Criterios mínimos de seguridad para el desarrollo y adquisición de software

Esta guía de criterios mínimos de seguridad se desprende de la "Guía de Controles Críticos de Ciberseguridad", un conjunto de 20 controles aplicables a sistemas, equipos, redes, datos, personas y procesos de negocio, diseñadas para mejorar el nivel de seguridad de las instituciones a través de acciones específicas, prácticas, priorizadas y medibles.

En la presente Guía se establecen aquellos criterios de seguridad mínimos que una institución debe contemplar en los requerimientos para el desarrollo y adquisición de software e implementaciones con software de terceros de modo a poder cumplir con los controles establecidos en la "Guía de Controles Críticos de Ciberseguridad".

Estos criterios mínimos de seguridad son aplicables al software desarrollado internamente por las instituciones públicas, adquirido de una empresa o desarrollador tercerizado y/o a través de donaciones a la institución.

Glosario:

A continuación, se definen algunos términos en el contexto de la presente guía:

- **Soporte de software:** se refiere al suministro de servicio pos-venta a clientes para actualizaciones y correcciones del código y cualquier problema relacionado al funcionamiento del código que puedan tener con respecto al funcionamiento esperado, según los requerimientos establecidos a la hora del desarrollo o adquisición. Se excluye la asistencia en el uso, instalación y/o configuración del software.
- **Cifrado:** se refiere al procedimiento reversible que utiliza un algoritmo matemático con cierta clave (clave de cifrado) para transformar un mensaje de tal forma que solo pueda ser accedido por quien posea la clave de descifrado.
- **Función criptográfica de hash:** es una función matemática que permite verificar fácilmente que algunos datos de entrada se asignan a un valor dado (hash o resumen), pero si los datos de entrada son desconocidos, es deliberadamente difícil reconstruirlo (o cualquier alternativa equivalente) conociendo el valor hash almacenado.
- **Vulnerabilidad**: falla o debilidad en un sistema de información que permiten que un atacante comprometer la integridad, disponibilidad y/o confidencialidad del mismo. En el contexto de software, se trata de fallas o debilidades en la programación del código y/o su configuración. Se excluye aquellas fallas que causan un funcionamiento indeseado pero que no permiten comprometer la integridad, disponibilidad ni confidencialidad.

Soporte y gestión continua del software:

1. Todo el software desarrollado debe contar con soporte de software del fabricante. Al momento de la adquisición se debe establecer claramente el tiempo de vida mínimo que se requiere para el software o sistema, y el fabricante debe ofrecer un tiempo de soporte igual o superior a dicho tiempo de vida.
2. En caso de que no sea posible contar con soporte de software del fabricante, el modelo de licenciamiento y la disponibilidad del código fuente debe ser tal que permita a la institución o a otra empresa o desarrollador de software nacional asumir dicho soporte.
3. El fabricante o servicio de soporte debe tener un canal de comunicación y/o mecanismo de reporte de vulnerabilidades o bugs de programación, de manera a que el cliente pueda contactarlo en caso de descubrimiento de vulnerabilidades. En caso de que el reporte ocurra dentro de la ventana de tiempo de vida solicitado, el fabricante o servicio soporte debe ser capaz de proporcionar una corrección a la vulnerabilidad de manera oportuna, según el acuerdo del nivel del servicio (por sus siglas en inglés, Service Level Agreement o SLA) especificado en el contrato o pliego de bases y condiciones.
4. El software debe poder ser inventariado por herramientas estándar automatizadas de inventario de software basados en el estándar *Common Platform Enumeration* (CPE), debiendo incluir como mínimo la información del nombre, versión, autor y fecha de instalación del mismo.

Gestión de usuarios, sesiones y privilegios:

1. El software debe permitir una gestión de usuarios de acuerdo a los requerimientos de la institución, con niveles de privilegios de acuerdo a los roles que éstos requieran (administrador, editor, usuario, etc.), basados en el principio de mínima necesidad de conocimiento.
2. El software debe permitir la revocación de acceso de usuarios, mediante un estado “desactivado” o similar.
3. Debe ser posible establecer una fecha de expiración para las cuentas de usuarios, a partir de la cual la cuenta deberá entrar a un estado “desactivado” o similar, hasta tanto se apruebe la continuidad de la misma. El parámetro de fecha de expiración podrá ser fijo o configurable por la institución, de acuerdo a sus requerimientos de negocio.
4. El software debe contemplar la expiración de sesiones de acuerdo a parámetros temporales. Estos parámetros pueden ser fijos o configurables por la institución, de acuerdo a sus requerimientos de negocio.

Autenticación y gestión de credenciales:

1. El software debe permitir la gestión individual eficaz de credenciales, debiendo permitir que cada usuario sea capaz de cambiar su propia contraseña. Se debe contemplar también mecanismos de recuperación de contraseñas, ya sea a través de un usuario de mayores privilegios o de mecanismos de auto-gestión por parte del usuario. Preferentemente, debe ser posible que al momento de la creación de cuentas permita forzar el cambio de contraseña luego del primer inicio de sesión.

2. El software que almacene y/o procese información crítica y/o que se utilice para un proceso crítico de la institución debe soportar autenticación de doble factor para los usuarios de privilegios elevados.
3. El software debe permitir establecer políticas de contraseña, que incluyan, como mínimo, la posibilidad de establecer los siguientes parámetros:
    a. longitud mínima de la contraseña
    b. complejidad de contraseña (mayúsculas, minúsculas, números y caracteres especiales, etc.)

    Los mencionados parámetros serán configurables por la institución, preferentemente, o en su defecto, deberán ajustarse a los lineamientos y estándares mínimos indicados por la institución.
4. Las contraseñas no deben almacenarse en texto claro, sino mediante la aplicación de funciones hash o funciones resumen. Para el almacenamiento de las contraseñas se debe utilizar funciones criptográficas seguras no reversibles de hash combinadas con salt aplicadas a las contraseñas. Algoritmos aprobados son los siguientes:
    a. Argon2
    b. PBKDF2
    c. scrypt
    d. bcrypt
5. De manera alternativa, se puede cifrar las contraseñas utilizando técnicas criptográficas reversibles únicamente en aquellos casos en que la clave secreta y/o privada de cifrado quede bajo el poder exclusivo del usuario dueño de la contraseña.

Gestión de registros de auditoría:

1. El software debe ser capaz de generar registros de auditoría de todos los eventos relevantes, con los detalles suficientes para permitir una trazabilidad adecuada, que abarque como mínimo los siguientes eventos:
    a. inicios de sesión de usuarios (exitosos y fallidos)
    b. delegación/impersonificación de cuentas de usuarios
    c. modificación de parámetros del sistema
    d. gestión de usuarios (cambio de contraseña, creación/eliminación/modificación de usuarios y/o grupos)
    e. acciones críticas llevadas a cabo por usuarios en el marco del proceso de negocio del sistema (edición de datos sensibles, eliminación de datos, etc.)
2. El software debe contemplar un mecanismo configurable de rotación de registros de auditoría, de acuerdo al parámetro de cantidad de tiempo (diario, semanal, mensual, etc.), como mínimo.

Cifrado:

1. El software debe cifrar toda la información sensible en tránsito, especialmente aquella información de carácter confidencial y/o cuya integridad deba asegurarse. Para ello se deberán utilizar protocolos de red cifrados, tales como HTTPS, SSH, SCP, SFTP/FTPS, etc.
2. Para sistemas basados en web, se adoptará el modelo SSL/TLS para el cifrado del tráfico. Los protocolos aprobados son TLS v.1.2 o superiores. Los protocolos TLS v.1.1 e inferiores y SSLv3

e inferiores no deben ser utilizados. Se deben seleccionar suites de cifrado robustos; una guía de referencia es:
https://cheatsheetseries.owasp.org/cheatsheets/TLS_Cipher_String_Cheat_Sheet.html
Se deben evitar las suites de categoría C o inferiores.

3. Las claves de cifrado deben ser robustas. Se recomienda una longitud de 2048 bits para RSA o equivalente, de acuerdo al estándar NIST SP 800-57. La clave privada debe quedar en poder de la institución, exclusivamente.

Codificación del software:

1. Se debe utilizar estándares de buenas prácticas seguras de programación, la cual debe ser seleccionada e implementada de acuerdo al lenguaje de programación y el entorno de desarrollo utilizado. Guías de referencia recomendadas son las siguientes:
    a. SEI CERT Coding Standards
    https://wiki.sei.cmu.edu/confluence/display/seccode/SEI+CERT+Coding+Standards
    b. OWASP Secure Coding Practices y OWASP Secure Coding Cheat Sheet
    https://www.owasp.org/index.php/OWASP_Secure_Coding_Practices_-_Quick_Reference_Guide
    https://www.owasp.org/index.php/Secure_Coding_Cheat_Sheet
    c. Oracle Secure Coding Guidelines for Java SE
    http://www.oracle.com/technetwork/java/seccodeguide-139067.html

2. El software debe contemplar un manejo seguro de errores. Se debe realizar y documentar la verificación explícita de errores para todas las entradas, comprobando el tamaño, el tipo de datos, los rangos de valores y/o formatos aceptables de modo a que éstos sean válidos para la operación que están por realizar.
3. Todos lo componentes de terceros utilizados para el desarrollo del software deben estar actualizados a la última versión estable disponible, contar con soporte por un periodo igual o superior al exigido para el proyecto y ser de confianza. Esto es aplicable, pero no limitante, a librerías, *frameworks*, *scripts*, funciones, *plugins*, plantillas, generadores de código, compiladores, entre otros.
4. Se debe realizar y documentar las pruebas de vulnerabilidades de código, incluyendo análisis estático y dinámico de vulnerabilidades para verificar que se cumplan los estándares mínimos de codificación segura, utilizando herramientas y/o guías de testing de seguridad estándar y aceptados por la industria. Guías de referencia recomendadas son las siguientes:
    a. OWASP Testing Project: https://www.owasp.org/index.php/OWASP_Testing_Project
    b. Open Source Security Testing Methodology Manual (OSSTMM)
    http://www.isecom.org/research/
    c. Microsoft Security Development Lifecycle (SDL) - Pasos 10 a 13
    https://www.microsoft.com/es-ES/download/details.aspx?id=12379
    https://www.microsoft.com/en-us/securityengineering/sdl/practices

Generadores de código:

1. Todo software generado mediante el uso de tecnologías de generación automatizada de código debe cumplir con los criterios mínimos de seguridad de la presente guía.