Desarrollo de software seguro

Diaz Chaparro Luis Roberto
robertodiaz83@gmail.com
Universidad Piloto de Colombia
Bogotá, Colombia

Resumen—El artículo menciona aspectos importantes a tener en cuenta para vincular la seguridad y privacidad en el desarrollo de software. Los temas tratados giran entorno a la norma ISO 27034 y su adopción por parte de Microsoft para la creación de su modelo Microsoft SDL centrado en la incorporación de seguridad al ciclo de vida del desarrollo de software SDLC. No se trata de penetrar y parchear.

Abstract - In this article, important aspects for linking security and privacy in software development are presented. Topics like Microsoft adoption the ISO 27034 for the creation of the Microsoft SDL model focused add safety at lifecycle software development SDLC are presented. It is not penetrate and patch.

Índice de Términos— ISO 27034, ciclo de vida, desarrollo de aplicaciones, seguridad, programación, metodología, etapas, Microsoft SDL.

I. INTRODUCCIÓN

La ingeniería de software busca definir procesos, controlarlos y seguirlos con el fin de construir productos de calidad que garanticen al usuario su satisfacción, generalmente medida en términos de funcionalidad del producto, pero pocas veces evaluada en términos de usabilidad del mismo[1].

Las metodologías de desarrollo siempre se han esforzado por generar un producto que satisfaga las necesidades de negocio de una organización pero no determina una visión clara con respecto a la seguridad del proceso de desarrollo.

La figura 1 refleja una curva que basada en 3 aspectos importantes.

- Porcentaje de defectos introducidos según la fase del proyecto.
- Porcentaje de errores encontrados según la fase del proyecto.
- Costo de corregir los errores de acuerdo a la fase del proyecto

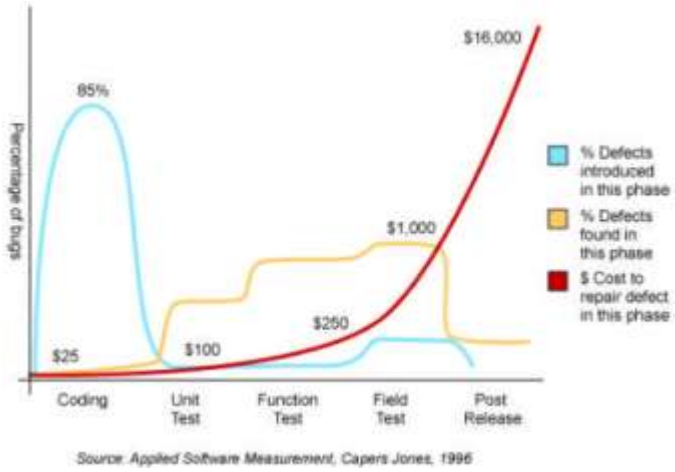

Fig.1 Porcentaje de defectos de un proyecto desarrollo de software [2]

Es muy común que en los proyectos se asuma la corrección de errores y/o defectos cuando el producto se encuentra terminado generando con ello ciclos infinitos de revisiones del diseño, modificaciones, actualizaciones y todo lo que eso implique a la hora de llevar a un ambiente en produccion.

Los riesgos detectados en un proyecto inciden de dos formas en el mismo. A corto plazo, van a condicionar la decisión sobre cuál va a ser la siguiente acción a tomar, encaminada a evitar, contrarrestar o asumir el riesgo detectado. A medio y largo plazo, los riesgos detectados en proyectos pasados pueden determinar también los niveles de calidad y las acciones que se van a exigir a los proyectos futuros [3].

Cuando hablamos de aspectos puntuales de desarrollo de la adopción de estrategias de desarrollo de software seguro, el panorama no es muy alentador, generalmente es fundamental reclutar y/o entrenar profesionales que sean capaces en el tiempo de desarrollar proyectos más rápidamente buscando siempre obtener como producto final una herramienta acorde a los requerimientos funcionales, pero no precisamente a los más seguros.

De acuerdo al estudio realizado por Forrester (State of Application Security), el 46% de las empresas consultadas afirman tener una metodología de seguridad propia. Mientras que las metodologías CMM y SDL se encuentran en los siguientes porcentajes de utilización [4].

Software security	% of respondents
We have developed our own software security methodology	46%
CMM or CMMI	20%
SDL	15%
Other	2%
OpenSAMM	1%
DISA STIG (for operations)	1%
We do not use any such methodology	14%
Don't know	16%

Fig.2 Uso de una metodología de software seguro [4]

En las siguientes líneas se expondrán algunas estrategias que buscar apoyar el ciclo de vida del desarrollo de software orientado a la obtención de aplicaciones que cubran las necesidades de seguridad y privacidad requeridas por el negocio.

II. SYSTEMS DEVELOPMENT LIFE CYCLE-SDLC

El SDLC es un enfoque por fases del análisis y diseño que sostiene que los sistemas son desarrollados de mejor manera mediante el uso de un ciclo específico de actividades del analista y del usuario[5]. La aplicación acertada de metodologías de desarrollo basadas en el ciclo de vida permite la consecución de aplicaciones que se ajustan a las necesidades operativas del negocio.

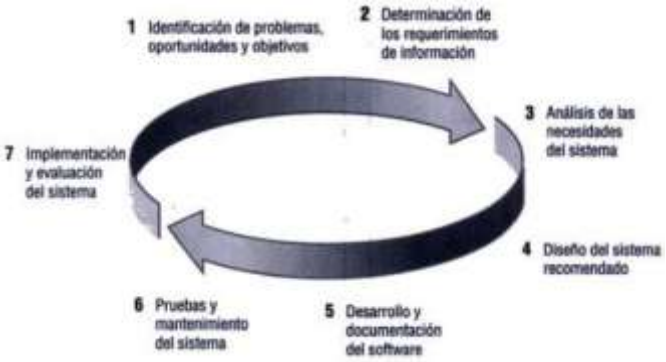

Fig.3 Fases del ciclo de vida del desarrollo de Sistemas [5]

III. ISO 27034

La aplicabilidad de la norma ISO 27034 apoya al software implícito de una aplicación y a los factores que contribuyen y afectan su seguridad, tales como: datos, tecnología, procesos del ciclo de vida de desarrollo de aplicaciones, procesos de apoyo y actores.

Por otro lado, la ISO27034 no proporciona directrices para la seguridad física, no proporciona controles o mediciones, ni proporciona especificaciones de codificación segura para lenguajes de programación[6]. La norma ISO27034 tampoco es un estándar de desarrollo de aplicaciones de software ni una norma de gestión de proyectos de aplicación.

Los requisitos y procedimientos especificados en la ISO 27034 no están destinados a ejecutarse de manera aislada, por lo contrario, serán integrados en los procesos existentes de la organización. De acuerdo a esto, las organizaciones deben asignar sus procesos y marcos existentes a los propuestos por la

norma ISO27034, lo que reduce el impacto de la aplicación de la norma.

Proceso de gestión de seguridad de las aplicaciones (ASMP). El proceso de gestión de seguridad de las aplicaciones que expone la norma ISO 27034-1, describe 5 pasos para enmarcar y alcanzar la seguridad en las aplicaciones:

• *Paso 1: Marco Normativo de la Organización.* La primera etapa de la ASMP implica el marco normativo Organizacional (ONF). Este marco contiene todas las normas, leyes, prácticas, roles y responsabilidades aceptadas por la organización. En este se definen los contextos de organización y se convierte en la única referencia organizacional para la aplicación de la seguridad.

• *Paso 2: Gestión de riesgos de seguridad de aplicaciones (ASRM).*
El propósito de este paso es recibir la aprobación de la organización en un nivel de confianza para las aplicaciones de negocio. La organización puede tener ya implementado un proceso de análisis de riesgos a nivel de empresa, pero un análisis de riesgos orientado a una aplicación específica debe ser realizado para cada proyecto de aplicación empresarial.

• *Paso 3: Marco normativo de la aplicación (ANF).*
Identificar todos los elementos relevantes del marco normativo organizacional que se aplicará a un proyecto de aplicaciones de negocio específico. Esto se convierte en el marco normativo de la aplicación (ANF). El nivel de confianza previsto, los contextos de aplicación (legal, comercial y tecnológica), los actores y las características de la aplicación, determinarán el contenido exacto del ANF.

En este paso también se definirá la seguridad del ciclo de vida de la aplicación (ASL) para el proyecto de aplicación de negocios. El ASL será un subconjunto genérico de la seguridad del ciclo de vida de aplicaciones (GASL). Para un proyecto de aplicación específica, el ASL contendrá sólo los procesos y medidas de seguridad de la GASL que fueron seleccionados de acuerdo con el nivel previsto de la confianza, las necesidades de la organización, los contextos de aplicación y las características de la misma.

• *Paso 4: Proyecto de Aplicaciones de Negocio.* Este paso es el uso real del marco normativo de la aplicación en los proyectos de aplicación del negocio. El equipo de ejecución pondrá en práctica las actividades de seguridad contenidas en la ANF.

• *Paso 5: Aplicación de Verificación de Seguridad.* Este proceso puede ser realizado por un interno o un equipo de verificación externa, utilizando los controles previstos por el marco normativo de la aplicación. El propósito de este paso es verificar y proporcionar evidencia de que una aplicación ha alcanzado y mantiene el nivel previsto de confianza. Se medirá el nivel real de confianza de la aplicación en un tiempo determinado[7].

Fig.4 ISO 27034 vista general [8]

Es resumen, la ISO 27034-1 es una norma basada en riesgos, maneja gestión compatible, mejora continua al sistema de gestión de la seguridad aplicado al ciclo de vida de las aplicaciones. [8]

IV. MICROSOFT SDL (SECURITY DEVELOPMENT LIFECYCLE)

Microsoft ha utilizado un enfoque basado en el riesgo para orientar las inversiones en seguridad de software a través de un programa de mejora continua y procesos desde el Security Development Lifecycle (SDL) se convirtió en una política obligatoria de toda la compañía en 2004.

En 2012, Microsoft utiliza la norma ISO / IEC 27034-1, un estándar internacional de seguridad de aplicaciones como una línea de base para evaluar

las políticas obligatorias de ingeniería, normas y procedimientos, junto con sus personas de apoyo, procesos y herramientas.

Todos los actuales obligatorios relacionados con la seguridad de aplicaciones políticas, normas y procedimientos, junto con sus personas de apoyo, procesos y herramientas cumplen los lineamientos de la norma ISO / IEC 27034-1 como publicó en 2011[9].

Microsoft SDL es un proceso de desarrollo de software que ayuda a los desarrolladores a crear software y los requisitos de cumplimiento de seguridad de direcciones más seguras al tiempo que reduce los costes de desarrollo[10].

Las actividades incluidas en el SDL de Microsoft reflejan el orden en el cual se deben desarrollar y se encuentran a la vez enmarcadas en un proceso de desarrollo de software tradicional.

Fig.5 ISO 27034 Ciclo de vida de desarrollo de software de Microsoft: simplificado [10]

- *Formación:* Como aspecto fundamental se destaca la necesidad y divulgar acertadamente la filosofía de la seguridad a todos los involucrados en el proceso, sin duda los conocimientos técnicos específicos son necesarios, pero no logran ningún objetivo si no se tiene una convicción plena de la importancia de otorgar seguridad al proceso de desarrollo de software.

  Los roles técnicos (desarrolladores, evaluadores y administradores de programas) deben asistir como mínimo una vez al año a una clase de formación en materia de seguridad en la que se aborden temas como:

  • *Diseño seguro:* reducción de la superficie de ataques, defensa en profundidad, principio de privilegios mínimos, configuraciones predeterminadas seguras.
  • *Modelos de riesgos:* Información general sobre los modelos de riesgos, implicaciones de diseño de un modelo de riesgos, restricciones de codificación basadas en un modelo de riesgos.

  • *Codificación segura:* Saturaciones de búfer (para aplicaciones que usen C y C++), errores aritméticos de enteros (para aplicaciones que usen C y C++), XSS (para código administrado y aplicaciones web), Inyección de código SQL (para código administrado y aplicaciones web), criptografía débil.

  • *Pruebas de seguridad:* Diferencias entre pruebas de seguridad y pruebas funcionales, evaluación de riesgos, métodos para poner a prueba la seguridad

  • *Privacidad:* Tipos de datos confidenciales, procedimientos recomendados de diseño de privacidad, evaluación de riesgos, procedimientos recomendados de desarrollo de privacidad, procedimientos recomendados de pruebas de privacidad.

- *Primera fase: Requisitos*
  • *Seguridad y la privacidad:* Se realiza un análisis de los requisitos de seguridad y privacidad, que incluya los requisitos de seguridad mínimos de la aplicación en el entorno operativo previsto, y un sistema de seguimiento de los elementos de trabajo y de las vulnerabilidades de seguridad.

  • *Umbrales de calidad y límites de errores:* Se trata de proporcionar una calificación a la aplicación evaluando que no existen vulnerabilidades como críticas o importantes.

  • *Evaluación de los riesgos de seguridad y privacidad:* Son procesos obligatorios que identifican los aspectos funcionales del software que requieren una revisión exhaustiva. Incluyendo información sobre las partes del proyecto que requieren modelado de riesgos, revisiones del diseño de la seguridad, pruebas de penetración así como el impacto sobre la privacidad (alto, moderado, bajo).

- *Segunda fase: Diseño*
  • *Requisitos:* Las especificaciones de diseño deben describir cómo se implementa de manera segura toda la funcionalidad proporcionada documentando de manera exacta y completa el uso previsto y la forma como se implementa de la seguridad.

  • *Modelo de riesgos:* Se consideran, documentan y describen de manera estructurada las implicaciones para la seguridad de los diseños en el marco de su entorno operativo previsto.

- *Tercera fase: Implementación*
  • *Usar herramientas aprobadas*: Se procura usar la versión más reciente de compiladores, framework y en general todas herramientas aprobadas a fin de aprovechar las nuevas protecciones y funciones de análisis de seguridad.

  • *Prohibir funciones no seguras*: Se deben analizar las funciones y API que se van a usar y prohibir las que consideran inseguras. Usar herramientas de análisis de código para comprobar si hay funciones prohibidas en el código y reemplazarlas por alternativas más seguras.

  • *Análisis estático:* Permite revisar el código de seguridad de forma escalable y contribuye a asegurar que se observan las directivas de codificación segura.

- *Cuarta fase: Comprobación*
  • *Análisis dinámico de los programas*: SDL usa herramientas en tiempo de ejecución como AppVerifier, junto con otras técnicas como pruebas de exploración de vulnerabilidades.
  • Pruebas de exploración de vulnerabilidades mediante datos aleatorios: La idea es provocar errores introduciendo datos aleatorios o de formato incorrecto en una aplicación.

  • *Revisión de los modelos de riesgos y de la superficie de ataques:* Si una aplicación se desvía de las especificaciones funcionales y de diseño es importante que se revisen los modelos de riesgos y la medición de la superficie de ataques para asegurar que se mitigan los nuevos riesgos creados como resultado de los cambios.

- *Quinta fase: Lanzamiento*
  • *Plan de respuesta a incidentes:* Se trata básicamente de tener definidos los equipos ingenieros, personal con capacidad para toma decisiones (7 por 24), planes para código heredado y planes para código de terceros.

  • *Revisión de seguridad final:* No cosiste en "penetrar y parchear". Consiste en un estudio de los modelos de riesgos, las solicitudes de excepciones, los resultados de las herramientas y el rendimiento teniendo en cuenta los umbrales de calidad y los límites de errores previamente determinados. Al final se puede dar cualquiera de los siguientes veredictos (Revisión de seguridad final superada, Revisión de seguridad final superada con excepciones, revisión de la seguridad final con remisión a una instancia superior).

  • *Lanzamiento o archivado:* El asesor de seguridad al igual que el asesor de privacidad deben certificar que el proyecto ha cumplido los requisitos de privacidad y seguridad para que se pueda enviar el software.

V. CONCLUSIONES

La convicción de la necesidad de la revisión temprana del código y procesos, así como el entrenamiento adecuado para cubrir dicha necesidad construyen el pilar del desarrollo seguro de aplicaciones.

Es importante que sin importar el tamaño de la solución de software que se quiera desarrollar, se cuente con una metodología para el desarrollo seguro de aplicaciones.

El desarrollo de software seguro reduce la necesidad de tener dependencia al parcheo de

aplicaciones.

Los parches de seguridad han creado una idea errónea con respecto a la gestión del desarrollo seguro.

No es una muy buena práctica el tener metodologías propias de gestión del código seguro ya que se tiende a ajustar la metodología a la mala práctica.

REFERENCIAS

[1] J. M. M. Gómez, M. E. H. Marín, and E. A. Díaz, "ENFOQUE METODOLÓGICO PARA EL DISEÑO DE INTERFACES DURANTE EL CICLO DE VIDA DE DESARROLLO DE SOFTWARE," *REVISTA GTI*, vol. 12, 2014.

[2] J. Capers, "Applied software measurement," ed: McGraw-Hill, 1996.

[3] L. F. Sanz and P. B. Silva, "Gestión de riesgos en proyectos de desarrollo de software en España: estudio de la situación," *Revista Facultad de Ingeniería Universidad de Antioquia*, pp. 233-243, 2014.

[4] I. Forrester Research, "State of Application Security," 2010. http://www.microsoft.com/en-us/download/details.aspx?id=2629

[5] K. E. Kendall and J. E. Kendall, *Análisis y diseño de sistemas*: Pearson educacion, 2005.

[6] www.owasp.org, "OWASP ISO IEC 27034 Application Security Controls Project," 2013. https://www.owasp.org/index.php/OWASP_ISO_IEC_27034_Application_Security_Controls_Project

[7] I. I. 27034, "Text for ISO/IEC WD 27034 – Information technology – Security techniques – Application security – Part 1: Guidelines to application security," ed, 2008. http://www.open-std.org/jtc1/SC22/open/n4309.pdf

[8] J. P.-S. P. M. a. Microsoft, "Software Assurance Forum - How International Standards Efforts Help Address Challenges in Today's Global Market Placeinterview," vol. 2013, ed, 2013. http://www.theedison.com/pdf/2013_Samples_Microsoft_SDL_White_Paper.pdf

[9] M. Corporation, "Microsoft SDL Conforms to ISO/IEC 27034-1:2011," 2013. http://blogs.msdn.com/b/sdl/archive/2013/05/14/microsoft-sdl-conforms-to-iso-iec-27034-1-2011.aspx

[10] M. Corporation, "Security Development Lifecycle," ed, 2010. http://www.microsoft.com/security/sdl/