UNIDAD DE SERVICIOS PENITENCIARIOS Y CARCELARIOS - USPEC

GUÍA DE DESARROLLO SEGURO

BOGOTÁ D.C. ENERO 2021

USPEC
UNIDAD DE SERVICIOS
PENITENCIARIOS Y CARCELARIOS

TABLA DE CONTENIDO

1. PROCESO

Gestión de las Tecnologías de la Información.

2. SUB PROCESO

NA.

3. OBJETIVO

Establecer lineamientos de seguridad de información para el desarrollo de aplicaciones y sistemas de información que serán propiedad de la Unidad de Servicios Penitenciarios y Carcelarios - USPEC, ya sean In-house o tercerizados, a través del estándar de desarrollo de software seguro, con el propósito de aplicar las mejores prácticas de la ingeniería de software.

4. ALCANCE

Comprende los lineamientos inmersos en el ciclo de vida del desarrollo de software incluyendo principios seguros y consideraciones para el desarrollo seguro que definen acciones e impactos de seguridad a nivel de aplicación, de sistema operativo y de base de datos.

5. DEFINICIONES

- **Buffer overflow o Buffer overrun (Desbordamiento de búfer):** es un error de software que se produce cuando un programa no controla adecuadamente la cantidad de datos que se copian sobre un área de memoria reservada a tal efecto (buffer): Si dicha cantidad es superior a la capacidad preasignada, los bytes sobrantes se almacenan en zonas de memoria adyacentes, sobrescribiendo su contenido original, que probablemente pertenecían a datos o código almacenados en memoria. Esto constituye un fallo de programación.

- **Inyección SQL:** es un método de infiltración de código intruso que se vale de una vulnerabilidad informática presente en una aplicación en el nivel de validación de las entradas para realizar operaciones sobre una base de datos.

- **FTP**: el Protocolo de transferencia de archivos (en inglés File Transfer Protocol o FTP) es un protocolo de red para la transferencia de archivos entre sistemas conectados a una red TCP (Transmission Control Protocol), basado en la arquitectura cliente-servidor. Desde un equipo cliente se puede conectar a un servidor para descargar archivos desde él o para enviarle archivos, independientemente del sistema operativo utilizado en cada equipo.

- **OWASP:** (Open Web Application Security Project)

6. DISPOSICIONES GENERALES

- En la fase de diseño y levantamiento de requerimientos, los líderes del proceso deben identificar los requerimientos establecidos por las normativas o estándares aplicables, como son ISO 27001, Ley 1581 de protección de datos personales, así como lo establecido por la Oficina de Tecnología en los lineamientos del Ciclo de Vida de Software.

- Todo proveedor o tercero que participe en las etapas de diseño, desarrollo, implementación o mantenimiento de aplicaciones debe firmar el acuerdo de confidencialidad establecido por la USPEC.

- Todo tercero debe seguir estándares, buenas prácticas o modelos de madurez de desarrollo seguro, basándose en (o publicadas por) OWASP, NIST, SANS, SAMM, BSIMM o MICROSOFT. Al respecto de esta consideración, en el presente documento se contemplaran algunas de las vulnerabilidades referidas en OWASP (Validación de datos de entrada, Administración de autenticación y contraseñas, Administración de sesiones, Control de Acceso, Prácticas Criptográficas, Manejo de errores y Logs, y Protección de datos).

- Los riesgos que se encuentren a lo largo del ciclo de vida de desarrollo deben quedar documentados (principalmente aquellos que no han sido gestionados o que deben ser aceptados por la USPEC).

- Para los desarrollos que se realicen o contraten, se debe asegurar los derechos de autor y la propiedad intelectual del código fuente a la USPEC, para lo cual el software se debe registrar y patentar, de acuerdo a la legislación colombiana.

- Se debe contar con los ambientes de desarrollo, pruebas y producción, teniendo en cuenta que la separación de funciones y ambientes lógicos de trabajo es un método para reducir el riesgo accidental o deliberado del mal uso del sistema.

- Así mismo, es necesario implementar para todos los ambientes, a todos y cada uno de los equipos software de seguridad (antivirus, privilegios de acceso y otros que apliquen), con el propósito de proteger la integridad de los sistemas informáticos y de telecomunicaciones.

- Se debe contemplar el Principio del mínimo privilegio, los intervinientes en el desarrollo de software deben tener habilitado exclusivamente los derechos de acceso (escritura, lectura, etc.) a los objetos que ineludiblemente requieran para cumplir las funciones del puesto que ocupan.

7. CONTENIDO

7.1 PRINCIPIOS DE DESARROLLO SEGURO

- Se debe contar con ambiente de producción, pruebas y desarrollo y estos deben ser independientes. Estos ambientes deben ser lo más similar posible, a efectos de prevenir situaciones en las cuales el software desarrollado presente comportamientos distintos y errores en el ambiente de pruebas y producción.

- Los desarrolladores deben realizar su trabajo exclusivamente en ambiente de desarrollo, nunca en otros ambientes directamente.

- Los nombres de dominio para los ambientes de producción, pruebas y desarrollo, deben ser diferentes a efectos de evitar confusión durante la ejecución de las pruebas, puesta en producción y desarrollo.

- Es necesario que se tenga instalado el mismo manejador de base de datos y versión en los ambientes de prueba y producción. Si esto no es posible, usar herramientas automatizadas de propagación de una base de datos a otros.

- Se debe contemplar incluir réplicas de todos los componentes con los cuales el software tendrá interoperación en producción incluyendo: otras aplicaciones cliente servidor, bases de datos relacionales, componentes middleware, interfaces, demonios (daemons), procesos personalizados, utilidades FTP y otros.

- "Partir siempre de un modelo de permisos mínimos, es mejor ir escalando privilegios por demanda de acuerdo a los perfiles establecidos en las etapas de diseño.

- Si se utiliza un lenguaje que no sea compilado, asegurarse de limpiar el código que se pone en producción, para que no contenga rutinas de pruebas, comentarios o cualquier tipo de mecanismo que pueda dar lugar a un acceso indebido.

- Nunca confiar en los datos que ingresan a la aplicación, todo debe ser verificado para garantizar que lo que está ingresando a los sistemas es lo esperado y además evitar inyecciones de código.

- Hacer un seguimiento de las tecnologías utilizadas para el desarrollo. Estas van evolucionando y cualquier mejora que se haga puede dejar obsoleta o inseguras versiones anteriores.

- Todos los accesos que se hagan a los sistemas deben ser validados.

- Para intercambiar información sensible utilizar protocolos para cifrar las comunicaciones, y en el caso de almacenamiento la información confidencial debería estar cifrada utilizando algoritmos fuertes y claves robustas.

- Cualquier funcionalidad, campo, botón o menú nuevo debe agregarse de acuerdo a los requerimientos de diseño. De esta forma se evita tener porciones de código que resultan siendo innecesarias.

- La información almacenada en dispositivos móviles debería ser la mínima, y más si se trata de contraseñas o datos de sesión. Este tipo de dispositivos son los más propensos a ser que se pierdan y por lo tanto su información puede ser expuestas más fácilmente.

- Cualquier cambio que se haga debería quedar documentado, esto facilitará modificaciones futuras.

- Poner más cuidado en los puntos más vulnerables, no hay que olvidar que el nivel máximo de seguridad viene dado por el punto más débil."[1]

[1] https://www.welivesecurity.com/la-es/2014/02/28/10-principios-basicos-para-desarrollo-seguro/

7.2 CONSIDERACIONES PARA EL DESARROLLO SEGURO

Divulgación de las buenas prácticas en el desarrollo seguro de software: La socialización de las directrices impartidas a través de este documento están a cargo de la OTEC, con el fin de garantizar su uso y apropiación en todos los procesos de desarrollo adelantados desde y para la Entidad. Se debe hacer énfasis en la importancia del buen uso de la aplicación, así como la confidencialidad que maneja, las restricciones, roles y perfiles, manejos de usuarios y contraseñas con los que va a contar la aplicación.

Manejo de Entradas; Nunca confíe en las entradas (Buenas prácticas en el desarrollo de software)

- Una de las medidas más importante de defensa que los desarrolladores pueden tomar es validar las entradas que recibe su software. Esto no es sólo responsabilidad de los desarrollos, también comprende la validación de metodologías y procedimientos de desarrollo.

- Revisiones de entradas no seguras. Las entradas son la mayor causa de algunas de las vulnerabilidades más peligrosas (incluyendo desbordamiento de Buffer, inyección SQL, entre otras).

- Si una aplicación consta de más de un proceso, valide las entradas para cada proceso incluso si la entrada es proporcionada por otra parte de la aplicación. Valide la entrada incluso si esta es entregada sobre una conexión segura, si llega de una fuente confiable o si está protegida por permisos estrictos de archivo.
- Revisar las variables de entrada (incluyendo variables de entorno, valores de registro, servicios de red y nombres de ruta), teniendo en cuenta que pueden ser vulnerables a ataques y alterar su contenido.

Que se debe validar

Es importante tener en cuenta los siguientes aspectos para llevar a cabo una validación del desarrollo:

- Datos Incompletos o No validos: No repare los datos de entrada que fallen las validaciones de entrada, sólo rechácelos.

- Longitud de las entradas: Siempre realice la revisión contra un mínimo y un máximo de longitud esperada.

- Comprobación del límite de entradas numéricas: Siempre verifique las entradas numéricas contra valores máximos y mínimos. Sin una comprobación de límite para entradas numéricas, los atacantes pueden crear un desbordamiento de enteros.

- Filtrado de Meta-caracteres: Verifique que no existan caracteres especiales como por ejemplo una comilla simple (‘) que es un carácter en solicitudes SQL o periodos dobles (slash o backslash) ya que pueden ser utilizados para acceder a rutas de sistemas de archivos.

Control de Mensajes de Error

- No revele información sensible en las respuestas de error, incluyendo detalles del sistema, identificadores de sesión o información de la cuenta.

- Utilice errores controlados que no muestran la depuración o el seguimiento de la pila de la información.

- Implemente mensajes de errores genéricos y utilice páginas de error personalizadas.
- La aplicación debe manejar errores controlados de aplicación y no debe revelar configuración del servidor.
- La lógica de errores identificados, implica denegar el acceso por defecto.

Seguridad a nivel de aplicación, Sistema operativo y base de datos

- Las acciones e impactos que se presentan en el siguiente cuadro son una guía clave de los lineamientos para el desarrollo de software seguro.

Tabla 1
Lineamientos para Desarrollo Seguro

SEGURIDAD A NIVEL DE APLIACIÓN	
ACCIONES	**IMPACTO**
SQL INJECTION O COMMAND INJECTION	Es la habilidad de modificar la sentencia SQL del aplicativo para ejecutar código SQL arbitrario sobre el motor de base de datos. Se presenta cuando no existe una validación de entradas de usuario. Podría afectar la confidencialidad e integridad de los datos almacenados en una base de datos que una aplicación web tenga acceso.
CROSS SITE SCRIPTING XSS	Es la habilidad de ejecutar código script en el servidor de forma arbitraria. Ocurre cuando no se validan entradas en el aplicativo. Se usa para robar sesiones web, cookies, robar archivos y hasta producir phishing. Podría afectar la confidencialidad de los usuarios a la aplicación WEB, así como la imagen de la entidad.
BUFFER OVERFLOWS O BUFFER OVERRUNS	Es el ataque más común sobre las aplicaciones. Se presenta cuando no se valida correctamente el uso de memoria: se excede el tamaño de un arreglo o se excede el valor de una variable.
USO DE SESIONES	Podría comprometer la confidencialidad e integridad al permitir que pueda ser posible modificar la actividad de los usuarios en la aplicación WEB.
USO DE COOKIES	Podría comprometer la confidencialidad e integridad al permitir que pueda ser posible modificar la actividad de los usuarios en la aplicación WEB.
REVISIÓN DE CÓDIGO FUENTE POR ELEMENTOS MALICIOSOS	El uso de código proveniente de internet para las aplicaciones web de la entidad, podrían tener contenido malicioso.
PREVENCIÓN DE INYECCIÓN DE ARCHIVOS MALICIOSOS	Se podría comprometer la confidencialidad, integridad o disponibilidad de la aplicación Web.
ESTABLECER CADUCIDAD DE APLICACIONES WEB NO UTILIZADAS	Al desconocer que las páginas web se tienen almacenadas en el Servidor Web, podría permitir que páginas antiguas expongan con problemas con la integridad, disponibilidad o confidencialidad.
REVISIÓN POR MENSAJES DE ERROR	Algunos mensajes de error permiten conocer la versión del servidor web, así como posibles falencias que podrían poner en riesgo la seguridad de la aplicación.

REVISAR PÁGINAS QUE CONTENGAN CAJAS DE TEXTO PARA EVALUAR SU VULNERABILIDAD A HERRAMIENTAS AUTOMATIZADAS(SPAMBOTS)	El permitir que herramientas de ejecución automatizada sobre cajas de texto, podría afectar el funcionamiento de la aplicación, de tal forma que los datos reales que un usuario podría ingresar son sustituidos por datos falsos o erróneos.
LINEAMIENTOS DE CONTRASEÑAS EN CUENTAS DE USUARIO EN BASES DE DATOS	El permitir contraseñas sencillas o por defecto compromete la confidencialidad, integridad o disponibilidad.
REVISIÓN POR CONFIGURACIÓN POR DEFECTO	La configuración por defecto en plataforma, podría permitir que cualquier usuario, pudiera tener acceso no autorizado al servidor y con ello afectar la confidencialidad, integridad o disponibilidad de las aplicaciones web que mantiene el mismo.
REVISIÓN POR ARCHIVOS POTENCIALMENTE SENSIBLES (ZIP, RAR, CONFIG, CFG, ETC.)	El dejar archivos zip, rar, etc. Podría permitir a un atacante obtener toda la información con respecto al funcionamiento de la aplicación WEB.
USO DE TLS	El no usar TLS en servidores Web, expone a que los datos que fluyen entre el servidor y el cliente, puedan ser capturados en la red.
REVISIÓN DE DIRECTORIOS WEB QUE SE VISUALIZAN COMO DIRECTORIOS	Al poder ver los directorios web es posible conocer el contenido del sitio WEB.
ESTABLECER UNA ADECUADO CONTROL DE AUTORIZACIÓN	El no tener un adecuado modelo de autorización sobre los directorios de un servidor Linux, puede exponer la información sensible de las diferentes aplicaciones a usuarios no autorizados.
REVISAR MECANISMO DE NO REPUDIACIÓN	La ejecución de scripts, aplicaciones sobre bases de datos en los servidores Linux, no deben dejar evidencia de datos como Fecha, Hora, Usuario y Actividad.
SEGURIDAD A NIVEL DE SISTEMA OPERATIVO	
MÓDULO MODEVASIVE PARA APACHE	Usado para contener ataques Distribuidos Denegación de Servicios (DDoS) sobre el servidor web.
MÓDULO MODSECURITY PARA APACHE	Usado como firewall de aplicaciones web, ayuda a ataques de inyección de código, intrusiones no autorizadas sobre el aplicativo y sistema operativo. Permite filtrado de peticiones, técnicas evasivas, filtrado HTTPS, logs de auditoría entre otras funcionalidades.
SISTEMA OPERATIVO ACTUALIZADO	Si el sistema operativo usado es Oracle Linux debe ser actualizado a la última versión estable de la línea. Con parches de seguridad y vinculado al sistema de ULN que provee Oracle para actualizaciones.
FIREWALL DE SISTEMA OPERATIVO	Se cuenta con un firewall de sistema operativo que permite entradas a puertos no habilitados de la máquina, con el fin de adquirir información y acceso no autorizado a la misma.
INSTALACIÓN DE MALDET	MalDet (Malware Detection), es un aplicativo que se instala sobre el sistema operativo y se encarga de validar la presencia de software malicioso para luego eliminarlos o ponerlos como cuarentena para ser analizados por el administrador posteriormente.

SEGURIDAD A NIVEL DE BASE DE DATOS	
PROPIETARIO DE OBJETOS. ECENSO	• No deben existir conexiones directas a este usuario. • La contraseña no se entrega. • Cambios estructurales formalizando un control de cambio en producción. • Cambio de contraseña obligatoria cada 3 meses. • Se bloquea el usuario después de 3 intentos fallidos de conexión y dura un día bloqueado el usuario. • Auditoría de bd prendida sobe "CREATE SESSION" Auditoría de table, procedure, view.
USUARIO DE CONEXIÓN DE LA APLICACIÓN	• Se le otorgan privilegios CRUD sobre todas las tablas y objetos de CNP_WEB_INSCRICPCION. • La contraseña nunca caduca. • Tiempo de inactividad: 5minutos. • Se bloquea al usuario después de 10 intentos fallidos de conexión y dura un día bloqueado el usuario. • Responsables contraseña: Administradores Capa media (Aplicación PHP).
USUARIO DE CONEXIÓN DE APLICACIÓN DE MONITOREO DE ECENSO	• Se otorgan privilegios solo de SELECT sobre tablas de CNP_WEB_INSCRIPCION. • La contraseña nunca caduca. • Tiempo de inactividad: 5minutos. • Se bloquea al usuario después de 10 intentos fallidos de conexión y dura un día bloqueado el usuario. Responsables contraseña: Administradores Aplicación Monitoreo. • Privilegios de CRUD sobre CNP_WEB_INSCRIPCION Cambio de contraseña obligatoria cada 3 meses Se bloquea el usuario después de 3 intentos fallidos desconexión y dura un día bloqueado el usuario.
USUARIO PARA INGENIEROS QUE BRINDAN SOPORTE AL APLICATIVO	• Auditoría de INSERT, UPDATE, DELETE, sobre las tablas: CNP_ADMIN_CONTROL, CNP_ADMIN_USUARIOS, CNP_ADMIN_USUARIOS_ADM • Auditoría de sesión.

Nota: adaptado de *OWASP Top 10-2017*

- Los ambientes de desarrollo, pruebas y producción se deben encontrar claramente definida, independiente y controlados. Se sugiere la no existencia de ambiente local del desarrollador (Maquinas de usuario). Para los ambientes de prueba y desarrollo, no deben contener información real vigente o copiada de los sistemas de producción.

7.3 OTROS LINEAMIENTOS A TENER EN CUENTA

Estos lineamientos se aplican a todos los funcionarios y contratistas, que intervienen en los procesos de planeación, liderazgo y desarrollo de proyectos de desarrollo de software.

- El acceso al código fuente, debe ser restringido exclusivamente a los Desarrolladores.
- Se recomienda incorporar buenas prácticas de desarrollo seguro, como por ejemplo el estándar OWASP (*Open Web Application Security Project)* dentro de los requisitos de seguridad.

- Exigir pruebas técnicas de vulnerabilidad, para autorizar su salida a producción.

- Los desarrolladores revisarán y determinarán la acción a seguir para el tratamiento de las vulnerabilidades, para evitar que tengan brechas de seguridad.

- Se debe hacer implementación de controles para el manejo de versiones del código fuente en ambientes de producción, calidad y desarrollo.

7.4 GESTIÓN DE CONTRASEÑAS

La gestión de las contraseñas se debe realizar según lo establecido en la Guía Manejo de Contraseñas Seguras TI-GU-001.

RESUMEN DE CAMBIOS

Versión	Fecha	Numerales	Descripción de la modificación
01	31/07/2019	Todos	Se crea el documento.
02	07/01/2021	Todos	Se modifican todos los numerales teniendo en cuenta las necesidades del SGSI.
	04/10/2021	N/A	Se ajustan los códigos de los documentos de acuerdo con la actualización del Mapa de Procesos Institucional. La versión del documento se mantiene debido a que su contenido no se modificó.

RESPONSABILIDAD Y AUTORIDAD		
Elaboró / Actualizó:	**Revisó:**	**Aprobó:**
Nombre: Mayra Alexandra Agudelo Carvajal	Nombre: Diana Paola Cárdenas Huertas	Nombre: Camilo Alejandro Romero González
Cargo: Profesional Especializado	Cargo: Profesional Universitario	Cargo: Coordinador Grupo Comunicaciones
Dependencia: Oficina de Tecnología	Dependencia: Oficina de Tecnología	Dependencia: Oficina de Tecnología