De conformidad con lo dispuesto en los Decretos 2482 de 2012 y 1499 de 2017, los cuales establecen la creación y actualización del Modelo Integrado de Planeación y Gestión, así como las Políticas de Gobierno Digital y Seguridad Digital, la ITRC da cumplimiento a los criterios establecidos en el **Anexo 3** de la **Resolución 1519 de 2020**, y en observancia de la política de **Seguridad y Privacidad de la Información**, la cual ha sido definida y aprobada mediante la **Resolución 295 del 9 de julio de 2024**.

N°	REQUERIMIENTO	DESCRIPCIÓN DEL CUMPLIMIENTO
1	Implementación de Controles de Seguridad en el Ciclo de Vida del Desarrollo de Software.	El portal ya se encuentra en ambiente de producción. Actualmente, no hay desarrollos adicionales programados para mejoras de funcionalidades. Sin embargo, en caso de requerirse ajustes futuros, se cuenta con un entorno sandbox para realizar las pruebas correspondientes. El procedimiento establecido contempla que, una vez superadas todas las fases de prueba en el sandbox, los cambios se migran al entorno de producción. En caso de presentarse algún inconveniente tras la implementación, se ejecutará el procedimiento de rollback, garantizando así la estabilidad del sistema.
2	Implementación de Controles de Seguridad Relacionados con la Autenticación, Roles y Separación de Funciones.	Cuenta de Administrador del CMS del Portal (Sistema de Gestión de Contenidos). Se cuenta con una única y principal cuenta de administrador para el CMS del portal. A través de esta cuenta, se gestiona y carga todo el contenido relevante tanto para el público interno (Intranet) como externo (página web). Este rol es esencial para asegurar la correcta administración y actualización de la información disponible en ambas plataformas.
3	Exigencia de Medidas de Seguridad al Proveedor de Hosting.	Los proveedores de hosting nos proporcionan políticas de seguridad robustas y prácticas de seguridad avanzadas. Incluyen medidas de protección adecuadas frente a amenazas cibernéticas, así como un nivel de madurez en seguridad optimizado, que asegure la integridad, confidencialidad y disponibilidad de los datos almacenados y gestionados en sus infraestructuras. Además, el proveedor cuenta con mecanismos de

		prevención de acceso no autorizado, monitoreo constante, y protocolos de respuesta ante incidentes de seguridad. Un nivel adecuado de madurez en seguridad implica que el proveedor no solo cumpla con estándares de seguridad reconocidos internacionalmente, sino que también se adapte a las mejores prácticas y evolucione de acuerdo con los cambios en el panorama de amenazas.
4	Aplicación de Mecanismos de Hardening en la Infraestructura.	Se procedió a la eliminación de los usuarios alternativos al principal, garantizando que solo los usuarios autorizados tengan acceso al sistema. El CMS ha sido asegurado mediante la asignación de permisos específicos para los roles de usuario público y administrador, permitiendo la carga de información de manera controlada y conforme a las políticas de seguridad. Además, se han realizado las actualizaciones pertinentes tanto de los plugins como del núcleo (CORE) del sistema, asegurando su funcionamiento óptimo y la corrección de posibles vulnerabilidades.
5	Protección de la Integridad del Código.	El CMS está configurado con varias capas de seguridad para garantizar que la información enviada a través del sistema esté protegida de acuerdo con las mejores prácticas en seguridad informática. Las medidas clave implementadas incluyen: Cifrado de la Información: Toda la información transmitida entre el servidor y los usuarios es cifrada utilizando protocolos de seguridad robustos como TLS (Transport Layer Security), lo que asegura que los datos no sean interceptados ni manipulados por actores malintencionados durante su tránsito. Limitación del Perfil Público: El perfil público dentro del CMS está restringido a un conjunto mínimo de funciones y accesos, garantizando que los usuarios no autenticados solo puedan visualizar contenido que no comprometa la seguridad del sistema. No se permite que los perfiles públicos interactúen con información sensible ni realizar acciones que puedan alterar el funcionamiento del

		CMS. Los permisos y privilegios para los perfiles públicos están controlados y ajustados para minimizar riesgos, evitando la exposición de datos sensibles o funcionalidades críticas. Uso de HTTPS para Conexiones Seguras: Todo el tráfico entre el cliente y el servidor está protegido mediante HTTPS (HyperText Transfer Protocol Secure), lo cual previene que la información sensible, como credenciales de usuario o datos personales, sea interceptada por terceros durante la transmisión.
6	Ejecución de Monitoreos de Seguridad en Páginas Web.	El monitoreo y protección de las páginas web se lleva a cabo utilizando las soluciones de seguridad FortiWeb y FortiGate, que proporcionan una defensa integral contra una amplia variedad de amenazas: • FortiWeb: Proporciona protección avanzada contra ataques web, realizando escaneos periódicos para identificar archivos infectados, vulnerabilidades, y patrones sospechosos. Además, permite la verificación continua contra listas negras y la implementación de políticas específicas para mitigar ataques web como inyecciones SQL, Cross-Site Scripting (XSS), y otros vectores de ataque comunes. • FortiGate: Opera como un firewall de nueva generación, monitorizando el tráfico de red en busca de posibles ataques de denegación de servicio (DDoS) y otras amenazas. Además, asegura el tráfico entrante y saliente, proporcionando una capa adicional de defensa y control sobre el acceso a las aplicaciones web. Ambas soluciones trabajan de manera conjunta para garantizar una protección robusta, monitoreo constante y respuesta rápida ante incidentes, asegurando la integridad, disponibilidad y confidencialidad de las páginas web.
7	Exigencia de Mecanismos de Autenticación Seguros y	Para garantizar la seguridad del CMS, se ha establecido que solo exista el usuario necesario para la administración del sistema. Este usuario es el único autorizado para realizar tareas de gestión y

	Accesibles.	configuración del CMS, minimizando así el riesgo de acceso no autorizado. Adicionalmente, se aplica una política de renovación periódica de contraseñas cada tres meses, siguiendo las mejores prácticas de seguridad. Las contraseñas deben cumplir con los requisitos establecidos para garantizar su robustez, tales como: • Longitud mínima recomendada. • Uso de una combinación de caracteres alfanuméricos y símbolos. • Evitar el uso de contraseñas fácilmente adivinables o basadas en información personal. Estas medidas buscan reforzar la seguridad del CMS y protegerlo frente a posibles accesos no autorizados.
8	Actualización Continua del Software, Frameworks y Plugins de los Sitios Web.	Se han realizado las actualizaciones pertinentes tanto de los plugins como del núcleo (CORE) del sistema, asegurando su funcionamiento óptimo y la corrección de posibles vulnerabilidades.
9	Restricción de Accesos a Login Contra Ataques de Fuerza Bruta.	Para prevenir ataques de fuerza bruta, se ha implementado un control estricto sobre los intentos de acceso al sistema. El número de intentos fallidos de login está limitado a **5**. Si se superan estos intentos sin éxito, el acceso se bloquea temporalmente, lo que impide que se sigan realizando intentos automatizados o no autorizados en un corto período de tiempo.
10	Restricción y Ocultamiento de Páginas de Acceso Administrativo.	Las páginas de acceso interno o destinadas al personal administrativo están adecuadamente aseguradas para garantizar que solo los usuarios autorizados puedan acceder a ellas. Las medidas implementadas incluyen: • Autenticación Restrictiva: El acceso a estas páginas se encuentra limitado a usuarios con credenciales específicas y permisos asignados, de manera que se asegura que solo el personal autorizado pueda visualizar o interactuar con el

		contenido sensible. • Protección por Roles y Permisos: Se implementa un sistema de control de acceso basado en roles (RBAC), lo que permite restringir las acciones que cada usuario puede realizar dentro de las páginas internas, de acuerdo con su nivel de autorización. • Cifrado de Datos: La información transmitida a través de las páginas internas está cifrada mediante protocolos seguros (como TLS), garantizando que los datos sean protegidos durante su tránsito. Estas medidas aseguran que las páginas de interés interno y del personal administrativo se mantengan seguras, protegiendo la confidencialidad y la integridad de la información sensible.
11	Restricción de Escritura de Archivos desde la Web.	Los permisos asignados a las carpetas del CMS en el servidor web se han configurado de acuerdo con las recomendaciones oficiales del fabricante. Esta práctica asegura que las configuraciones de seguridad sean óptimas y alineadas con las mejores prácticas, minimizando el riesgo de accesos no autorizados o modificaciones indeseadas. • Permisos de Acceso Restringidos: Solo los usuarios y procesos autorizados tienen acceso a las carpetas críticas del CMS, con permisos estrictamente necesarios para su funcionamiento. Esto se implementa siguiendo el principio de mínimos privilegios, garantizando que solo los componentes del sistema que realmente necesitan modificar los archivos puedan hacerlo. • Protección de Archivos y Directorios Críticos: Las carpetas que contienen archivos de configuración, bases de datos o contenido sensible son protegidas con permisos adecuados para evitar que usuarios no autorizados puedan escribir o eliminar información. Este enfoque asegura la integridad y seguridad de las carpetas del CMS, protegiendo tanto los datos como el

		funcionamiento adecuado del sistema.
12	Crear copias de respaldo.	Contamos con una política específica de copias de seguridad que define claramente los procedimientos, responsabilidades y tiempos establecidos para garantizar la integridad y disponibilidad de la información. Esta política asegura que las copias de seguridad sean ejecutadas de manera eficiente y segura, tanto en entornos on-premise como en la nube. • Procedimiento de Copias de Seguridad: El proceso está claramente documentado y abarca tanto la frecuencia como los tipos de datos a respaldar, incluyendo sistemas, aplicaciones y bases de datos críticas. Esto incluye copias diarias, semanales y mensuales, conforme a las necesidades de recuperación de la organización. • Copia On-Premise: Las copias de seguridad se realizan de forma regular en almacenamiento local (servidores físicos o dispositivos dedicados), lo que permite un acceso rápido y confiable en caso de incidentes, asegurando la disponibilidad continua de los sistemas. • Copia en la Nube: Además de los respaldos locales, se implementa un sistema de copias de seguridad en la nube, proporcionando una capa adicional de protección contra desastres locales, como fallos de hardware o desastres naturales. Esto también facilita la recuperación remota de la información en caso de incidentes críticos. • Tiempos Establecidos: Se siguen tiempos y períodos predefinidos para la realización de las copias de seguridad, lo que asegura que la información esté siempre actualizada y lista para ser recuperada en caso de cualquier eventualidad. Estas acciones aseguran la protección de los activos informáticos, permitiendo una recuperación ágil y efectiva en caso de pérdida o corrupción de datos.

13	Almacenamiento de Trazas o Logs de Auditoría.	Los logs de auditoría relacionados con eventos de seguridad y actividades administrativas son gestionados y asegurados de acuerdo con el rol asignado para la administración de dichos archivos. Este enfoque garantiza que solo el personal autorizado tenga acceso a la información sensible y asegura la integridad y confidencialidad de los registros. • Acceso Basado en Roles (RBAC): El acceso a los logs se restringe estrictamente a los usuarios con roles específicos asignados para la administración y monitoreo de estos archivos. Solo aquellos usuarios con privilegios administrativos o de auditoría pueden visualizar, modificar o eliminar los logs. • Seguridad de los Logs Administrativos: Los logs se almacenan de manera segura, garantizando su protección contra accesos no autorizados o manipulaciones. Las configuraciones de acceso y permisos se revisan regularmente para asegurar que solo los administradores pertinentes puedan interactuar con los registros. • Protección de Logs Críticos: Los logs administrativos se almacenan en ubicaciones protegidas, con permisos de solo lectura para los registros históricos, lo que evita que sean alterados o eliminados sin la debida autorización. Además, los logs se cifran para proteger su contenido tanto en reposo como durante la transmisión. Estas medidas aseguran que los registros de auditoría sean tratados con el nivel adecuado de seguridad, cumpliendo con las políticas internas de protección de datos y garantizando la trazabilidad de todas las actividades administrativas en el sistema.
14	Garantizar de Conexiones Seguras y Configuración de Cabeceras de Seguridad.	Para asegurar la integridad y confidencialidad de la información en los portales transaccionales, se implementó un enfoque integral de seguridad basado en el uso de certificados SSL/TLS, HTTPS y otras medidas avanzadas de protección en las peticiones web. Estas acciones evitan la manipulación de parámetros y garantizan una experiencia segura para

		los usuarios. • Cifrado SSL/TLS y HTTPS: Se establece HTTPS como el protocolo obligatorio para todas las comunicaciones del portal web, asegurando que los datos transmitidos entre el usuario y el servidor estén cifrados mediante certificados SSL/TLS. Esto impide que los atacantes intercepten o modifiquen los parámetros de las peticiones y respuestas, asegurando la confidencialidad y la integridad de las transacciones. • Cifrado en la Estructura de las Peticiones: Además del cifrado SSL, se implementan métodos adicionales de cifrado que protegen la estructura interna de las peticiones web, asegurando que los parámetros sensibles, como credenciales de usuario y detalles de transacciones, no puedan ser manipulados ni comprometidos durante su transmisión. • Habilitación de Cabeceras de Seguridad HTTP: Se configuran una serie de cabeceras de seguridad para mitigar riesgos asociados con ataques web como Cross-Site Scripting (XSS), clickjacking, y otros tipos de vulnerabilidades. Las cabeceras configuradas incluyen: ○ Content-Security-Policy (CSP): Define una política de seguridad para controlar de manera estricta las fuentes de contenido permitidas (scripts, imágenes, estilos, etc.), reduciendo la posibilidad de inyección de código malicioso y ataques XSS. ○ X-Content-Type-Options: Previene que el navegador interprete de manera incorrecta el tipo de contenido de los archivos (como scripts) que pueden ser ejecutados de forma involuntaria, protegiendo contra ciertos ataques. ○ X-Frame-Options: Protege contra ataques de clickjacking, al impedir que el contenido del portal sea cargado dentro de un iframe en dominios no confiables.

		○ X-XSS-Protection: Activa la protección contra ataques XSS en navegadores compatibles, bloqueando la ejecución de scripts maliciosos detectados. ○ Strict-Transport-Security (HSTS): Obliga al navegador a usar únicamente conexiones HTTPS para acceder al portal, evitando posibles ataques que intenten redirigir al usuario a una versión no segura del sitio (HTTP). ○ Public-Key-Pins (HPKP): Permite especificar las claves públicas del servidor, asegurando que los navegadores validen solo certificados específicos, protegiendo contra ataques de suplantación de certificados. ○ Referrer-Policy: Controla la información que se incluye en el encabezado Referer al realizar peticiones, protegiendo la privacidad del usuario y evitando la exposición innecesaria de datos. ○ Feature-Policy: Limita el acceso a ciertas funcionalidades del navegador (como la cámara o el micrófono), mejorando la seguridad al restringir capacidades que podrían ser explotadas maliciosamente. Estas medidas garantizan que el portal transaccional esté protegido contra una amplia gama de vulnerabilidades, asegurando que las conexiones sean seguras, que los datos sensibles estén cifrados y que la confianza del usuario en la plataforma sea mantenida. Además, el uso de las cabeceras de seguridad refuerza la protección contra amenazas comunes, minimizando el riesgo de ataques.
15	Implementación de Mensajes de Error Genéricos y Accesibles.	Con el fin de proteger la seguridad del sistema y mejorar la experiencia del usuario, se implementan mensajes de error genéricos que no revelan detalles técnicos sensibles sobre la tecnología utilizada, las excepciones internas ni los parámetros específicos que han disparado el error. Esta medida ayuda a prevenir la exposición de vulnerabilidades y a garantizar que los usuarios, incluyendo personas con discapacidades,

		puedan comprender y actuar adecuadamente ante los errores. • Mensajes de Error Genéricos: Los mensajes de error mostrados al usuario son diseñados de manera que no proporcionen información sensible como detalles del sistema, código de error interno o stack traces. Por ejemplo, en lugar de mostrar un mensaje como "Error en la base de datos: conexión fallida", se mostrará un mensaje más general como "Ocurrió un error al procesar su solicitud. Por favor, intente nuevamente más tarde." Esto impide que un atacante pueda obtener información sobre la infraestructura o las tecnologías utilizadas en el sistema. • Evitar la Exposición de Parámetros Específicos: Los mensajes de error no revelan detalles sobre los parámetros de la solicitud que pudieron haber causado el fallo, ni información sobre el entorno de ejecución, como las versiones de las tecnologías o configuraciones de servidor. • Mensajes Comprensibles para Todos los Usuarios: Los mensajes de error son redactados de manera clara, sencilla y comprensible para el usuario promedio, sin necesidad de conocimientos técnicos. Además, los mensajes son diseñados para ser lo más informativos posibles sin comprometer la seguridad, proporcionando al usuario información sobre qué hacer a continuación (por ejemplo, "Intente recargar la página" o "Contacte con soporte si el problema persiste"). • Accesibilidad para Personas con Discapacidad: Se aseguran de que los mensajes de error sean accesibles para personas con discapacidades, siguiendo las directrices de accesibilidad web (WCAG). Esto incluye: ○ Uso de texto claro y legible, compatible con lectores de pantalla. ○ Mensajes de error presentados de manera destacada y en un formato que sea fácilmente percibido por usuarios con

		dificultades visuales o de movilidad. ○ Integración de mensajes en formato alternativo, como audio o texto a voz cuando sea necesario, para personas con discapacidades visuales. • Registro Detallado de Errores Internos: Mientras que los mensajes de error mostrados al usuario son genéricos, los errores internos son registrados de manera detallada en los logs del sistema. Esto permite a los administradores y desarrolladores identificar y solucionar problemas sin comprometer la seguridad o la experiencia del usuario. Estas prácticas ayudan a minimizar los riesgos de seguridad y mejorar la experiencia del usuario, garantizando que los errores sean manejados de manera profesional y accesible, sin revelar detalles internos que puedan ser aprovechados por actores malintencionados.
16	Protección del Binario de la Aplicación mediante Ofuscación y Seguridad en el Servidor Web.	Para evitar que actores malintencionados realicen ingeniería inversa (reversing) sobre la aplicación y accedan a su lógica interna, se implementan métodos de ofuscación avanzados en el binario de la aplicación. Este proceso garantiza que la lógica y el flujo de la aplicación sean difíciles de entender, incluso si se accede al código fuente o se intenta realizar un análisis detallado del binario. Además, se asegura la protección del servidor web desde el código y la infraestructura. • Ofuscación del Binario de la Aplicación: Se aplican técnicas de ofuscación sobre el código fuente antes de su compilación, de modo que cualquier análisis de ingeniería inversa sea extremadamente difícil. Esto incluye: ○ Renombrado de variables y métodos: Los nombres de las variables, clases y funciones se modifican a valores no significativos, lo que dificulta la comprensión de la estructura del código. ○ Codificación de cadenas: Las cadenas de texto sensibles, como claves de autenticación o URLs, son cifradas o

		codificadas, lo que impide que puedan ser fácilmente extraídas durante el proceso de análisis. ○ Modificación de la estructura del código: Se insertan instrucciones y modificaciones innecesarias que hacen que el flujo de control del programa sea más difícil de seguir, dificultando la tarea de un atacante que intente entender la lógica de la aplicación. ○ Protección de archivos binarios: Se implementan herramientas de protección de binarios para prevenir que el archivo compilado sea fácilmente descompilado o modificado. • Prevención de Ingeniería Inversa (Reversing): La ofuscación de código, junto con el uso de herramientas especializadas de protección de binarios, dificulta considerablemente la reversión del binario a su código fuente. Además, se pueden integrar métodos de detección de depuradores o herramientas de análisis que interrumpan la ejecución de la aplicación cuando se detecten técnicas de ingeniería inversa. • Aseguramiento del Servidor Web: Además de la protección del binario, el servidor web que aloja la aplicación está fortalecido a nivel de infraestructura para prevenir accesos no autorizados. Las medidas implementadas incluyen: ○ Configuración de permisos estrictos sobre archivos y directorios, asegurando que solo el personal autorizado pueda modificar o acceder al código. ○ Implementación de firewalls y sistemas de prevención de intrusiones (IPS) para detectar y bloquear posibles intentos de acceso no autorizado. ○ Cifrado de comunicaciones mediante SSL/TLS para garantizar que la transferencia de datos entre el servidor

		web y el cliente esté protegida. ○ Revisión y actualización periódica de la infraestructura del servidor, garantizando que se apliquen los parches de seguridad más recientes y se mitiguen vulnerabilidades conocidas. Estas medidas combinadas garantizan que la aplicación y su infraestructura estén protegidas contra intentos de análisis de ingeniería inversa, asegurando la confidencialidad de la lógica interna de la aplicación y evitando la exposición de vulnerabilidades a través del código. Además, al asegurar el servidor web, se previene el acceso no autorizado y se protege la aplicación desde múltiples capas de seguridad.
17	Sanitización de Parámetros de Entrada y Restricción de Formatos para Subida de Archivos.	Como parte de las medidas de seguridad implementadas en la Agencia ITRC, se han adoptado prácticas rigurosas de sanitización de parámetros de entrada para prevenir ataques como Cross-Site Scripting (XSS) y otros vectores de inyección maliciosa. Adicionalmente, se han implementado restricciones de formatos y tamaños para la subida de archivos, con el fin de proteger la aplicación y los datos del sistema contra amenazas. • Sanitización de Parámetros de Entrada: Todos los parámetros de entrada ya sean provenientes de formularios, URLs o cookies, son sometidos a un proceso de sanitización exhaustiva que incluye: ○ Eliminación de Etiquetas HTML/JS: Se eliminan todas las etiquetas HTML y JavaScript de los parámetros de entrada, evitando así que el contenido malicioso inyectado sea ejecutado en el navegador del usuario. ○ Eliminación de Caracteres Especiales: Se restringe la entrada de caracteres especiales (como <, >, &, ", ') que son comúnmente utilizados para construir scripts maliciosos, evitando la ejecución no autorizada de código. ○ Eliminación de Saltos de Línea y Espacios en Blanco: Se eliminan saltos de línea y

		espacios en blanco innecesarios que pueden ser utilizados para la evasión de filtros o para la manipulación de la entrada de datos. ○ Validación de Tipos de Datos: Se verifica que los parámetros recibidos correspondan al tipo de dato esperado (por ejemplo, que los campos numéricos no contengan caracteres alfabéticos). • Restricción de Formatos y Tamaños para Subida de Archivos: Para mitigar riesgos asociados con la carga de archivos, se aplican políticas estrictas que incluyen: ○ Limitación de Tipos de Archivos Permitidos: Se restringe la subida a formatos específicos de archivo, asegurando que solo se acepten tipos de archivos conocidos y seguros, como imágenes (JPG, PNG) o documentos (PDF, DOCX). Los archivos ejecutables, como .exe o .bat, son explícitamente prohibidos. ○ Control de Tamaño de Archivos: Se establecen límites rigurosos sobre el tamaño de los archivos que se pueden subir al sistema, evitando la sobrecarga del servidor y reduciendo la posibilidad de ataques mediante la carga de archivos grandes maliciosos. ○ Escaneo de Archivos Subidos: Los archivos cargados son escaneados automáticamente en busca de malware utilizando soluciones especializadas, como antivirus o sistemas de detección de malware, para garantizar que no contengan código malicioso. Todas estas medidas de sanitización de parámetros de entrada y restricciones en la subida de archivos han sido implementadas y validadas en la Agencia ITRC como parte de las políticas de seguridad web. Estas acciones contribuyen significativamente a la protección contra vulnerabilidades comunes en aplicaciones web y garantizan la integridad y confiabilidad de los datos

		procesados.
18	Sanitización de Caracteres Especiales y Secuencia de Escape de Variables en el Código de Programación.	Para proteger la aplicación contra ataques de inyección y vulnerabilidades de seguridad como Cross-Site Scripting (XSS) y SQL Injection, se implementa una rigurosa sanitización de caracteres especiales mediante secuencias de escape en las variables dentro del código de programación. Este proceso asegura que cualquier entrada de usuario o parámetro dinámico no sea tratado como código ejecutable, sino como datos válidos y seguros. • Sanitización de Caracteres Especiales: Los caracteres especiales que pueden ser interpretados como parte de un script o como comandos en la base de datos (por ejemplo, <, >, ', ", &, ;, entre otros) son sanitizados antes de ser procesados o almacenados. Esto se realiza mediante la eliminación, sustitución o codificación de estos caracteres, para evitar que sean utilizados con fines maliciosos, como la inyección de scripts o consultas maliciosas. ○ Ejemplo de sanitización de entrada: ▪ Entrada maliciosa: <script>alert('XSS');</script> ▪ Entrada sanitizada: <script>alert('XSS');</script> • Secuencia de Escape de Variables: Las secuencias de escape son utilizadas para asegurarse de que las variables no sean interpretadas como parte del código ejecutable. Esto es esencial especialmente cuando los datos son incluidos en consultas SQL, atributos HTML o URL, donde los caracteres especiales pueden tener significados especiales. ○ En el contexto de SQL, por ejemplo, se utilizan funciones de escape de caracteres para proteger las consultas de SQL

		Injection. Las entradas que contienen comillas simples (') se reemplazan por secuencias de escape (como \') para evitar que el código malicioso sea ejecutado en la base de datos. ○ En el contexto de HTML y JavaScript, se aplican secuencias de escape como <, >, ", ', y & para evitar que el contenido sea interpretado como código ejecutable. • Uso de Funciones de Escape Específicas por Contexto: Es esencial que las funciones de escape sean utilizadas según el contexto en el que se va a emplear la variable. Por ejemplo: ○ Escapado en consultas SQL: Se debe usar prepared statements o query parameterization para evitar SQL Injection, en lugar de concatenar directamente las variables dentro de la consulta. ○ Escapado en HTML: Se debe emplear funciones específicas para escapar caracteres especiales en atributos HTML, etiquetas y contenido textual. ○ Escapado en URLs: Los parámetros de entrada utilizados dentro de URLs también deben ser codificados en URL para evitar la manipulación de los parámetros. • Aplicación en el Código de Programación: Al integrar estas prácticas en el código de programación, se asegura que todas las entradas de usuario y datos dinámicos son adecuadamente controlados y procesados. La sanitización y escape de variables debe realizarse antes de cualquier uso de los datos en el sistema, asegurando que no puedan ser aprovechados para ejecutar código malicioso. • Protección contra Inyecciones y Explotación: La sanitización de caracteres especiales y el uso de secuencias de escape actúan como barreras

		de defensa cruciales contra los siguientes tipos de ataques: ○ Cross-Site Scripting (XSS): Al evitar que los caracteres de código como <, >, y & sean interpretados como parte de un script, se impide que los atacantes inyecten JavaScript malicioso. ○ SQL Injection: La sanitización de comillas y caracteres especiales evita que los atacantes manipulen las consultas SQL, inyectando comandos maliciosos. ○ Command Injection: Si se emplea en el contexto de líneas de comandos, la sanitización puede prevenir la inyección de comandos arbitrarios que podrían ser ejecutados por el servidor.
19	Revisar las recomendaciones de seguridad en la guía de desarrollo seguro de aplicaciones y Servicios Web Seguros de la Open Web Application Security Project (OWASP).	No se cuenta con desarrollo de software en la Agencia ITRC.
20	Implementación de Controles de Seguridad en Servidores para Protección Contra Ataques.	Para garantizar la seguridad de la infraestructura de servidores y proteger las aplicaciones web de vulnerabilidades comunes, se implementan controles de seguridad tanto a nivel de hardware como de software. Estos controles están diseñados para mitigar ataques como Cross-Site Scripting (XSS), SQL Injection, Denial-of-Service (DoS), entre otros, asegurando que las aplicaciones sean resistentes a amenazas externas y accesibles solo para usuarios autorizados. 1. Controles de Protección de Acceso • Autenticación y Autorización: Los métodos de autenticación robustos (como autenticación multifactor, contraseñas fuertes y gestión de roles) deben ser implementados para garantizar

		que solo usuarios autorizados puedan acceder a las interfaces administrativas y recursos sensibles del servidor. • Firewalls y Filtrado de IP: Los firewalls de red y aplicación (WAF - Web Application Firewall) se configuran para restringir accesos no autorizados y bloquear tráfico malicioso desde direcciones IP no confiables. • Acceso SSH seguro: Se deben aplicar configuraciones de seguridad en el acceso remoto (SSH), como la autenticación por clave pública y desactivación de accesos por contraseña. También es recomendable utilizar VPNs para conexiones remotas seguras. • Control de Accesos Basado en Roles (RBAC): Se debe definir y gestionar adecuadamente quién tiene permisos de acceso y acción en cada recurso del servidor, utilizando un modelo de control de acceso basado en roles para limitar privilegios innecesarios. 2. Protección contra Cross-Site Scripting (XSS) • Validación y Sanitización de Entrada: Todos los datos introducidos por los usuarios deben ser validados y sanitizados antes de ser procesados. Se deben eliminar caracteres especiales (como <, >, &, ", etc.) que puedan ser usados para inyectar scripts maliciosos. • Content Security Policy (CSP): Implementar una política de seguridad de contenido (CSP) estricta para controlar los orígenes de scripts que pueden ejecutarse en el navegador, evitando la ejecución de código no autorizado. • Escapado de Salida: Cuando se devuelvan datos al navegador, se debe asegurar que cualquier contenido dinámico sea escapado adecuadamente para evitar que se interpreten como código ejecutable. 3. Protección contra SQL Injection • Uso de Consultas Preparadas: Las consultas SQL deben ser construidas utilizando consultas

		preparadas o parametrizadas, evitando la concatenación de entradas de usuario directamente en la consulta SQL, lo que previene que los atacantes inyecten comandos maliciosos. • Escapado de Caracteres Especiales: Se deben aplicar funciones de escapado de caracteres para todos los parámetros de entrada que se van a utilizar en las consultas SQL, garantizando que caracteres como ', ", ; no sean tratados como parte del comando SQL. • Restricción de Permisos en la Base de Datos: Los usuarios de la base de datos deben tener el mínimo nivel de permisos necesario para realizar su función. Las consultas críticas deben ser ejecutadas con cuentas que solo tengan permisos de lectura o escritura limitados, evitando la ejecución de comandos de administración por usuarios no autorizados. 4. Protección contra Denial-of-Service (DoS) y Distributed Denial-of-Service (DDoS). • Uso de WAF (Web Application Firewall): Se debe instalar un firewall de aplicaciones web (WAF) que monitoree el tráfico entrante y bloquee patrones sospechosos, como múltiples solicitudes de la misma IP, con el objetivo de prevenir ataques de denegación de servicio (DoS). • Rate Limiting: Implementar limitación de tasa (rate limiting) para restringir la cantidad de solicitudes que un usuario o dirección IP puede realizar en un intervalo de tiempo determinado. Esto ayuda a mitigar ataques DDoS. • Protección de Servidor con Balanceadores de Carga: Para proteger los servidores contra sobrecargas debido a un tráfico excesivo, se pueden emplear balanceadores de carga que distribuyan el tráfico entre múltiples servidores, aumentando la capacidad de respuesta ante picos inesperados de tráfico. • Monitoreo de Tráfico y Alertas: Se deben usar herramientas de monitoreo y detección de intrusiones (IDS/IPS) para identificar patrones

		de tráfico inusuales o sospechosos que puedan indicar un ataque DoS o DDoS. Además, es recomendable tener alertas configuradas para notificar a los administradores en tiempo real. 5. Otros Controles de Seguridad en el Servidor • Hardening del Sistema Operativo: El servidor debe ser sometido a un proceso de hardening, desactivando servicios innecesarios y aplicando configuraciones de seguridad en el sistema operativo que limiten las vulnerabilidades, como la desactivación de puertos abiertos no utilizados y la configuración de políticas de seguridad rigurosas. • Cifrado de Tráfico (TLS/SSL): Todo el tráfico sensible debe ser cifrado utilizando SSL/TLS para garantizar la confidencialidad e integridad de los datos transmitidos entre el cliente y el servidor. • Seguridad en el Código de la Aplicación: Se deben emplear prácticas de desarrollo seguro, tales como la validación de entradas y la protección contra vulnerabilidades comunes (OWASP Top 10), garantizando que la aplicación no sea susceptible a ataques. • Parches y Actualizaciones Regulares: Mantener el sistema operativo, aplicaciones, frameworks y dependencias actualizadas con los últimos parches de seguridad es esencial para reducir las vulnerabilidades explotables.
21	Incorporación de Validación de Formularios: Lado del Cliente y Lado del Servidor.	La validación de formularios es un componente crucial en el desarrollo de aplicaciones web seguras y confiables. Para garantizar la integridad de los datos y prevenir la entrada de información maliciosa, se implementó la validación tanto del lado del cliente como del lado del servidor. Ambas capas de validación trabajan conjuntamente para asegurar que solo se ingresen datos válidos, mejorar la experiencia del usuario y proteger contra inyecciones y ataques. 1. Validación del Lado del Cliente La validación del lado del cliente se realiza en el navegador del usuario antes de que los datos sean

		enviados al servidor. Este proceso tiene como objetivo mejorar la experiencia del usuario y reducir la carga del servidor. Sin embargo, no debe considerarse una medida de seguridad definitiva, ya que puede ser saltada o manipulada por un atacante experimentado. • Objetivos de la validación del lado del cliente: ○ Mejorar la experiencia del usuario: Evitar la necesidad de enviar datos al servidor para validaciones básicas (por ejemplo, formato de email, longitud de contraseñas, campos requeridos). ○ Eficiencia: Reducción del tráfico al servidor al prevenir envíos innecesarios de datos incorrectos. • Métodos comunes de validación del lado del cliente: ○ HTML5 Form Validation: Utilización de los atributos de HTML5 como required, pattern, minlength, maxlength, type, etc., para validar campos de forma sencilla y rápida. ○ JavaScript: Uso de JavaScript para realizar validaciones complejas, como verificar que las contraseñas coincidan o que un número esté dentro de un rango específico. ○ Librerías de validación: Uso de librerías como jQuery Validation o Parsley.js para agregar reglas personalizadas y retroalimentación interactiva. ○ Limitaciones: La validación del lado del cliente debe ser vista solo como una capa de conveniencia y no de seguridad. Un atacante podría modificar el código del navegador para eludirla, lo que hace indispensable la validación también del lado del servidor. 2. Validación del Lado del Servidor ○ La validación del lado del servidor es la

		medida más importante y debe ser siempre confiable. Incluso si se implementa una validación en el cliente, los datos siempre deben ser validados nuevamente en el servidor antes de ser procesados o almacenados. Esto garantiza que los datos sean correctos y seguros, independientemente de la manipulación que pueda realizarse del lado del cliente. ○ Objetivos de la validación del lado del servidor: ○ Seguridad: Proteger contra la manipulación de los datos, inyección SQL, Cross-Site Scripting (XSS), Cross-Site Request Forgery (CSRF) y otros ataques. ○ Integridad de los datos: Asegurarse de que los datos recibidos sean válidos, estén en el formato correcto y cumplan con las expectativas de la aplicación. ○ Autenticidad: Verificar que los datos provengan de una fuente confiable y no hayan sido alterados en el tránsito. ○ Métodos comunes de validación del lado del servidor: ○ Validación de tipo de datos: Verificar que el tipo de dato de cada campo coincida con el esperado (por ejemplo, asegurarse de que una fecha esté en el formato correcto o que un número sea realmente un número). ○ Validación de formato: Comprobar que los valores de ciertos campos (como correo electrónico o URL) tengan el formato adecuado mediante expresiones regulares o funciones internas del lenguaje de programación. ○ Validación de límites: Validar que los valores estén dentro de los límites permitidos (por ejemplo, la longitud de una contraseña o el tamaño de un archivo

		cargado). ○ Escapado y sanitización: Asegurarse de que cualquier entrada de usuario no contenga código malicioso (como scripts) mediante el uso de escapado de caracteres y sanitización de entradas. • Protección contra ataques comunes: ○ SQL Injection: Al utilizar consultas preparadas (prepared statements) y parámetros vinculados, se evita que los datos del usuario sean interpretados como parte de las consultas SQL. ○ Cross-Site Scripting (XSS): Los datos recibidos del usuario deben ser escapados y sanitizados antes de ser mostrados en el navegador, para evitar la ejecución de scripts maliciosos. ○ Cross-Site Request Forgery (CSRF): Se deben utilizar tokens CSRF para proteger los formularios y garantizar que las solicitudes provengan de usuarios autenticados. 3. Mejor Práctica: Validación Conjunta Aunque la validación del lado del cliente mejora la experiencia del usuario, siempre debe complementarse con la validación del lado del servidor. Esto asegura que no se confíe únicamente en la validación del cliente, que puede ser eludida por un atacante. • Flujo recomendado: 1. Validación del lado del cliente: Realizar validaciones iniciales en el navegador para facilitar la interacción con el usuario y reducir la carga del servidor. 2. Envío de datos al servidor: Después de que los datos son validados en el cliente, se envían al servidor. 3. Validación del lado del servidor: Validar los datos nuevamente en el servidor para

		garantizar que no hayan sido manipulados y que cumplan con las reglas de seguridad y formato. 4. Procesamiento o almacenamiento seguro: Solo después de que los datos hayan pasado todas las validaciones, se procesan o almacenan de forma segura.
22	Implementación de Monitoreo de Seguridad en la Plataforma Tecnológica del Sitio Web.	El monitoreo y protección de las páginas web se lleva a cabo utilizando las soluciones de seguridad FortiWeb y FortiGate, que proporcionan una defensa integral contra una amplia variedad de amenazas: • FortiWeb: Proporciona protección avanzada contra ataques web, realizando escaneos periódicos para identificar archivos infectados, vulnerabilidades, y patrones sospechosos. Además, permite la verificación continua contra listas negras y la implementación de políticas específicas para mitigar ataques web como inyecciones SQL, Cross-Site Scripting (XSS), y otros vectores de ataque comunes. • FortiGate: Opera como un firewall de nueva generación, monitorizando el tráfico de red en busca de posibles ataques de denegación de servicio (DDoS) y otras amenazas. Además, asegura el tráfico entrante y saliente, proporcionando una capa adicional de defensa y control sobre el acceso a las aplicaciones web. Ambas soluciones trabajan de manera conjunta para garantizar una protección robusta, monitoreo constante y respuesta rápida ante incidentes, asegurando la integridad, disponibilidad y confidencialidad de las páginas web.
23	Establecimiento de Planes de Contingencia, DRP (Disaster Recovery Plan) y BCP (Business Continuity Plan) para Garantizar la Continuidad del Sitio Web 24/7 los	El Plan de Recuperación ante Desastres (DRP) se encuentra debidamente implementado y cumple con los estándares requeridos para garantizar la recuperación de los sistemas críticos del sitio web en caso de un evento de interrupción o desastre. Este plan está diseñado para restaurar la operatividad de la sede electrónica y plataforma tecnológica en el menor tiempo posible, minimizando la pérdida de datos y la

	365 Días del Año.	interrupción del servicio. • Componentes clave del DRP: ○ Copia de seguridad (Backup): Las copias de seguridad de la información crítica se realizan de manera periódica, tanto en instalaciones on-premise como en la nube. ○ Recuperación de infraestructura: El DRP incluye procedimientos específicos para la recuperación de servidores, bases de datos y aplicaciones, permitiendo restaurar la funcionalidad completa del sitio web. Este plan se actualiza regularmente para alinearse con las nuevas tecnologías implementadas, así como con los cambios operativos y organizacionales. La cobertura abarca tanto fallas técnicas como desastres naturales que puedan comprometer la operatividad de la sede electrónica. Plan de Continuidad del Negocio (BCP) Por otro lado, el Plan de Continuidad del Negocio (BCP) aún no ha sido implementado en su totalidad para garantizar la continuidad de la sede electrónica o sitio web 24/7, 365 días del año. Si bien algunos componentes del BCP han sido considerados, como la infraestructura de respaldo y los planes de emergencia, la continua operatividad del sitio web frente a situaciones no previstas aún no se encuentra completamente resuelta.
24	Restricción de la Escritura de Archivos en el Servidor Web mediante Permisos de Roles y Privilegios Asociados.	La restricción de la escritura de archivos en el servidor web es un aspecto fundamental en la gestión de la seguridad de una plataforma tecnológica. Esto asegura que solo los usuarios autorizados y con los privilegios adecuados puedan modificar, agregar o eliminar archivos en el servidor, protegiendo así la integridad del sistema y evitando posibles vectores de ataque, como la inyección de código malicioso o la corrupción de datos. 1. Asignación de Roles y Privilegios

		El control adecuado sobre los privilegios de escritura en el servidor web debe estar basado en un modelo de Control de Acceso Basado en Roles (RBAC, por sus siglas en inglés). Este modelo facilita la asignación de permisos de forma eficiente y segura, permitiendo que solo los usuarios con roles específicos puedan realizar acciones críticas, como la escritura o modificación de archivos. Roles Comunes en la Administración del Servidor Web: • Administrador del Sistema: Este rol tiene permisos completos sobre el servidor, incluyendo la capacidad de escribir, modificar y eliminar archivos. Sin embargo, las acciones de los administradores deben estar debidamente auditadas. • Desarrollador: En algunos casos, los desarrolladores necesitan permisos para modificar archivos del servidor. No obstante, estos permisos deben estar estrictamente controlados y limitados solo a las carpetas de desarrollo o áreas específicas del servidor, evitando el acceso a directorios sensibles. • Usuario Público: Este rol, generalmente asociado con los usuarios del sitio web, no debe tener permisos de escritura. Solo debe tener acceso de solo lectura para acceder a los archivos públicos del servidor, como imágenes o documentos. El acceso a archivos críticos debe estar estrictamente prohibido. Privilegios Asociados: Los privilegios de escritura deben estar asociados a roles específicos, basándose en el principio de privilegio mínimo. Solo se debe conceder el acceso necesario para realizar tareas específicas y evitar que usuarios no autorizados accedan o modifiquen archivos críticos del servidor. 2. Implementación de Permisos de Archivos Para implementar la restricción efectiva de la escritura de archivos en el servidor web, se deben aplicar permisos específicos a los directorios y archivos del servidor. A continuación, se describen las mejores

		prácticas: • Permisos a Nivel de Sistema Operativo: o Usar grupos y usuarios del sistema operativo para controlar los permisos de acceso. Por ejemplo, en un entorno Linux/Unix, se deben asignar permisos de lectura, escritura y ejecución (rwx) a los archivos y directorios según el rol del usuario. Los usuarios de solo lectura, como los usuarios públicos, deberían tener permisos limitados a r-- (lectura). o Ejemplo de configuración en Linux/Unix: o Este comando garantiza que solo los usuarios con privilegios adecuados puedan modificar los archivos en el servidor. • Directorios de solo lectura: Para las carpetas de archivos públicos (como imágenes o documentos estáticos), se debe configurar el servidor web para que no permita modificaciones. Se debe asignar permiso de solo lectura a los directorios públicos y asegurarse de que solo los administradores o usuarios autorizados puedan escribir en directorios privados. • Uso de SUID y SGID: Evitar el uso innecesario de los bits SUID (Set User ID) o SGID (Set Group ID) en archivos ejecutables. Estos bits permiten que los archivos se ejecuten con los privilegios del propietario del archivo, lo cual puede ser un riesgo de seguridad si no se configuran adecuadamente. 3. Uso de Contenedores y Aislamiento de Funciones En aplicaciones más complejas o cuando se necesitan entornos aislados, el uso de contenedores (como Docker) o máquinas virtuales puede proporcionar una capa adicional de seguridad. Esto permite controlar estrictamente las acciones permitidas dentro de cada contenedor o máquina virtual, limitando el acceso a los archivos del sistema operativo y reduciendo el riesgo de escritura no autorizada.

		• Contenedores: Asignar permisos específicos dentro de los contenedores para cada servicio que interactúa con el sistema. Por ejemplo, un contenedor de base de datos no debe tener acceso a los archivos del servidor web, y viceversa. • Accesos limitados: Restringir las operaciones de escritura a contenedores específicos o a directorios particulares, evitando que servicios no relacionados puedan modificar archivos críticos en el servidor. 4. Monitorización y Auditoría de Archivos Es fundamental implementar sistemas de monitoreo y auditoría para detectar accesos no autorizados o intentos de escritura en directorios o archivos restringidos. • Registro de eventos: Configurar el servidor web y el sistema operativo para registrar todas las acciones de acceso y escritura en los archivos sensibles. Esto puede incluir la creación, modificación o eliminación de archivos. • Herramientas de monitoreo: Utilizar herramientas de seguridad como OSSEC, Auditd o Splunk para monitorear la integridad de los archivos y generar alertas si se detectan actividades inusuales o intentos de escritura no autorizada. 5. Revisión Periódica de Permisos Es esencial llevar a cabo una revisión periódica de los permisos y roles asociados a la escritura de archivos en el servidor web, para garantizar que se mantenga el principio de privilegio mínimo. • Auditoría interna: Realizar auditorías internas periódicas para verificar que los permisos de escritura en los archivos del servidor web estén configurados correctamente y que no haya excepciones no documentadas. • Control de acceso basado en políticas: Definir políticas de control de acceso claras, que especifiquen quién tiene permiso para escribir en

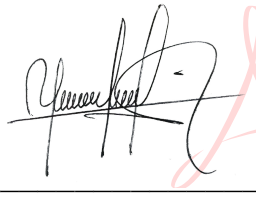
		qué directorios y bajo qué circunstancias.
25	Implementación de Sistemas Antivirus en el Servidor Web y Control de Escalamiento de Privilegios.	1. Implementación de Sistemas Antivirus en el Servidor Web La implementación de sistemas antivirus en los servidores web se tiene como medida crítica para proteger la infraestructura tecnológica frente a infecciones de malware, virus y otros tipos de software malicioso que puedan comprometer la seguridad del portal web y sus usuarios. Estos sistemas ayudan a garantizar la integridad de los archivos del servidor, evitando la ejecución de código malicioso que podría alterar la funcionalidad del sitio web o incluso robar información confidencial. Se manejan las mejores prácticas para Implementar un Sistema Antivirus en el Servidor Web: • Selección de software antivirus especializado: Es importante elegir un software antivirus que esté diseñado para entornos de servidor y que cuente con características específicas para la protección de servidores web, como detección en tiempo real y análisis de archivos ejecutables, scripts y contenido dinámico. Las soluciones que se tienen incluyen: ○ Kaspersky: Es uno de los proveedores más reconocidos de soluciones de seguridad informática a nivel mundial. En términos de antivirus y ciberseguridad, Kaspersky ofrece una amplia gama de garantías y características para proteger a los usuarios y las empresas contra amenazas cibernéticas. • Escaneo en tiempo real: Configurar el antivirus para realizar un escaneo en tiempo real de todos los archivos que se cargan, descargan o modifican en el servidor. Esto garantiza que cualquier archivo malicioso, como scripts o archivos adjuntos infectados, sea detectado y bloqueado antes de que pueda ejecutarse. • Análisis periódico de archivos: Programar análisis completos y periódicos de todo el servidor para

		detectar malware que haya podido pasar desapercibido en el monitoreo en tiempo real o que esté presente en archivos históricos. • Protección de directorios clave: Configurar el antivirus para que escanee directorios y archivos que son comúnmente atacados por malware, como aquellos asociados con el upload de archivos y directorios temporales donde se podrían almacenar archivos maliciosos. • Actualizaciones automáticas de firmas: Asegurarse de que las definiciones de virus (firmas antivirus) se actualicen automáticamente para detectar las amenazas más recientes. Esto es crucial, ya que los ataques de malware evolucionan rápidamente. 2. Control de Escalamiento de Privilegios en los Sistemas Operativos, Servidor Web y Bases de Datos El escalamiento de privilegios se refiere a la capacidad de un atacante o usuario malicioso para elevar sus permisos y obtener acceso no autorizado a recursos críticos del sistema. Un escalamiento exitoso de privilegios puede comprometer la seguridad de toda la infraestructura del portal web, incluyendo el servidor web, bases de datos y sistemas operativos. Mejores Prácticas para Controlar el Escalamiento de Privilegios: 1. Principio de Mínimos Privilegios. 2. Control de Acceso Basado en Roles (RBAC). 3. Uso de Contraseñas Fuertes y Autenticación Multifactor (MFA). 4. Monitoreo y Auditoría de Accesos. 5. Restricción de Ejecución de Comandos Elevados. 6. Segregación de Funciones. 7. Seguridad en las Bases de Datos. 8. Parcheo y Actualización Regular.

		9. Uso de Seguridad Basada en Hardware.

MOLINA RUBIANO
ANYELA JULIETH
2025.09.17 15:54:32
-05'00'

Elaboró:
Anyela Julieth Molina Rubiano
Gestor T1 Grado 09
Oficial de Seguridad de la Información.



Firmado digitalmente por
MURGAS GRANADOS
UBALDO ENRIQUE
Fecha: 2025.09.17
16:04:48 -05'00'

Revisó y Aprobó:
Ubaldo Enrique Murgas Granados
Jefe de Oficina Asesora de Tecnologías de la Información.