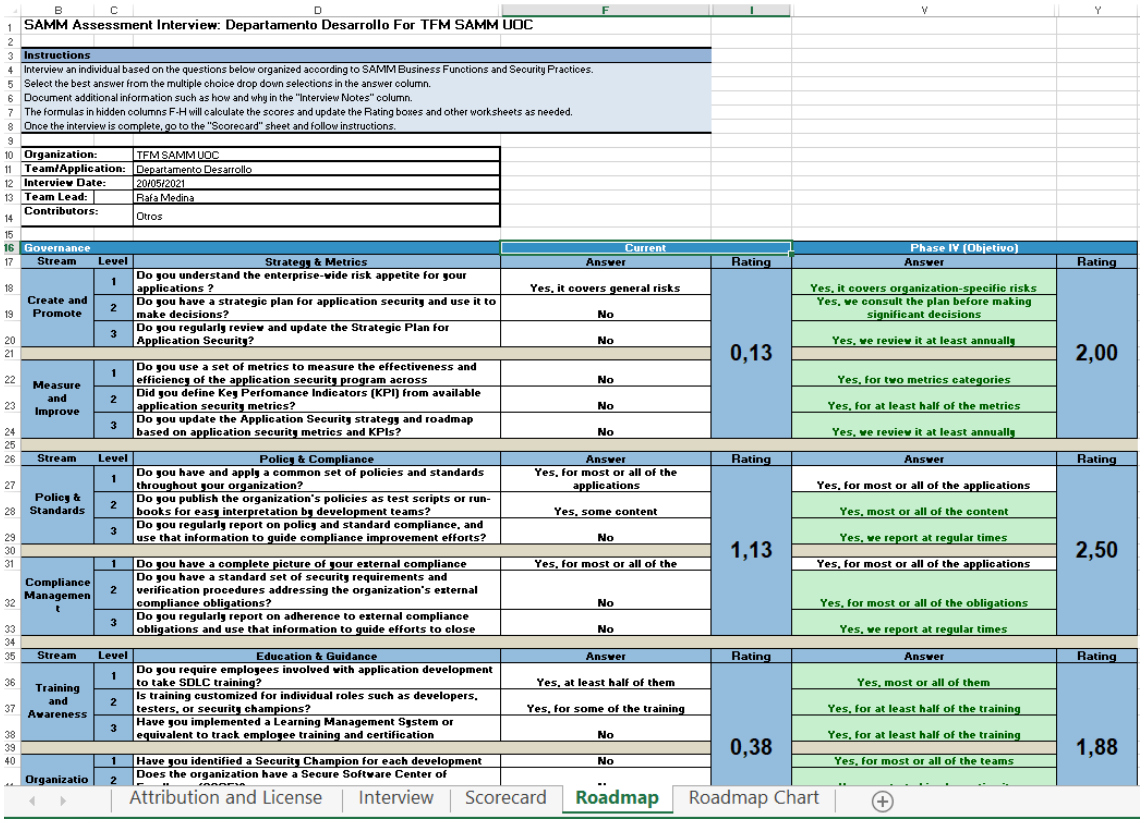
Una buena forma de determinar el objetivo es identificar aquellas actividades del modelo que la organización considera que se deberían llevar a cabo en un escenario ideal, teniendo en cuenta sus necesidades de negocio. Una vez identificadas, se deberá revisar la propuesta para asegurar que el conjunto total de actividades seleccionadas es coherente con los intereses de la organización y se tienen en cuenta las dependencias entre actividades.

A partir de la identificación de las actividades que la organización quiere implementar, se determinará el nivel de madurez objetivo (puntuación). Este servirá como referencia e instrumento de medida del progreso de la iniciativa de mejora.

El cálculo mencionado anteriormente se podrá realizar con la misma herramienta proporcionada para la evaluación del nivel de madurez, utilizando la función incluida para definir la hoja de ruta. En dicha hoja de cálculo, existen columnas para la evaluación inicial y una serie de fases (fase 1 a 4) donde se supone que se fijarán los objetivos para diversas iteraciones de mejora, siendo las fases 1 a 3 donde se definirán los objetivos intermedios. En la fase 4 se fijarán los objetivos finales. La cumplimentación de todos estos datos proporcionará la hoja de ruta a seguir con las diversas acciones de mejora que habrá que ir ejecutando en las diversas iteraciones.

En las siguientes figuras (Figura 28 Figura 29) se muestra una simulación donde, tras la evaluación inicial, se definen los objetivos finales en la herramienta

proporcionada por OWASP. Como se puede observar la herramienta nos permite comparar la situación inicial con proyectada para el futuro, así como el nivel de madurez final que se alcanzará para cada una de las funciones de negocio, prácticas y globalmente.



SAMM Assessment Interview: Departamento Desarrollo For TFM SAMM UOC

Instructions

Interview an individual based on the questions below organized according to SAMM Business Functions and Security Practices.
Select the best answer from the multiple choice drop down selections in the answer column.
Document additional information such as how and why in the "Interview Notes" column.
The formulas in hidden columns F-H will calculate the scores and update the Rating boxes and other worksheets as needed.
Once the interview is complete, go to the "Scorecard" sheet and follow instructions.

Organization:	TFM SAMM UOC
Team/Application:	Departamento Desarrollo
Interview Date:	20/05/2021
Team Lead:	Rafa Medina
Contributors:	Otros

Governance			Current		Phase IV (Objetivo)	
Stream	**Level**	**Strategy & Metrics**	**Answer**	**Rating**	**Answer**	**Rating**
Create and Promote	1	Do you understand the enterprise-wide risk appetite for your applications ?	Yes, it covers general risks	0,13	Yes, it covers organization-specific risks	2,00
	2	Do you have a strategic plan for application security and use it to make decisions?	No		Yes, we consult the plan before making significant decisions	
	3	Do you regularly review and update the Strategic Plan for Application Security?	No		Yes, we review it at least annually	
Measure and Improve	1	Do you use a set of metrics to measure the effectiveness and efficiency of the application security program across	No		Yes, for two metrics categories	
	2	Did you define Key Performance Indicators (KPI) from available application security metrics?	No		Yes, for at least half of the metrics	
	3	Do you update the Application Security strategy and roadmap based on application security metrics and KPIs?	No		Yes, we review it at least annually	
Stream	**Level**	**Policy & Compliance**	**Answer**	**Rating**	**Answer**	**Rating**
Policy & Standards	1	Do you have and apply a common set of policies and standards throughout your organization?	Yes, for most or all of the applications	1,13	Yes, for most or all of the applications	2,50
	2	Do you publish the organization's policies as test scripts or run-books for easy interpretation by development teams?	Yes, some content		Yes, most or all of the content	
	3	Do you regularly report on policy and standard compliance, and use that information to guide compliance improvement efforts?	No		Yes, we report at regular times	
Compliance Management	1	Do you have a complete picture of your external compliance	Yes, for most or all of the		Yes, for most or all of the applications	
	2	Do you have a standard set of security requirements and verification procedures addressing the organization's external compliance obligations?	No		Yes, for most or all of the obligations	
	3	Do you regularly report on adherence to external compliance obligations and use that information to guide efforts to close	No		Yes, we report at regular times	
Stream	**Level**	**Education & Guidance**	**Answer**	**Rating**	**Answer**	**Rating**
Training and Awareness	1	Do you require employees involved with application development to take SDLC training?	Yes, at least half of them	0,38	Yes, most or all of them	1,88
	2	Is training customized for individual roles such as developers, testers, or security champions?	Yes, for some of the training		Yes, for at least half of the training	
	3	Have you implemented a Learning Management System or equivalent to track employee training and certification	No		Yes, for at least half of the training	
Organizatio	1	Have you identified a Security Champion for each development	No		Yes, for most or all of the teams	
	2	Does the organization have a Secure Software Center of				

Attribution and License | Interview | Scorecard | **Roadmap** | Roadmap Chart

Figura 28. Definición objetivo final con herramienta SAMM Assessment Tool 2.0

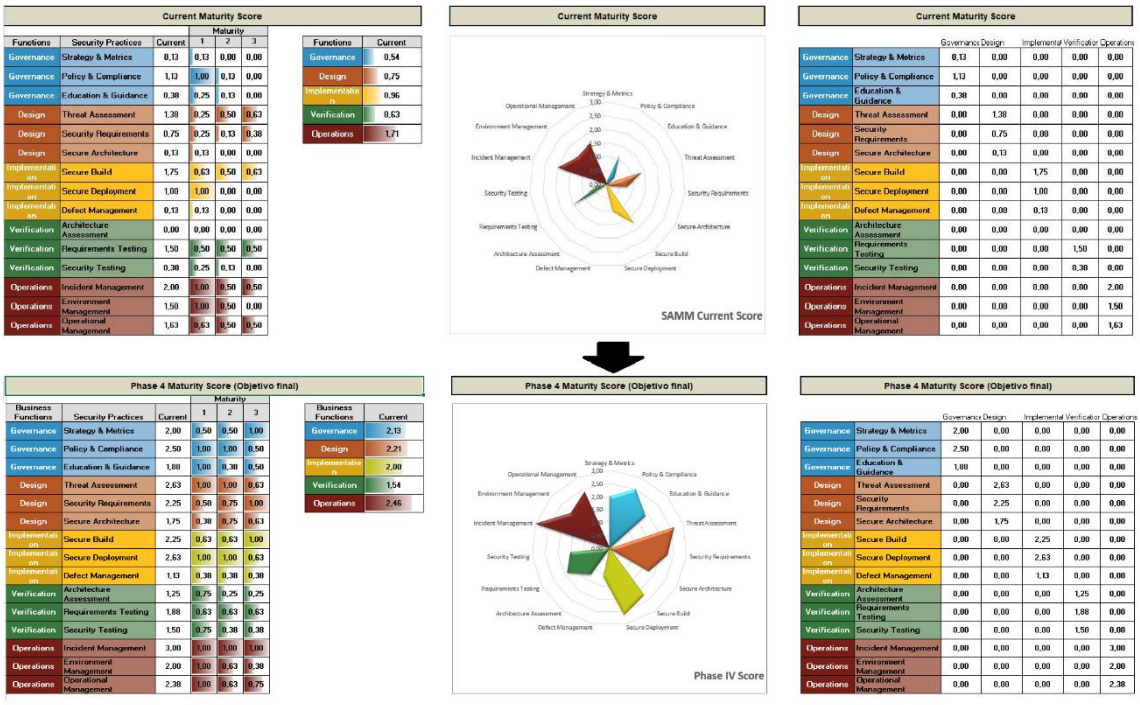


Figura 29. Situación inicial vs Objetivo con herramienta SAMM Assessment Tool 2.0

Estimar impacto global

Además de establecer el objetivo de la iniciativa, se estimará el impacto de dicho objetivo para la organización en términos de esfuerzo y recursos necesarios. En la medida de lo posible, se intentará expresar el impacto para la organización en términos presupuestarios. Según se define en la documentación de SAMM, se estima que el impacto global del esfuerzo del programa de aseguramiento del software seguro, supondrá entre el 5% y el 10% del coste total asociado al desarrollo de software.

5.4 Mejoras a realizar (GAP)

Conociendo la situación de partida y la meta final propuesta por la organización, la herramienta facilita la identificación de las diferencias en las actividades y prácticas de seguridad que nos separan de la situación futura proyectada. Esta información servirá para tener una idea de la magnitud de los cambios a realizar, y, por tanto, de los esfuerzos, recursos y tiempo que necesitará la organización para lleva a cabo su plan de mejora.

A partir de aquí, el siguiente paso consistirá en planificar la hoja de ruta que permitirá evolucionar a la organización desde la situación inicial hasta alcanzar los objetivos definidos.

5.5 Planificar (Road Map)

Los cambios que debe realizar la aplicación para llegar a su objetivo final, deben ser planificados cuidadosamente. En la mayoría de casos, el salto es tan grande que no sería prudente ni realista intentar acometerlo de una sola vez. En lugar de esto, se deberá diseñar una estrategia del cambio que defina con una hoja de ruta realista que establezca una serie de etapas o fases, a través de las cuales,

la organización se aproxime sucesivamente a su objetivo final. Según SAMM, una hoja de ruta típica constará de un total de 4 a 6 fases, con una duración de entre 3 y 12 meses cada una, en función de las capacidades de la organización.

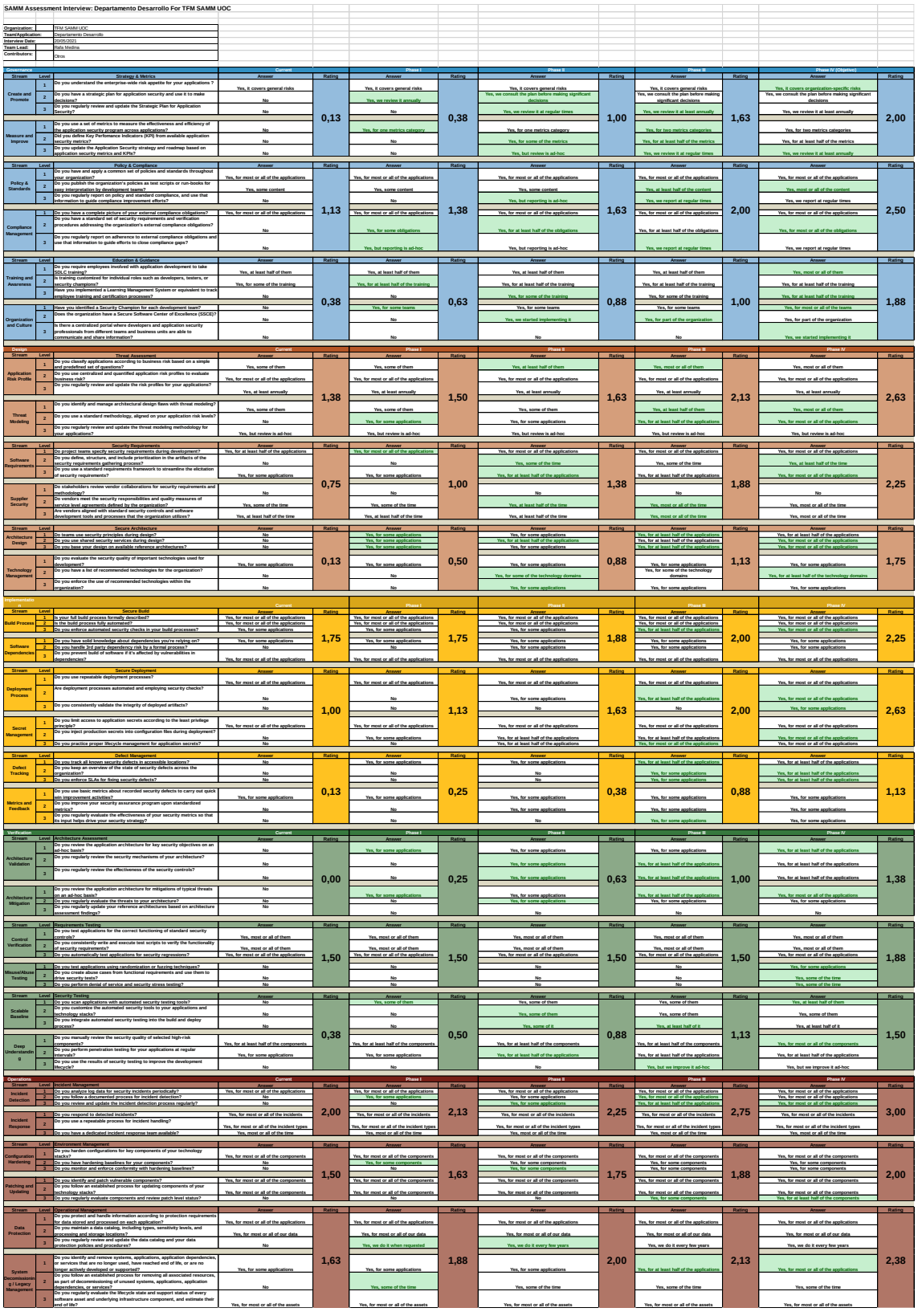
Para la elaboración del plan sería conveniente tener en cuenta las capacidades, prioridades, así como otros planes o proyectos clave de la organización.

Una buena práctica a la hora de planificar, es identificar un conjunto de acciones que pueden completarse de forma rápida y con éxito al principio del proyecto (*quick wins*). Esto genera confianza y favorece la obtención de un mayor respaldo. En definitiva, beneficia enormemente al desarrollo del proyecto.

Como primeras acciones a planificar, conviene tener en cuenta las actividades relacionadas con la concienciación y formación en materia de seguridad en el software, pues actúan como catalizador del proyecto.

En cualquier caso, se realizará una distribución de las actividades en las distintas fases de la hoja de ruta, teniendo en cuenta el esfuerzo asociado, las dependencias entre actividades, e intentando equilibrar la carga de trabajo en los distintos periodos de ejecución.

En las siguientes figuras (Figura 30 y Figura 31) se muestra una simulación donde se planifica una hoja de ruta consistente en 4 fases en la herramienta proporcionada por OWASP. Se puede observar la distribución de las diversas actividades así como la evolución del nivel de madurez de la organización a lo largo de las diferentes etapas.



SAMM Assessment Interview: Departamento Desarrollo For TFM SAMM UOC

Organization:	TFM SAMM UOC
Team/Application:	Departamento Desarrollo
Interview Date:	20/05/2021
Team Lead:	Rafa Medina
Contributors:	Otros

Governance	Current		Phase I		Phase II		Phase III		Phase IV (Objetivo)	
Stream	Level	Strategy & Metrics	Answer	Rating	Answer	Rating	Answer	Rating	Answer	Rating	Answer	Rating
Create and Promote	1	Do you understand the enterprise-wide risk appetite for your applications ?	Yes, it covers general risks	0,13	Yes, it covers general risks	0,38	Yes, it covers general risks	1,00	Yes, it covers general risks	1,63	Yes, it covers organization-specific risks	2,00
	2	Do you have a strategic plan for application security and use it to make decisions?	No		Yes, we review it annually		Yes, we consult the plan before making significant decisions		Yes, we consult the plan before making significant decisions		Yes, we consult the plan before making significant decisions	
	3	Do you regularly review and update the Strategic Plan for Application Security?	No		No		Yes, we review it at regular times		Yes, we review it at least annually		Yes, we review it at least annually	
Measure and Improve	1	Do you use a set of metrics to measure the effectiveness and efficiency of the application security program across applications?	No		Yes, for one metrics category		Yes, for one metrics category		Yes, for two metrics categories		Yes, for two metrics categories	
	2	Did you define Key Performance Indicators (KPI) from available application security metrics?	No		No		Yes, for some of the metrics		Yes, for at least half of the metrics		Yes, for at least half of the metrics	
	3	Do you update the Application Security strategy and roadmap based on application security metrics and KPIs?	No		No		Yes, but review is ad-hoc		Yes, we review it at regular times		Yes, we review it at least annually	
		Policy & Compliance	Answer	Rating	Answer	Rating	Answer	Rating	Answer	Rating	Answer	Rating
Policy & Standards	1	Do you have and apply a common set of policies and standards throughout your organization?	Yes, for most or all of the applications	1,13	Yes, for most or all of the applications	1,38	Yes, for most or all of the applications	1,63	Yes, for most or all of the applications	2,00	Yes, for most or all of the applications	2,50
	2	Do you publish the organization's policies as test scripts or run-books for easy interpretation by development teams?	Yes, some content		Yes, some content		Yes, some content		Yes, at least half of the content		Yes, most or all of the content	
	3	Do you regularly report on policy and standard compliance, and use that information to guide compliance improvement efforts?	No		No		Yes, but reporting is ad-hoc		Yes, we report at regular times		Yes, we report at regular times	
Compliance Management	1	Do you have a complete picture of your external compliance obligations?	Yes, for most or all of the applications		Yes, for most or all of the applications		Yes, for most or all of the applications		Yes, for most or all of the applications		Yes, for most or all of the applications	
	2	Do you have a standard set of security requirements and verification procedures addressing the organization's external compliance obligations?	No		Yes, for some obligations		Yes, for at least half of the obligations		Yes, for at least half of the obligations		Yes, for most or all of the obligations	
	3	Do you regularly report on adherence to external compliance obligations and use that information to guide efforts to close compliance gaps?	No		Yes, but reporting is ad-hoc		Yes, but reporting is ad-hoc		Yes, we report at regular times		Yes, we report at regular times	
		Education & Guidance	Answer	Rating	Answer	Rating	Answer	Rating	Answer	Rating	Answer	Rating
Training and Awareness	1	Do you require employees involved with application development to take SDLC training?	Yes, at least half of them	0,38	Yes, at least half of them	0,63	Yes, at least half of them	0,88	Yes, at least half of them	1,00	Yes, most or all of them	1,88
	2	Is training customized for individual roles such as developers, testers, or security champions?	Yes, for some of the training		Yes, for at least half of the training		Yes, for at least half of the training		Yes, for at least half of the training		Yes, for at least half of the training	
	3	Have you implemented a Learning Management System or equivalent to track employee training and certification processes?	No		No		Yes, for some of the training		Yes, for some of the training		Yes, for at least half of the training	
Organization and Culture	1	Have you identified a Security Champion for each development team?	No		Yes, for some teams		Yes, for some teams		Yes, for some teams		Yes, for most or all of the teams	
	2	Does the organization have a Secure Software Center of Excellence (SSCE)?	No		No		Yes, we started implementing it		Yes, for part of the organization		Yes, for part of the organization	
	3	Is there a centralized portal where developers and application security professionals from different teams and business units are able to communicate and share information?	No		No		No		No		Yes, we started implementing it	

Design	Current		Phase I		Phase II		Phase III		Phase IV	
Stream	Level	Threat Assessment	Answer	Rating	Answer	Rating	Answer	Rating	Answer	Rating	Answer	Rating
Application Risk Profile	1	Do you classify applications according to business risk based on a simple and predefined set of questions?	Yes, some of them	1,38	Yes, some of them	1,50	Yes, at least half of them	1,63	Yes, most or all of them	2,13	Yes, most or all of them	2,63
	2	Do you use centralized and quantified application risk profiles to evaluate business risk?	Yes, for most or all of the applications		Yes, for most or all of the applications		Yes, for most or all of the applications		Yes, for most or all of the applications		Yes, for most or all of the applications	
	3	Do you regularly review and update the risk profiles for your applications?	Yes, at least annually		Yes, at least annually		Yes, at least annually		Yes, at least annually		Yes, at least annually	
Threat Modeling	1	Do you identify and manage architectural design flaws with threat modeling?	Yes, some of them		Yes, some of them		Yes, some of them		Yes, at least half of them		Yes, most or all of them	
	2	Do you use a standard methodology, aligned on your application risk levels?	No		Yes, for some applications		Yes, for some applications		Yes, for at least half of the applications		Yes, for most or all of the applications	
	3	Do you regularly review and update the threat modeling methodology for your applications?	Yes, but review is ad-hoc		Yes, but review is ad-hoc		Yes, but review is ad-hoc		Yes, but review is ad-hoc		Yes, but review is ad-hoc	
		Security Requirements	Answer	Rating	Answer	Rating	Answer	Rating	Answer	Rating	Answer	Rating
Software Requirements	1	Do project teams specify security requirements during development?	Yes, for at least half of the applications	0,75	Yes, for most or all of the applications	1,00	Yes, for most or all of the applications	1,38	Yes, for most or all of the applications	1,88	Yes, for most or all of the applications	2,25
	2	Do you define, structure, and include prioritization in the artifacts of the security requirements gathering process?	No		No		Yes, some of the time		Yes, some of the time		Yes, at least half of the time	
	3	Do you use a standard requirements framework to streamline the elicitation of security requirements?	Yes, for some applications		Yes, for some applications		Yes, for at least half of the applications		Yes, for at least half of the applications		Yes, for most or all of the applications	
Supplier Security	1	Do stakeholders review vendor collaborations for security requirements and methodology?	No		No		No		No		No	
	2	Do vendors meet the security responsibilities and quality measures of service level agreements defined by the organization?	Yes, some of the time		Yes, some of the time		Yes, at least half of the time		Yes, most or all of the time		Yes, most or all of the time	
	3	Are vendors aligned with standard security controls and software development tools and processes that the organization utilizes?	Yes, at least half of the time		Yes, at least half of the time		Yes, at least half of the time		Yes, most or all of the time		Yes, most or all of the time	
		Secure Architecture	Answer	Rating	Answer	Rating	Answer	Rating	Answer	Rating	Answer	Rating
Architecture Design	1	Do teams use security principles during design?	No	0,13	Yes, for some applications	0,50	Yes, for some applications	0,88	Yes, for at least half of the applications	1,13	Yes, for at least half of the applications	1,75
	2	Do you use shared security services during design?	No		Yes, for some applications		Yes, for at least half of the applications		Yes, for at least half of the applications		Yes, for most or all of the applications	
	3	Do you base your design on available reference architectures?	No		Yes, for some applications		Yes, for some applications		Yes, for at least half of the applications		Yes, for most or all of the applications	
Technology Management	1	Do you evaluate the security quality of important technologies used for development?	Yes, for some applications		Yes, for some applications		Yes, for some applications		Yes, for some applications		Yes, for some applications	
	2	Do you have a list of recommended technologies for the organization?	No		No		Yes, for some of the technology domains		Yes, for some of the technology domains		Yes, for at least half of the technology domains	
	3	Do you enforce the use of recommended technologies within the organization?	No		No		Yes, for some applications		Yes, for some applications		Yes, for some applications	

Implementation	Current		Phase I		Phase II		Phase III		Phase IV	
Stream	Level	Secure Build	Answer	Rating	Answer	Rating	Answer	Rating	Answer	Rating	Answer	Rating
Build Process	1	Is your full build process formally described?	Yes, for most or all of the applications	1,75	Yes, for most or all of the applications	1,75	Yes, for most or all of the applications	1,88	Yes, for most or all of the applications	2,00	Yes, for most or all of the applications	2,25
	2	Is the build process fully automated?	Yes, for most or all of the applications		Yes, for most or all of the applications		Yes, for most or all of the applications		Yes, for most or all of the applications		Yes, for most or all of the applications	
	3	Do you enforce automated security checks in your build processes?	Yes, for some applications		Yes, for some applications		Yes, for some applications		Yes, for at least half of the applications		Yes, for most or all of the applications	
Software Dependencies	1	Do you have solid knowledge about dependencies you're relying on?	Yes, for some applications		Yes, for some applications		Yes, for some applications		Yes, for some applications		Yes, for some applications	
	2	Do you handle 3rd party dependency risk by a formal process?	No		No		Yes, for some applications		Yes, for some applications		Yes, for some applications	
	3	Do you prevent build of software if it's affected by vulnerabilities in dependencies?	Yes, for most or all of the applications		Yes, for most or all of the applications		Yes, for most or all of the applications		Yes, for most or all of the applications		Yes, for most or all of the applications	
		Secure Deployment	Answer	Rating	Answer	Rating	Answer	Rating	Answer	Rating	Answer	Rating
Deployment Process	1	Do you use repeatable deployment processes?	Yes, for most or all of the applications	1,00	Yes, for most or all of the applications	1,13	Yes, for most or all of the applications	1,63	Yes, for most or all of the applications	2,00	Yes, for most or all of the applications	2,63
	2	Are deployment processes automated and employing security checks?	No		No		Yes, for some applications		Yes, for at least half of the applications		Yes, for most or all of the applications	
	3	Do you consistently validate the integrity of deployed artifacts?	No		No		No		No		Yes, for some applications	
Secret Management	1	Do you limit access to application secrets according to the least privilege principle?	Yes, for most or all of the applications		Yes, for most or all of the applications		Yes, for most or all of the applications		Yes, for most or all of the applications		Yes, for most or all of the applications	
	2	Do you inject production secrets into configuration files during deployment?	No		Yes, for some applications		Yes, for at least half of the applications		Yes, for at least half of the applications		Yes, for most or all of the applications	
	3	Do you practice proper lifecycle management for application secrets?	No		No		Yes, for at least half of the applications		Yes, for most or all of the applications		Yes, for most or all of the applications	
		Defect Management	Answer	Rating	Answer	Rating	Answer	Rating	Answer	Rating	Answer	Rating
Defect Tracking	1	Do you track all known security defects in accessible locations?	No	0,13	Yes, for some applications	0,25	Yes, for some applications	0,38	Yes, for at least half of the applications	0,88	Yes, for at least half of the applications	1,13
	2	Do you keep an overview of the state of security defects across the organization?	No		No		No		Yes, for some applications		Yes, for at least half of the applications	
	3	Do you enforce SLAs for fixing security defects?	No		No		No		Yes, for some applications		Yes, for at least half of the applications	
Metrics and Feedback	1	Do you use basic metrics about recorded security defects to carry out quick win improvement activities?	Yes, for some applications		Yes, for some applications		Yes, for some applications		Yes, for some applications		Yes, for some applications	
	2	Do you improve your security assurance program upon standardized metrics?	No		No		Yes, for some applications		Yes, for some applications		Yes, for some applications	
	3	Do you regularly evaluate the effectiveness of your security metrics so that its input helps drive your security strategy?	No		No		No		Yes, for some applications		Yes, for some applications	

Verification	Current		Phase I		Phase II		Phase III		Phase IV	
Stream	Level	Architecture Assessment	Answer	Rating	Answer	Rating	Answer	Rating	Answer	Rating	Answer	Rating
Architecture Validation	1	Do you review the application architecture for key security objectives on an ad-hoc basis?	No	0,00	Yes, for some applications	0,25	Yes, for some applications	0,63	Yes, for some applications	1,00	Yes, for at least half of the applications	1,38
	2	Do you regularly review the security mechanisms of your architecture?	No		No		Yes, for some applications		Yes, for at least half of the applications		Yes, for at least half of the applications	
	3	Do you regularly review the effectiveness of the security controls?	No		No		Yes, for some applications		Yes, for at least half of the applications		Yes, for at least half of the applications	
Architecture Mitigation	1	Do you review the application architecture for mitigations of typical threats on an ad-hoc basis?	No		Yes, for some applications		Yes, for some applications		Yes, for at least half of the applications		Yes, for most or all of the applications	
	2	Do you regularly evaluate the threats to your architecture?	No		No		Yes, for some applications		Yes, for some applications		Yes, for some applications	
	3	Do you regularly update your reference architectures based on architecture assessment findings?	No		No		No		No		No	
		Requirements Testing	Answer	Rating	Answer	Rating	Answer	Rating	Answer	Rating	Answer	Rating
Control Verification	1	Do you test applications for the correct functioning of standard security controls?	Yes, most or all of them	1,50	Yes, most or all of them	1,50	Yes, most or all of them	1,50	Yes, most or all of them	1,50	Yes, most or all of them	1,88
	2	Do you consistently write and execute test scripts to verify the functionality of security requirements?	Yes, most or all of them		Yes, most or all of them		Yes, most or all of them		Yes, most or all of them		Yes, most or all of them	
	3	Do you automatically test applications for security regressions?	Yes, for most or all of the applications		Yes, for most or all of the applications		Yes, for most or all of the applications		Yes, for most or all of the applications		Yes, for most or all of the applications	
Misuse/Abuse Testing	1	Do you test applications using randomization or fuzzing techniques?	No		No		No		No		Yes, for some applications	
	2	Do you create abuse cases from functional requirements and use them to drive security tests?	No		No		No		No		Yes, some of the time	
	3	Do you perform denial of service and security stress testing?	No		No		No		No		Yes, some of the time	
		Security Testing	Answer	Rating	Answer	Rating	Answer	Rating	Answer	Rating	Answer	Rating
Scalable Baseline	1	Do you scan applications with automated security testing tools?	No	0,38	Yes, some of them	0,50	Yes, some of them	0,88	Yes, some of them	1,13	Yes, at least half of them	1,50
	2	Do you customize the automated security tools to your applications and technology stacks?	No		No		Yes, some of them		Yes, some of them		Yes, some of them	
	3	Do you integrate automated security testing into the build and deploy process?	No		No		Yes, some of it		Yes, at least half of it		Yes, at least half of it	
Deep Understanding	1	Do you manually review the security quality of selected high-risk components?	Yes, for at least half of the components		Yes, for at least half of the components		Yes, for at least half of the components		Yes, for at least half of the components		Yes, for most or all of the components	
	2	Do you perform penetration testing for your applications at regular intervals?	Yes, for some applications		Yes, for some applications		Yes, for at least half of the applications		Yes, for at least half of the applications		Yes, for at least half of the applications	
	3	Do you use the results of security testing to improve the development lifecycle?	No		No		No		Yes, but we improve it ad-hoc		Yes, but we improve it ad-hoc	

Operations	Current		Phase I		Phase II		Phase III		Phase IV	
Stream	Level	Incident Management	Answer	Rating	Answer	Rating	Answer	Rating	Answer	Rating	Answer	Rating
Incident Detection	1	Do you analyze log data for security incidents periodically?	Yes, for most or all of the applications	2,00	Yes, for most or all of the applications	2,13	Yes, for most or all of the applications	2,25	Yes, for most or all of the applications	2,75	Yes, for most or all of the applications	3,00
	2	Do you follow a documented process for incident detection?	No		Yes, for some applications		Yes, for some applications		Yes, for most or all of the applications		Yes, for most or all of the applications	
	3	Do you review and update the incident detection process regularly?	No		No		Yes, for some applications		Yes, for at least half of the applications		Yes, for most or all of the applications	
Incident Response	1	Do you respond to detected incidents?	Yes, for most or all of the incidents		Yes, for most or all of the incidents		Yes, for most or all of the incidents		Yes, for most or all of the incidents		Yes, for most or all of the incidents	
	2	Do you use a repeatable process for incident handling?	Yes, for most or all of the incident types		Yes, for most or all of the incident types		Yes, for most or all of the incident types		Yes, for most or all of the incident types		Yes, for most or all of the incident types	
	3	Do you have a dedicated incident response team available?	Yes, most or all of the time		Yes, most or all of the time		Yes, most or all of the time		Yes, most or all of the time		Yes, most or all of the time	
		Environment Management	Answer	Rating	Answer	Rating	Answer	Rating	Answer	Rating	Answer	Rating
Configuration Hardening	1	Do you harden configurations for key components of your technology stacks?	Yes, for most or all of the components	1,50	Yes, for most or all of the components	1,63	Yes, for most or all of the components	1,75	Yes, for most or all of the components	1,88	Yes, for most or all of the components	2,00
	2	Do you have hardening baselines for your components?	No		Yes, for some components		Yes, for some components		Yes, for some components		Yes, for some components	
	3	Do you monitor and enforce conformity with hardening baselines?	No		No		Yes, for some components		Yes, for some components		Yes, for some components	
Patching and Updating	1	Do you identify and patch vulnerable components?	Yes, for most or all of the components		Yes, for most or all of the components		Yes, for most or all of the components		Yes, for most or all of the components		Yes, for most or all of the components	
	2	Do you follow an established process for updating components of your technology stacks?	Yes, for most or all of the components		Yes, for most or all of the components		Yes, for most or all of the components		Yes, for most or all of the components		Yes, for most or all of the components	
	3	Do you regularly evaluate components and review patch level status?	No		No		No		Yes, for some components		Yes, for at least half of the components	
		Operational Management	Answer	Rating	Answer	Rating	Answer	Rating	Answer	Rating	Answer	Rating
Data Protection	1	Do you protect and handle information according to protection requirements for data stored and processed on each application?	Yes, for most or all of the applications	1,63	Yes, for most or all of the applications	1,88	Yes, for most or all of the applications	2,00	Yes, for most or all of the applications	2,13	Yes, for most or all of the applications	2,38
	2	Do you maintain a data catalog, including types, sensitivity levels, and processing and storage locations?	Yes, for most or all of our data		Yes, for most or all of our data		Yes, for most or all of our data		Yes, for most or all of our data		Yes, for most or all of our data	
	3	Do you regularly review and update the data catalog and your data protection policies and procedures?	No		Yes, we do it when requested		Yes, we do it every few years		Yes, we do it every few years		Yes, we do it every few years	
System Decommissioning / Legacy Management	1	Do you identify and remove systems, applications, application dependencies, or services that are no longer used, have reached end of life, or are no longer actively developed or supported?	Yes, for some applications		Yes, for some applications		Yes, for some applications		Yes, for at least half of the applications		Yes, for most or all of the applications	
	2	Do you follow an established process for removing all associated resources, as part of decommissioning of unused systems, applications, application dependencies, or services?	No		Yes, some of the time		Yes, some of the time		Yes, some of the time		Yes, some of the time	
	3	Do you regularly evaluate the lifecycle state and support status of every software asset and underlying infrastructure component, and estimate their end of life?	Yes, for most or all of the assets		Yes, for most or all of the assets		Yes, for most or all of the assets		Yes, for most or all of the assets		Yes, for most or all of the assets	

Figura 30. Planificación hoja de ruta con herramienta SAMM Assessment Tool 2.0

SAMM Assessment Scorecard: Departamento Desarrollo For TFM SAMM UOC

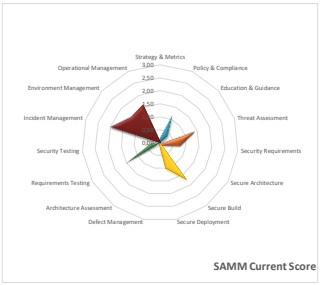
Organization: TFM SAMM UOC
Team/Application: Departamento Desarrollo
Interview Date: 20/05/2021
Team Lead: Rafa Medina
Contributors: Otros

Current Maturity Score

Functions	Security Practices	Current	Maturity 1	2	3
Governance	Strategy & Metrics	0,13	0,13	0,00	0,00
Governance	Policy & Compliance	1,13	1,00	0,13	0,00
Governance	Education & Guidance	0,38	0,25	0,13	0,00
Design	Threat Assessment	1,38	0,25	0,50	0,63
Design	Security Requirements	0,75	0,25	0,13	0,38
Design	Secure Architecture	0,13	0,13	0,00	0,00
Implementation	Secure Build	1,75	0,63	0,50	0,63
Implementation	Secure Deployment	1,00	1,00	0,00	0,00
Implementation	Defect Management	0,13	0,13	0,00	0,00
Verification	Architecture Assessment	0,00	0,00	0,00	0,00
Verification	Requirements Testing	1,50	0,50	0,50	0,50
Verification	Security Testing	0,38	0,25	0,13	0,00
Operations	Incident Management	2,00	1,00	0,50	0,50
Operations	Environment Management	1,50	1,00	0,50	0,00
Operations	Operational Management	1,63	0,63	0,50	0,50

Functions	Current
Governance	0,54
Design	0,75
Implementation	0,96
Verification	0,63
Operations	1,71

Current Maturity Score



Current Maturity Score

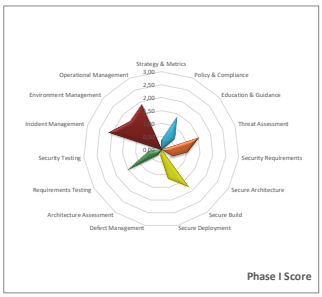
		Governance Design		Implementati Verification		Operations
Governance	Strategy & Metrics	0,13	0,00	0,00	0,00	0,00
Governance	Policy & Compliance	1,13	0,00	0,00	0,00	0,00
Governance	Education & Guidance	0,38	0,00	0,00	0,00	0,00
Design	Threat Assessment	0,00	1,38	0,00	0,00	0,00
Design	Security Requirements	0,00	0,75	0,00	0,00	0,00
Design	Secure Architecture	0,00	0,13	0,00	0,00	0,00
Implementation	Secure Build	0,00	0,00	1,75	0,00	0,00
Implementation	Secure Deployment	0,00	0,00	1,00	0,00	0,00
Implementation	Defect Management	0,00	0,00	0,13	0,00	0,00
Verification	Architecture Assessment	0,00	0,00	0,00	0,00	0,00
Verification	Requirements Testing	0,00	0,00	0,00	1,50	0,00
Verification	Security Testing	0,00	0,00	0,00	0,38	0,00
Operations	Incident Management	0,00	0,00	0,00	0,00	2,00
Operations	Environment Management	0,00	0,00	0,00	0,00	1,50
Operations	Operational Management	0,00	0,00	0,00	0,00	1,63

Phase 1 Maturity Score

Business Functions	Security Practices	Current	Maturity 1	2	3
Governance	Strategy & Metrics	0,38	0,25	0,13	0,00
Governance	Policy & Compliance	1,38	1,00	0,25	0,13
Governance	Education & Guidance	0,63	0,38	0,25	0,00
Design	Threat Assessment	1,50	0,25	0,63	0,63
Design	Security Requirements	1,00	0,50	0,13	0,38
Design	Secure Architecture	0,50	0,25	0,13	0,13
Implementation	Secure Build	1,75	0,63	0,50	0,63
Implementation	Secure Deployment	1,13	1,00	0,13	0,00
Implementation	Defect Management	0,25	0,25	0,00	0,00
Verification	Architecture Assessment	0,25	0,25	0,00	0,00
Verification	Requirements Testing	1,50	0,50	0,50	0,50
Verification	Security Testing	0,50	0,38	0,13	0,00
Operations	Incident Management	2,13	1,00	0,63	0,50
Operations	Environment Management	1,63	1,00	0,63	0,00
Operations	Operational Management	1,88	0,63	0,63	0,63

Business Functions	Current
Governance	0,79
Design	1,00
Implementation	1,04
Verification	0,75
Operations	1,88

Phase 1 Maturity Score



Phase 1 Maturity Score

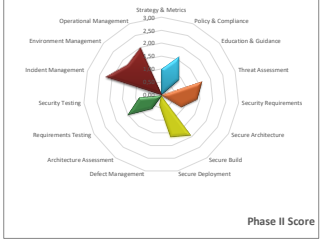
		Governance Design		Implementati Verification		Operations
Governance	Strategy & Metrics	0,38	0,00	0,00	0,00	0,00
Governance	Policy & Compliance	1,38	0,00	0,00	0,00	0,00
Governance	Education & Guidance	0,63	0,00	0,00	0,00	0,00
Design	Threat Assessment	0,00	1,50	0,00	0,00	0,00
Design	Security Requirements	0,00	1,00	0,00	0,00	0,00
Design	Secure Architecture	0,00	0,50	0,00	0,00	0,00
Implementation	Secure Build	0,00	0,00	1,75	0,00	0,00
Implementation	Secure Deployment	0,00	0,00	1,13	0,00	0,00
Implementation	Defect Management	0,00	0,00	0,25	0,00	0,00
Verification	Architecture Assessment	0,00	0,00	0,00	0,25	0,00
Verification	Requirements Testing	0,00	0,00	0,00	1,50	0,00
Verification	Security Testing	0,00	0,00	0,00	0,50	0,00
Operations	Incident Management	0,00	0,00	0,00	0,00	2,13
Operations	Environment Management	0,00	0,00	0,00	0,00	1,63
Operations	Operational Management	0,00	0,00	0,00	0,00	1,88

Phase 2 Maturity Score

Business Functions	Security Practices	Current	Maturity 1	2	3
Governance	Strategy & Metrics	1,00	0,25	0,38	0,38
Governance	Policy & Compliance	1,63	1,00	0,38	0,25
Governance	Education & Guidance	0,88	0,38	0,38	0,13
Design	Threat Assessment	1,63	0,38	0,63	0,63
Design	Security Requirements	1,38	0,50	0,38	0,50
Design	Secure Architecture	0,88	0,25	0,38	0,25
Implementation	Secure Build	1,88	0,63	0,63	0,63
Implementation	Secure Deployment	1,63	1,00	0,38	0,25
Implementation	Defect Management	0,38	0,25	0,13	0,00
Verification	Architecture Assessment	0,63	0,25	0,25	0,13
Verification	Requirements Testing	1,50	0,50	0,50	0,50
Verification	Security Testing	0,88	0,38	0,38	0,13
Operations	Incident Management	2,25	1,00	0,63	0,63
Operations	Environment Management	1,75	1,00	0,63	0,13
Operations	Operational Management	2,00	0,63	0,63	0,75

Business Functions	Current
Governance	1,17
Design	1,29
Implementation	1,29
Verification	1,00
Operations	2,00

Phase 2 Maturity Score



Phase 2 Maturity Score

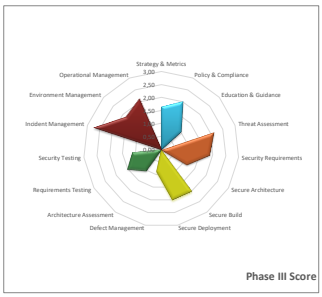
		Governance Design		Implementati Verification		Operations
Governance	Strategy & Metrics	1,00	0,00	0,00	0,00	0,00
Governance	Policy & Compliance	1,63	0,00	0,00	0,00	0,00
Governance	Education & Guidance	0,88	0,00	0,00	0,00	0,00
Design	Threat Assessment	0,00	1,63	0,00	0,00	0,00
Design	Security Requirements	0,00	1,38	0,00	0,00	0,00
Design	Secure Architecture	0,00	0,88	0,00	0,00	0,00
Implementation	Secure Build	0,00	0,00	1,88	0,00	0,00
Implementation	Secure Deployment	0,00	0,00	1,63	0,00	0,00
Implementation	Defect Management	0,00	0,00	0,38	0,00	0,00
Verification	Architecture Assessment	0,00	0,00	0,00	0,63	0,00
Verification	Requirements Testing	0,00	0,00	0,00	1,50	0,00
Verification	Security Testing	0,00	0,00	0,00	0,88	0,00
Operations	Incident Management	0,00	0,00	0,00	0,00	2,25
Operations	Environment Management	0,00	0,00	0,00	0,00	1,75
Operations	Operational Management	0,00	0,00	0,00	0,00	2,00

Phase 3 Maturity Score

Business Functions	Security Practices	Current	Maturity 1	2	3
Governance	Strategy & Metrics	1,63	0,38	0,50	0,75
Governance	Policy & Compliance	2,00	1,00	0,50	0,50
Governance	Education & Guidance	1,00	0,38	0,50	0,13
Design	Threat Assessment	2,13	0,75	0,75	0,63
Design	Security Requirements	1,88	0,50	0,63	0,75
Design	Secure Architecture	1,13	0,38	0,38	0,38
Implementation	Secure Build	2,00	0,63	0,63	0,75
Implementation	Secure Deployment	2,00	1,00	0,50	0,50
Implementation	Defect Management	0,88	0,38	0,25	0,25
Verification	Architecture Assessment	1,00	0,38	0,38	0,25
Verification	Requirements Testing	1,50	0,50	0,50	0,50
Verification	Security Testing	1,13	0,38	0,38	0,38
Operations	Incident Management	2,75	1,00	1,00	0,75
Operations	Environment Management	1,88	1,00	0,63	0,25
Operations	Operational Management	2,13	0,75	0,63	0,75

Business Functions	Current
Governance	1,54
Design	1,71
Implementation	1,63
Verification	1,21
Operations	2,25

Phase 3 Maturity Score



Phase 3 Maturity Score

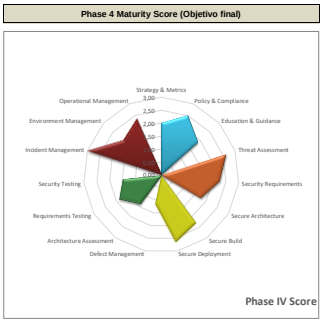
		Governance Design		Implementati Verification		Operations
Governance	Strategy & Metrics	1,63	0,00	0,00	0,00	0,00
Governance	Policy & Compliance	2,00	0,00	0,00	0,00	0,00
Governance	Education & Guidance	1,00	0,00	0,00	0,00	0,00
Design	Threat Assessment	0,00	2,13	0,00	0,00	0,00
Design	Security Requirements	0,00	1,88	0,00	0,00	0,00
Design	Secure Architecture	0,00	1,13	0,00	0,00	0,00
Implementation	Secure Build	0,00	0,00	2,00	0,00	0,00
Implementation	Secure Deployment	0,00	0,00	2,00	0,00	0,00
Implementation	Defect Management	0,00	0,00	0,88	0,00	0,00
Verification	Architecture Assessment	0,00	0,00	0,00	1,00	0,00
Verification	Requirements Testing	0,00	0,00	0,00	1,50	0,00
Verification	Security Testing	0,00	0,00	0,00	1,13	0,00
Operations	Incident Management	0,00	0,00	0,00	0,00	2,75
Operations	Environment Management	0,00	0,00	0,00	0,00	1,88
Operations	Operational Management	0,00	0,00	0,00	0,00	2,13

Phase 4 Maturity Score (Objetivo final)

Business Functions	Security Practices	Current	Maturity 1	2	3
Governance	Strategy & Metrics	2,00	0,50	0,50	1,00
Governance	Policy & Compliance	2,50	1,00	1,00	0,50
Governance	Education & Guidance	1,88	1,00	0,50	0,38
Design	Threat Assessment	2,63	1,00	1,00	0,63
Design	Security Requirements	2,25	0,50	0,75	1,00
Design	Secure Architecture	1,75	0,38	0,75	0,63
Implementation	Secure Build	2,25	0,63	0,63	1,00
Implementation	Secure Deployment	2,63	1,00	1,00	0,63
Implementation	Defect Management	1,13	0,38	0,38	0,38
Verification	Architecture Assessment	1,38	0,75	0,38	0,25
Verification	Requirements Testing	1,88	0,63	0,63	0,63
Verification	Security Testing	1,50	0,75	0,38	0,38
Operations	Incident Management	3,00	1,00	1,00	1,00
Operations	Environment Management	2,00	1,00	0,63	0,38
Operations	Operational Management	2,38	1,00	0,63	0,75

Business Functions	Current
Governance	2,13
Design	2,21
Implementation	2,00
Verification	1,58
Operations	2,46

Phase 4 Maturity Score (Objetivo final)



Phase 4 Maturity Score (Objetivo final)

		Governance Design		Implementati Verification		Operations
Governance	Strategy & Metrics	2,00	0,00	0,00	0,00	0,00
Governance	Policy & Compliance	2,50	0,00	0,00	0,00	0,00
Governance	Education & Guidance	1,88	0,00	0,00	0,00	0,00
Design	Threat Assessment	0,00	2,63	0,00	0,00	0,00
Design	Security Requirements	0,00	2,25	0,00	0,00	0,00
Design	Secure Architecture	0,00	1,75	0,00	0,00	0,00
Implementation	Secure Build	0,00	0,00	2,25	0,00	0,00
Implementation	Secure Deployment	0,00	0,00	2,63	0,00	0,00
Implementation	Defect Management	0,00	0,00	1,13	0,00	0,00
Verification	Architecture Assessment	0,00	0,00	0,00	1,38	0,00
Verification	Requirements Testing	0,00	0,00	0,00	1,88	0,00
Verification	Security Testing	0,00	0,00	0,00	1,50	0,00
Operations	Incident Management	0,00	0,00	0,00	0,00	3,00
Operations	Environment Management	0,00	0,00	0,00	0,00	2,00
Operations	Operational Management	0,00	0,00	0,00	0,00	2,38

Figura 31. Evolución de nivel de madurez a lo largo de las etapas de la hoja de ruta con SAMM Assessment Tool 2.0

5.6 Implementar

Una vez planificadas las acciones, estas se ejecutarán tal y como se procedería con cualquier otro proyecto. Es decir, se asignarán recursos y responsabilidades, se realizará seguimiento sobre el progreso, detección de cuellos de botella, etc.

Se ejecutarán las actividades definidas para cada una de las fases, teniendo en consideración el impacto sobre los procesos, personas, conocimientos y herramientas. Para ayudar a lidiar con todo esto, se tendrán en cuenta las prescripciones y consejos prácticos que el propio SAMM ofrece en la descripción de las prácticas y actividades de seguridad. Otros proyectos de OWASP también servirán de especial ayuda para implementar muchas de las acciones.

Durante esta fase de implementación, SAMM recomienda tratar aparte el software *legacy*, evitando las actuaciones sobre el mismo a menos que sea realmente importante.

5.7 Puesta en marcha del programa de seguridad

Tras la finalización de proyecto de mejora de la seguridad en el software, el siguiente paso consiste en la puesta en marcha efectiva del programa de seguridad del software. Esto consiste en garantizar que los nuevos procesos, actividades y herramientas queden disponibles y se utilicen de forma efectiva dentro de la organización.

Para lograr la puesta en marcha efectiva, se comunicará y formará a todas las partes interesadas sobre los nuevos procesos y mejoras conseguidas, asegurando su visibilidad dentro de la organización y fomentando su utilización. Para tal fin, se requerirá con la colaboración de los denominados "*Security Champions*". Inicialmente, la organización se centrará sobre las aplicaciones más relevantes o de alto impacto.

Además de publicitar y difundir, se comenzará a medir el grado de adopción y eficacia de los nuevos procesos y actividades implementadas, con el fin de evaluar el éxito de la puesta en marcha del programa y llevar a cabo acciones correctivas en caso necesario.

6 Gobierno de la seguridad en el desarrollo de software

En el apartado anterior, se ha tratado la forma de establecer e iniciar un programa de seguridad en el software basado en el modelo SAMM. La puesta en marcha del programa se ha abordado a través de un proyecto de implantación, dividido en diversas etapas de aproximación sucesiva que hacen más manejable el proyecto.

Una vez puesto en marcha el programa de seguridad, este deberá ser gestionado para garantizar su éxito. Es decir, se realizará seguimiento, se evaluarán los resultados y se trabajará en la mejora del mismo siguiendo un ciclo PDCA.

Como ya se comentó anteriormente, "*Lo que no se define no se puede medir. Lo que no se mide, no se puede mejorar. Lo que no se mejora, se degrada siempre*". Por tanto, para gestionar el programa de aseguramiento del software será preciso medir.

Tal y como se comentó en la práctica de Estrategia y Métricas, se deberán establecer al menos tres tipos de métricas (ver apartado 4.2.1.1.1):

- Esfuerzo: horas de formación, el tiempo dedicado a la revisión del código, el número de aplicaciones analizadas en busca de vulnerabilidades, etc.
- Resultados: defectos de seguridad sin parchear, número de incidentes de seguridad, etc.
- Entorno: número de aplicaciones, número de líneas de código, etc.

Sobre la base de las métricas recopiladas y otros factores como la evolución de los riesgos y necesidades de negocio de la organización, se establecerán los correspondientes planes de mejora del programa de seguridad. Es decir, una vez puesto en marcha el programa de seguridad del software, este irá evolucionando con el fin de aumentar la eficacia de los procesos y adaptarse a las necesidades cambiantes de la organización.

Para la gestión de la mejora del programa de seguridad, se podrá utilizar la aplicación web proporcionada por el grupo de trabajo de SAMM [25]. Mediante esta herramienta una organización puede gestionar los planes de mejora y controlar la evolución del programa de seguridad en el tiempo (Figura 32).

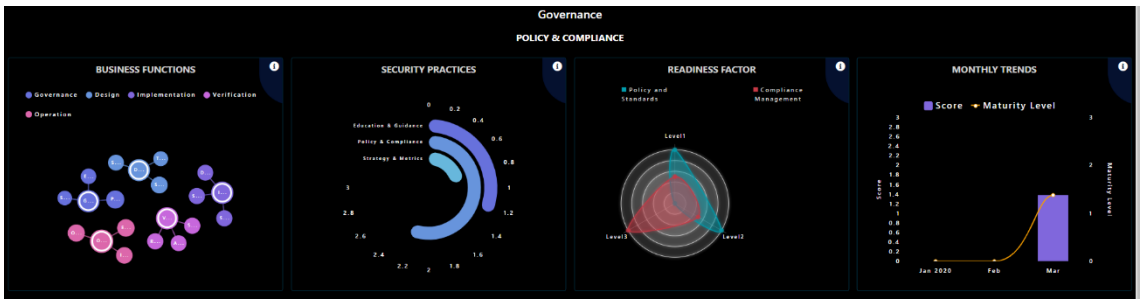


Figura 32. Cuadro de mando de seguimiento y evolución. Fuente: OWASP SAMM

7 Soluciones y herramientas aplicables

En este capítulo se enumerarán algunas herramientas, utilidades y referencias que pueden servir de gran ayuda para dar cumplimiento a diferentes actividades de seguridad del modelo. De forma prioritaria, nos enfocaremos en herramientas open source. Estas referencias se proponen como una alternativa de solución, si bien pueden existir otros productos y herramientas que también podrían constituir una buena alternativa para ello.

7.1 Práctica de Estrategia y Métricas: herramientas

7.1.1 Crear y promover programa de seguridad de las aplicaciones (Stream A)

En lo que respecta a esta línea de acción, se relaciona fuertemente con la identificación de activos, riesgos y planes de gestión o mitigación. A partir de estos elementos, se definirá la estrategia de seguridad de la organización.

Las organizaciones ya suelen disponer de sistemas para gestionar estos aspectos. En otros muchos casos la documentación se soporta sobre hojas de cálculo, documentos con planes, etc. Por ello, no parece muy necesario incidir sobre herramientas para tratar los aspectos relacionados con la identificación de la postura ante el riesgo de las organizaciones o los planes de seguridad asociados. No obstante, se citará una herramienta gratuita que puede ayudar a empezar en la práctica de definir riesgos, hacer seguimiento de los mismos, gestionar mitigaciones, visualizar los riesgos por criticidad, etc.

Herramientas

- ***SimpleRisk***. SimpleRisk es una herramienta que además de para la gestión de riesgos, también permite la gestión integrada de políticas, estándares y aspectos de cumplimiento normativo. Aunque existen diversos módulos añadidos de pago, la se puede descargar gratuitamente una versión básica con la que iniciar las prácticas de gobierno. URL: https://www.simplerisk.com/

7.1.2 Medir y mejorar el programa de seguridad (Stream B)

En relación a la esta línea de acción de otra línea de acción de la práctica, la relacionada con el uso de métricas para la mejora del programa de seguridad, hay que tener en cuenta que la definición y captura de métricas es algo muy particular y específico de cada organización. No obstante, se referenciarán algunas herramientas que pueden facilitar el proceso de recopilación de métricas relacionadas con la seguridad del software y la presentación de estos datos.

Herramientas

- ***InfluxDB***. InfluxDB es una base de datos para el almacenamiento y consulta de series temporales. Está diseñada para el almacenamiento y la recuperación rápidos de datos de series temporales provenientes de monitorización de operaciones, las métricas de aplicaciones, etc. Obviamente, para el almacenamiento de las métricas del software no se precisan sistemas que requieran de grandes capacidades, pero puede ser

una solución para no tener que diseñar un sistema específico. URL: https://www.influxdata.com/

- ***TimescaleDB***. TimescaleDB es una base de datos relacional para el almacenamiento y consulta de series temporales. Está construida sobre la base de PostgreSQL. URL: https://www.timescale.com/
- ***Grafana***. Grafana es un software libre para la visualización y el formateo de datos provenientes de métricas. Permite crear cuadros de mando y gráficos a partir de diversas fuentes, incluidas bases de datos de series temporales. URL: https://grafana.com/

7.2 Práctica de Política y Cumplimiento

La práctica de Políticas y Cumplimiento (PC) se centra en comprender los requisitos legales y reglamentarios externos e impulsar normas y estándares de seguridad internas para garantizar el cumplimiento de todos los requisitos asociados en las aplicaciones.

Para la definición de políticas y estándares de seguridad existen diversos repositorios de documentación como los proporcionados por el NIST, CCN [26], etc. En el propio OWASP existe un proyecto denominado “OWASP Application Security Verification Standard” [27] que puede proporcionar gran ayuda para la definición de los estándares de seguridad en las aplicaciones.

Respecto a la parte de cumplimiento normativo, la organización tendrá que llevar a cabo de identificar los requerimientos que le apliquen en función de la industria o sector donde desarrolle la actividad, la ubicación geográfica, etc.

Para la publicación y mantenimiento de la documentación mencionada, la principal necesidad es disponer de un repositorio corporativo donde se publicarán las políticas y estándares de seguridad, así como la documentación asociada al cumplimiento normativo y las correspondientes matrices de cumplimiento. Por citar alguna herramienta, haremos referencia de nuevo a SimpleRisk, comentada anteriormente.

7.3 Práctica de Formación y Orientación

En este apartado, se proporcionarán referencias a utilidades que pueden servir para entrenar a los equipos de desarrollo. Entre estas, se incluyen herramientas específicas para que los equipos de desarrollo aprendan sobre seguridad en las aplicaciones y herramientas destinadas a que la organización construya su propio sistema de gestión para la formación (LMS).

Hay que tener en cuenta que la práctica del modelo también exige que se registre y controle la formación realizada por el personal de la organización. Esto deberá ser gestionado con las herramientas con las que ya cuente la organización para la gestión de la formación.

Herramientas

- OWASP Security Knowledge Framework. Security Knowledge Framework es una aplicación web que explica los principios de codificación segura en múltiples lenguajes de programación. Se puede emplear para que los desarrolladores adquieran conocimientos sobre seguridad del software y

aprendan a integrar la seguridad desde el inicio del proceso de desarrollo. Proporciona guías, checklist de verficación y laboratorios para practicar la verificación de la seguridad. URL: https://owasp.org/www-project-security-knowledge-framework/

- ***OWASP Juice Shop***. Juice Shop es una aplicación web insegura que puede ser usada para la formación en materia de seguridad, demostraciones, concienciación, y como aplicación de pruebas para entrenarse en el uso de herramientas de seguridad. La aplicación contiene todas las vulnerabilidades de OWASP Top 10, así como otros muchos fallos de seguridad encontrados en aplicaciones reales. URL: https://owasp.org/www-project-juice-shop/
- ***OWASP Security Shepherd***. Esta es una plataforma para formación en seguridad sobre aplicaciones web y móviles. Puede ser usada para la formación en materia de seguridad, demostraciones, concienciación y está pensada tanto para desarrolladores poco expertos en temas de seguridad como para personal experimentado. URL: https://owasp.org/www-project-security-shepherd/
- ***Damn Vulnerable iOS Application (DVIA)***. Damn Vulnerable iOS App (DVIA) es una aplicación para iOS que es deliberadamente vulnerable. Su principal objetivo es proporcionar una plataforma para poner a prueba las habilidades de pruebas de penetración de iOS. URL: http://damnvulnerableiosapp.com/
- ***OWASP Mutillidae II***. OWASP Multillidae II es una aplicación web gratuita, de código abierto y deliberadamente vulnerable que puede ser usada para la formación de seguridad de aplicaciones web. URL: https://github.com/webpwnized/mutillidae
- ***Moodle.*** Esta conocida plataforma está diseñado para crear y gestionar espacios de aprendizaje online adaptados a las necesidades particulares de cualquier organización. Permite gestionar evaluaciones, certificaciones, etc. URL: https://moodle.org/
- ***Chamilo***. Chamilo, al igual que Moodle, es una plataforma crear aulas virtuales y gestionar cursos de e-Learning. URL: https://chamilo.org/.

7.4 Práctica de Evaluación de Amenazas

Esta práctica trata sobre la evaluación del riesgo para cada una de las aplicaciones de la organización, así como el impacto que supondría para la organización la materialización de dichos riesgos. Adicionalmente, la práctica promueve el modelado de amenazas durante la fase de diseño del software con el fin de identificar vulnerabilidades y defectos lo antes posible.

Herramientas

- ***OWASP Risk Rating Methodology***. Metodología que se puede utilizar para valorar los riesgos de las aplicaciones de forma cuantitativa y sistemática. URL: https://wiki.owasp.org/index.php/OWASP_Risk_Rating_Methodology
- ***OWASP Threat Dragon***. Threat Dragon es una aplicación de modelado de amenazas gratuita, de código abierto y multiplataforma que permite la

elaboración de diagramas, y dispone de un motor de reglas para autogenerar amenazas y propuestas de mitigación. URL: https://github.com/OWASP/threat-dragon

- ***Microsoft Threat Modeling Tool (STRIDE)***. Esta herramienta facilita el modelado de amenazas. Al igual que Threat Dragon, permite elaborar diagramas y dispone de un motor de reglas para autogenerar amenazas y propuestas de mitigación. URL: https://www.microsoft.com/en-us/securityengineering/sdl/threatmodeling

7.5 Práctica de Requisitos de Seguridad

En relación a los requisitos de seguridad, la práctica indica que se deberán gestionar desde el inicio del proyecto tal y como tradicionalmente se han gestionado los requisitos funcionales. Por ello, se deberán utilizar las mismas herramientas con la que lo equipos de proyecto gestionan los requisitos, categorizándolos con la tipología adecuada.

Para esta práctica no se han encontrado herramientas open source específicas para la gestión de requisitos, aunque se ha encontrado una que podría servir para tal fin.

Herramientas

- ***Testlink***. TestLink en realidad es una herramienta para gestionar casos de prueba. Permite crear y gestionar casos de pruebas y organizarlos en planes de prueba, ejecutar casos de test y registrar los resultados, generar informes, etc. Adicionalmente permite la definición de requisitos para mantener trazabilidad, priorizar pruebas y asegurar que todos los requerimientos quedan cubiertos. URL: https://www.testlink.org/

7.6 Práctica de Arquitectura de Seguridad

Como se comentó esta práctica se centra en la seguridad de los componentes y tecnologías empleadas para el diseño de la arquitectura del software, evaluando la seguridad de los componentes que forman la arquitectura de la aplicación, así como las tecnologías y marcos de trabajo (*frameworks)* empleados.

Las actividades de esta práctica son de naturaleza manual y no se han encontrado herramientas específicas para dar cumplimiento a lo establecido por SAMM. En lo que respecta a las arquitecturas de referencia, tampoco se significa ningún software específico, pues bastará con disponer de un repositorio corporativo donde se publicar los artefactos de información. Únicamente se mencionará un *framework* que podría ayudar a realizar parte del trabajo de evaluación.

Herramientas

- ***ArchUnit***. ArchUnit es una librería open source, con la que es posible verificar que una aplicación Java cumple con un conjunto de reglas de arquitectura, entendiendo como tal en este caso a la forma en que se organizan, relacionan y comunican las clases de la aplicación. Así, se podrán ejecutar test automáticos durante la fase de construcción dependencias entre paquetes y clases, convenciones de codificación o

herencia, comprobaciones de anotaciones, uso de capas, etc. URL: https://www.archunit.org/

7.7 Práctica de Construcción Segura

Para ayudar a implementar las actividades de esta práctica, sí existen un buen número de herramientas que ayudan a dar cumplimiento a los controles de SAMM. Desde herramientas para automatizar los procesos de construcción hasta utilidades para verificar el cumplimiento de ciertos requisitos de seguridad, calidad, etc.

Herramientas:

- ***Maven***. Maven es una herramienta de software para la gestión y construcción de proyectos Java. Permite automatizar en gran medida el proceso de construcción del software, pero además es capaz de realizar otras tareas (a través de sus plugins) como pasar test unitarios y de integración, validar ciertos criterios de calidad, etc. URL: http://maven.apache.org/
- ***Jenkins***. Jenkins es un servidor de automatización que sirve de gran ayuda para automatizar gran parte del proceso de construcción del software. Permite que la generación se realice de forma centralizada y es capaz de integrar la verificación, la obtención de métricas de calidad, la ejecución de tests y obtención del resultado, establecer *quality gates* para para el proceso de construcción ante ciertas condiciones, desplegar los artefactos construidos en un repositorio central, etc. URL: https://www.jenkins.io/
- ***Nexus Repository***. Nexus es un repositorio de artefactos software donde se puede almacenar de forma centralizada las librerías y dependencias necesarias (y permitidas) para construir nuestros productos software. También podrá almacenar los productos construidos para diferentes entornos (testing, pre-producción, producción etc). Nexus ofrece soporte para repositorios de diversos tipos que almacena imágenes de Docker, paquetes npm, artefactos de Python y Go, etc. URL: https://www.sonatype.com/
- ***Sonarqube***. SonarQube es una plataforma de análisis estático de código (SAST) que permite evaluar el código fuente antes de la construcción del producto final. Tras el análisis proporciona un conjunto de métricas de calidad que se podrán utilizar posteriormente para mejorar la calidad del producto. También es capaz de detectar bugs y vulnerabilidades de seguridad en el código. En combinación con Jenkins permitiría parar la construcción y despliegue del software si este no cumple ciertos criterios de calidad. URL: https://www.sonarqube.org/
- ***OWASP Dependency Check***. Dependency Check es una herramienta que nos permite identificar las dependencias de nuestro proyecto y comprobar si tienen vulnerabilidades de seguridad conocidas. Se integra fácilmente con Maven a través del mecanismo de plugins. URL: https://owasp.org/www-project-dependency-check/

- ***OWASP Dependency Track***. Dependency-Track es una herramienta de análisis de composición de software (SCA) que realiza un seguimiento de todos los componentes de terceros utilizados en las aplicaciones (Bill of Materials -BOM-). Permite la consulta a múltiples bases de datos de vulnerabilidades, como NVD, VulnDB, etc. Sirve para monitorizar las aplicaciones con el fin de identificar vulnerabilidades de las dependencias de las aplicaciones. URL: https://dependencytrack.org/

7.8 Práctica de Despliegue Seguro

Para dar cumplimiento a algunos de los controles definidos por SAMM para esta práctica, sirven de gran ayuda algunas de las herramientas ya mencionadas en la práctica anterior. Adicionalmente se mencionan algunas herramientas específicas asociadas a la práctica de despliegue.

Herramientas:

- ***OWAS ZAP (Zed Attack Proxy)***. Es una herramienta DAST que permite escanear aplicaciones web para detectar vulnerabilidades de seguridad. OWASP ZAP se interpone entre el navegador y la aplicación web, intercepta los mensajes, los inspecciona y puede modificar el contenido si es necesario. Permite detectar automáticamente vulnerabilidades básicas del OWASP Top Ten y otro tipo de vulnerabilidades. URL: https://www.zaproxy.org/
- ***Hashicorp Vault***. Es una herramienta que permite almacenar de forma segura secretos y gestionarlos de forma dinámica. Por ejemplo, puede proporcionar de forma dinámica las credenciales de acceso a una base de datos o a cualquier otro sistema. Sirve para evitar el almacenamiento de secretos en el código fuente y ficheros de configuración. Ofrece un cifrado de datos seguro, control de acceso a los secretos, la entrega dinámica y la revocación de los mismos. Además cuenta con registros de auditoría y un API que permite a las aplicaciones y otros sistemas acceder a credenciales de forma dinámica y sencilla. URL: https://www.hashicorp.com/products/vault

7.9 Práctica de Gestión de Defectos

En este apartado no se han encontrado demasiadas referencias a aplicaciones opensource. Únicamente se propone un enlace a una herramienta desarrollada por OWASP. En alguna actividad también puede ser de gran ayuda alguna de las herramientas mencionadas en la práctica de Construcción Segura.

Herramientas:

- ***Defect Dojo***. DefectDojo es una herramienta gestión de defectos permite inventariar los defectos de las aplicaciones, realizar seguimiento, generar informes, métricas, etc. y se integra fácilmente con otras herramientas de ticketing como JIRA. URL: https://github.com/DefectDojo/django-DefectDojo

7.10 Práctica de Evaluación de la Arquitectura

Para dar cumplimiento a los controles de esta práctica no se han identificado herramientas específicas. Las evaluaciones se basarán fundamentalmente en revisiones manuales y pruebas específicas diseñadas *ad hoc*. Para automatizar dichas pruebas pueden ser de utilidad algunas de las herramientas mencionadas anteriormente como Jenkins.

7.11 Práctica de Pruebas Orientadas a Requisitos

Para llevar a cabo las pruebas que comprueban el cumplimiento de requisitos asociados a la seguridad se podrán utilizar herramientas genéricas de pruebas

Herramientas:

- JUnit. JUnit es un *framework* que permite la automatización de pruebas unitarias con el fin de verificar que los métodos de una clase realizan su función de forma adecuada. También sirve como herramienta para realizar las pruebas de regresión. URL: https://junit.org
- Selenium. Selenium es *framework* de pruebas funcionales que permite comprobar si el software que se está desarrollando funciona correctamente según los requisitos establecidos. Selenium permite grabar, editar, depurar y automatizar casos de pruebas. También sirve como herramienta para automatizar las pruebas funcionales de regresión. URL: https://www.selenium.dev

7.12 Práctica de Pruebas de Seguridad

Para dar cumplimiento a los controles de SAMM establecidos para esta práctica se podrán utilizar las mismas herramientas mencionadas en el punto anterior, así como la también mencionada OWASP ZAP.

Herramientas:

- ***OWASP Proactive Controls***. Se trata de una especificación con una propuesta de controles específicos de seguridad. Se describen las categorías de controles más relevantes que deberían incluirse de forma incondicional en cualquier desarrollo de software. URL: https://owasp.org/www-project-proactive-controls/

7.13 Práctica de Gestión de Incidentes

En este apartado no se han encontrado apenas herramientas open source para la gestión de incidentes de seguridad. Únicamente se menciona una referencia.

Herramientas:

- ***Cyphon***. Cyphon es una herramienta para la gestión y respuestas a incidentes. Permite recibir, procesar y priorizar los eventos para facilitar la labor de loa analistas en la investigación y documentación de incidentes. URL: https://www.cyphon.io/

8 Conclusiones finales

En el presente capítulo se enumeran y describen las principales conclusiones del Trabajo y algunos aprendizajes adquiridos durante el desarrollo del mismo.

8.1 Conclusiones principales

En la sociedad actual existe una gran dependencia de los productos software, pues cada vez son más necesario e insustituibles para la prestación de muchos servicios básicos y esenciales para las personas, empresas e instituciones. Esta fuerte dependencia y el aumento de vulnerabilidades que observa año tras año, genera escenarios de riesgo inasumibles debido impacto potencial que suponen los ataques informáticos. Los datos aportados ponen de manifiesto los graves perjuicios económicos que pueden suponer para las organizaciones los incidentes de seguridad, llegando a poner en riesgo incluso la propia supervivencia del negocio.

El enfoque más adecuado para poner solución a los problemas de seguridad del software es tener en consideración la seguridad desde la etapa de concepción de cualquier sistema de información. Este enfoque se ha denominado "*security by design"*. Según los datos analizados, la mayor parte de las vulnerabilidades del software tienen su origen en defectos de diseño y de programación. Incorporando la seguridad desde el inicio del proceso de desarrollo, se consigue eliminar una buena parte de los problemas de seguridad de los productos software. Por ello, no sorprende que la integración de la seguridad en el ciclo de vida del desarrollo de software sea actualmente una de las principales tendencias en las organizaciones.

Por otra parte, al igual que sucede con los aspectos de seguridad, los requisitos de cumplimiento también constituyen una amenaza para la supervivencia de las organizaciones. De la misma forma que en el caso de la seguridad, incorporando los requisitos de cumplimiento normativo desde el inicio del proceso de desarrollo ("*compliance by design*"), se eliminarán o reducirán una buena parte de los potenciales problemas de los productos finales. Si bien son requisitos de distinta naturaleza, se observa que el modelo SAMM los trata de forma equivalente en sus prácticas.

Existen diversas aproximaciones, métodos y buenas prácticas para implementar un ciclo de desarrollo de software seguro (S-SDLC). Sin embargo, estas no tratan el problema de la seguridad del software de una forma holística, ni permiten que una organización pueda medir, conocer y gobernar el grado de excelencia de sus prácticas de desarrollo de software seguro. El modelo de madurez SAMM no solo sirve para orientar a las organizaciones en la implementación de un S-SDLC, sino que proporciona orientación para la definición de un programa de seguridad del software, con un enfoque global y adecuado a los intereses y necesidades estratégicas de la organización. Por supuesto, SAMM también sirve como instrumento de medida, lo que permite a la organización establecer estrategias de implantación y mejora que le permitan alcanzar sus objetivos y valorar de forma objetiva su situación. En definitiva, facilita a la organización el gobierno de la seguridad en el desarrollo de software.

El enfoque de SAMM no se limita a las actividades que clásicamente se consideran que forman parte del proceso de desarrollo, sino que aborda la

seguridad del software de forma global e incluye prácticas relacionadas con aspectos de gobierno (entendido como la forma en la que se gestiona el proceso de desarrollo de software seguro), y con las operaciones. Precisamente, la inclusión de los aspecto relacionados con el gobierno, es lo que le confiere al modelo ese enfoque holístico que, en mi opinión, lo hace especialmente atractivo.

Como se ha comentado en diversos puntos del proyecto, SAMM es un modelo prescriptivo. Como tal, establece un conjunto de actividades y prácticas organizadas en niveles de madurez, cuya finalidad es que la organización disponga de un proceso de desarrollo de software más seguro. El hecho de ser un modelo prescriptivo es precisamente lo que hace que SAMM sea especialmente útil para aquellas organizaciones que tienen un nivel de madurez básico o nulo, pues proporciona una guía y facilita enormemente la puesta en marcha del programa de seguridad en el desarrollo de software.

Otra característica importante que se puede observar prácticamente en la mayoría de actividades del modelo, es la necesidad de automatizar. El principal objetivo de esta directriz no es la reducción de esfuerzo o la mejora del rendimiento, sino la implementación de procesos y procedimientos consistentes y repetibles. Así, minimizando la intervención manual por parte de las personas, se incrementa la fiabilidad de los procesos y se reduce drásticamente la posibilidad de que se produzcan errores humanos que puedan afectar a la seguridad. Por ello, cualquier programa de seguridad el software que esté basado en SAMM, implicará la utilización de herramientas que proporcionen la automatización necesaria para satisfacer los requisitos de las actividades del modelo.

Para finalizar, podría decirse que se han cumplido los principales objetivos planteados al inicio de este proyecto, en especial los referidos a la descripción del modelo y los elementos principales que lo constituyen, así como en la aplicación del mismo para la puesta en marcha de un programa de mejora incremental de la seguridad en el ciclo de vida de desarrollo del software.

8.2 Seguimiento de la planificación.

En este apartado se realiza una revisión de la planificación definida al inicio del proyecto.

En líneas generales, se ha sufrido de cierto retraso sobre el calendario previsto. Especialmente entre las entregas parciales PEC2 y PEC3. La causa fundamental ha sido la extensión del modelo. En la estimación inicial no se detectó el gran número de actividades que integran el modelo, lo que supuso por un lado un retraso importante en la entrega de la PEC3, así como la necesidad de reducir el alcance de alguno de los objetivos iniciales.

8.3 Evaluación de objetivos alcanzados

A continuación se realiza una valoración sobre el cumplimiento de los objetivos planteados al inicio del Trabajo.

- **Presentar el *framework* SAMM de OWASP y describir los elementos principales que constituyen el modelo**.

El objetivo se considera cumplido. La presentación preliminar del modelo es fundamental para comprender posteriormente los detalles del mismo. Adicionalmente, se considera que se ha profundizado en el modelo y se ha documentado con suficiente detalle todas y cada una de las actividades que lo componen.

- **Comparar OWASP SAMM con otros modelos de madurez**.
  El objetivo se considera cumplido. Se ha llevado a cabo la comparación de SAMM con BSIMM. Estos dos modelos de madurez son los de mayor relevancia en el ámbito del desarrollo seguro de software. Se han analizado y resumido las principales características de ambos modelos y sus diferencias.

- **Aplicación del modelo para la puesta en marcha de un programa de mejora incremental de la seguridad en el desarrollo.**
  El objetivo se considera cumplido casi en su totalidad. Se ha analizado el proceso de implementación de un programa de seguridad basado en SAMM, y se han descrito los pasos para la puesta en marcha de la iniciativa. Además, se ha realizado simulación conceptual sobre cómo abordar el proceso haciendo uso de las herramientas proporcionadas por OWASP SAMM para dicho fin.

- **Investigar la aplicación del modelo para el gobierno de la seguridad en el proceso de desarrollo de software.**
  El objetivo se considera cumplido. No se ha realizado un ejemplo práctico del proceso aplicado a una organización ficticia que mostrara diversas iteraciones de mejora continua alineadas con la estrategia de la organización. No obstante, a partir de la simulación realizada anteriormente se considera que es sencillo extrapolarla a un ciclo de mejora continua.

- **Desarrollar un ejemplo práctico de aplicación de SAMM a una organización ficticia con un equipo de desarrollo de tamaño pequeño y obtención de resultados de proceso de implantación incremental del modelo.**
  Pese a que inicialmente se había planteado realizar un ejemplo práctico del proceso aplicado a una organización ficticia, se ha considerado que con la simulación realizada queda suficientemente claro cómo aplicar SAMM para la puesta en marcha de un programa de seguridad en el software.

- **Elaborar propuesta basada en soluciones de código abierto para implementar diversas prácticas recogidas en el modelo.**
  Se considera parcialmente cumplido. Debido al sobreesfuerzo no previsto en las tareas relacionadas con la presentación y descripción del modelo, no ha sido posible llevar a cabo una propuesta práctica y completa que abordara la realización de ciertas actividades del modelo. En su lugar, se ha realizado una enumeración limitada de diversas herramientas que pueden ayudar a la implementación de los controles que propone SAMM.

8.4 Trabajo futuro

A continuación se enumeran varias líneas de trabajo futuro para complementar y ampliar el trabajo realizado.

- Investigar y realizar propuesta de aplicación de SAMM en diferentes metodologías de desarrollo de software. Por ejemplo, dada la creciente adopción de metodologías agile, sería interesante describir cómo encajar los controles del modelo en estos métodos de trabajo.
- En una línea más práctica, podría ser interesante llevar a cabo propuestas de implementación de los controles propuestos por SAMM para varias prácticas del modelo, basadas en el uso de herramientas *open source*. Esto podría ser interesante para proporcionar orientación a pequeñas y medianas empresas que no disponen de demasiados recursos para desarrollar sus programas de seguridad en el desarrollo de software.
- En la misma línea que la anterior, desarrollar propuestas de implementación de los controles de SAMM para diversos escenarios de desarrollo de software: aplicaciones Web, aplicaciones móviles, contextos de desarrollo cloud-native, etc.

9 Bibliografía y fuentes consultadas

[1] IEEE, «IEEE Standard Computer Dictionary: A Compilation of IEEE Standard Computer Glossaries.,» *IEEE Std 610,* pp. 1-217, 1991.

[2] IBM, «Informe sobre el coste de una brecha de datos 2020,» 2020.

[3] A. J. Wang, H. Wang, M. Guo y M. Xia, «Security metrics for software systems,» de *Proceedings of the 47th Annual Southeast Regional Conference, March 19-21, 2009*, Clemson, South Carolina, USA,, 01/01/2009.

[4] M. W. (Hrsg.), M. B. (Hrsg.), J. M.-Q. (Hrsg.), E. Bodden, M. Schneider, M. Kreutzer, M. Mezini, C. Hammer, A. Zeller, D. Achenbach, M. Huber y D. Kraschewsk, «Development of Secure Software with Security by Design,» 2013.

[5] M. Dawson, D. Burrell, E. Rahim y S. Brewster, «Integrating Software Assurance into the Software Development Life Cycle (SDLC),» *Journal of Information Systems Technology and Planning,* vol. 3, nº 6, pp. 49-53, 2010.

[6] Cognizant Technology Solutions, «The Security Challenge: What's next?,» 2019.

[7] Microsoft, «Microsoft SDL,» [En línea]. Available: https://www.microsoft.com/en-us/securityengineering/sdl. [Último acceso: 15 Mayo 2021].

[8] SAFE Code, «SAFE Code,» [En línea]. Available: https://safecode.org/wp-content/uploads/2018/03/SAFECode_Fundamental_Practices_for_Secure_Software_Development_March_2018.pdf. [Último acceso: 15 Mayo 2021].

[9] G. R. McGraw, Software Security: Building Security In, Addison-Wesley Professional, 2006.

[10] NIST, «Systems Security Engineering Considerations for a Multidisciplinary Approach in the Engineering of Trustworthy Secure Systems,» Noviembre 2016. [En línea]. Available: https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-160v1.pdf. [Último acceso: 2018 Mayo 2021].

[11] ISO, *ISO/IEC 15026-4. Systems and software engineering — Systems and software assurance — Part 4: Assurance in the life cycle,* 2012.

[12] Software Engineering Institute (SEI), CMMI® for Development, Version 1.3, Improving processes for developing better products and services, Technical Report CMU/SEI-2010-TR-033 ESC-TR -2010-033, 2010.

[13] Siemens AG, Security by Design with CMMI for Development, Version 1.3, CMMI Institute, 2013.

[14] A. L. Mesquida y A. Mas, «Implementing information security best practices on software lifecycle processes: The ISO/IEC 15504 Security Extension,» *Computers & Security,* vol. 48, pp. 19 - 34, 2015.

[15] Synopsys, «BSIMM framework,» [En línea]. Available: https://www.bsimm.com/framework.html. [Último acceso: 20 Abril 2021].

[16] OWASP, «SAMM Benchmarking,» [En línea]. Available: https://owaspsamm.org/benchmarking/. [Último acceso: 22 Abril 2021].

[17] R. Maués, «Blog Conviso Application Security,» Conviso, 31 07 2020. [En línea]. Available: https://blog.convisoappsec.com/en/comparing-samm-bsimm-models/. [Último acceso: 20 Abril 2021].

[18] OWASP, «OWASP Top Ten,» [En línea]. Available: https://owasp.org/www-project-top-ten/. [Último acceso: 28 Abril 2021].

[19] OWASP, «OWASP Risk Rating Methodology,» [En línea]. Available: https://wiki.owasp.org/index.php/OWASP_Risk_Rating_Methodology. [Último acceso: 22 Mayo 2021].

[20] O. L. Hevroni, «Threat Modeling as Code,» [En línea]. Available: https://www.omerlh.info/2019/01/19/threat-modeling-as-code/. [Último acceso: 3 Mayo 2021].

[21] OWASP, «Proactive Control: Encode and Escape Data,» [En línea]. Available: https://owasp.org/www-project-proactive-controls/v3/en/c4-encode-escape-data. [Último acceso: 28 Abril 2021].

[22] OWASP, «Quick start guide for SAMM 2.0,» [En línea]. Available: https://owaspsamm.org/guidance/quick-start-guide/. [Último acceso: 18 Mayo 2021].

[23] OWASP SAMM, «SAMM Assessment,» [En línea]. Available: https://owaspsamm.org/assessment/. [Último acceso: 20 Mayo 2021].

[24] Concord, «Concord | SAMM 2.0 Calculator,» [En línea]. Available: https://concordusa.com/SAMM/. [Último acceso: 20 Mayo 2021].

[25] OWASP SAMM, «OWASP SAMM Resources,» [En línea]. Available: https://github.com/OWASP/samm/tree/master/Supporting%20Resources/app. [Último acceso: 20 Mayo 2021].

[26] CCN-CERT, «GUÍA DE SEGURIDAD DE LAS TIC (CCN-STIC-812),» [En línea]. Available: https://www.ccn-cert.cni.es/series-ccn-stic/800-guia-

esquema-nacional-de-seguridad/522-ccn-stic-812-seguridad-en-entornos-y-app-web/file.html.

[27] OWASP, «OWASP Application Security Verification Standard,» [En línea]. Available: https://owasp.org/www-project-application-security-verification-standard/. [Último acceso: 25 Mayo 2021].

[28] H. Krasner, «The Cost of Poor Quality Software in the US: A 2018 Report,» 2018.

[29] B. Glas, «PG Security Advisors,» 11 Octubre 2020. [En línea]. Available: https://www.pgsecurityadvisors.com/samm-and-bsimm. [Último acceso: 22 Abril 2021].