



Evaluación de seguridad y rediseño de arquitectura de un sistema oncológico orientado al cumplimiento de OWASP ASVS y normativas de protección de datos clínicos: caso SIGO

Vicente Joaquín Pérez Castro

vicente.perezc@usm.cl

Profesor guía: Raúl Monge

Resumen: El Sistema de Información y Gestión Oncológica (SIGO) procesa información de salud protegida (PHI) sin contar con una evaluación formal de seguridad, generando una brecha entre sus obligaciones regulatorias y su implementación técnica. Este trabajo ejecuta una auditoría de caja blanca sobre SIGO combinando análisis estático de código (SAST), análisis de composición de software (SCA) y modelado de amenazas STRIDE, contrastando los hallazgos contra un subconjunto trazable de controles OWASP ASVS v5.0.0 derivado de la legislación chilena (Leyes N.º 20.584, 19.628 y 21.258) y el estándar internacional HIPAA §164.312. El diagnóstico identifica 22 brechas de nivel Crítico distribuidas en ocho dominios de verificación. A partir de esta evidencia, se diseña una arquitectura de seguridad *to-be* basada en defensa en profundidad, cuya efectividad se demuestra mediante una matriz de trazabilidad que vincula cada brecha con su control técnico de mitigación y el requisito normativo satisfecho.

Palabras clave: OWASP ASVS, *White box testing*, SAST, Modelado de amenazas, Salud digital.

1 Introducción

1.1 Contexto

La gestión de la información oncológica en Chile enfrenta desafíos críticos que afectan directamente la calidad y la eficiencia de la atención médica. A pesar de ser una de las principales causas de mortalidad en el país, las cuales registran más de 59 000 casos nuevos y 31 000 muertes anuales [1], un número significativo de recintos hospitalarios carece de sistemas informáticos especializados para la gestión integral de pacientes oncológicos. La dependencia de sistemas de información hospitalaria de propósito general y la persistencia de procesos manuales para la actualización de fichas clínicas y el registro de exámenes incrementan el riesgo de errores, lo que retrasa la toma de decisiones críticas para los pacientes.

Para mitigar estas deficiencias, se implementan soluciones tecnológicas como el Sistema de Información y Gestión Oncológica (SIGO). Sin embargo, la adopción de este tipo de software introduce un nuevo desafío: el procesamiento de información privada altamente sensible, incluyendo diagnósticos y datos personales de salud protegidos (PHI). En consecuencia, la operación de estas plataformas se encuentra sujeta a estrictos marcos regulatorios, tales como la Ley chilena N.º 20.584 [2] y normativas internacionales como la *Health Insurance Portability and Accountability Act* (HIPAA) [3], cuyo cumplimiento es indispensable para asegurar la privacidad del paciente y mitigar riesgos legales e institucionales.

1.2 Definición del problema

La implementación inicial del sistema SIGO se desarrolló bajo las restricciones temporales de la 32ª Feria de Software (2024). La obligación de entregar requerimientos funcionales e interfaces usables en etapas tempranas relegó los controles de seguridad, generando una deuda técnica significativa. Al carecer de un enfoque de seguridad desde el diseño (*security by design*), la plataforma omite validaciones defensivas centralizadas.



CONSTANCIA DE VALIDACIÓN Y CONFIDENCIALIDAD DE MONOGRAFÍA A REPOSITORIO ACADÉMICO

1.- IDENTIFICACIÓN DEL TRABAJO ACADÉMICO

Tipo de monografía (marcar una opción): ☒ Memoria o trabajo de título ☐ Tesis de Postgrado

Título del trabajo: "Evaluación de seguridad y rediseño de arquitectura de un sistema oncológico orientado al cumplimiento de OWASP ASVS y normativas de protección de datos clínicos: caso SIGO"

Nombre del candidato(a): Vicente Pérez Castro (RUT 20.996.490-2)

Carrera / Grado: Ingeniero Civil Informático

Campus: Casa Central

Departamento: Informática

2.- VALIDACIÓN DEL PROFESOR GUÍA/DIRECTOR DE TESIS

Yo, Raúl Monge Anwandter, en mi calidad de profesor(a) guía/director(a) del trabajo académico mencionado anteriormente **DEJO CONSTANCIA** que:

- He revisado esta versión del documento y corresponde a la versión final aprobada del trabajo.
- El trabajo cumple con los requisitos académicos y de formato establecidos por la institución.

3.- EVALUACIÓN DE CONFIDENCIALIDAD POR PROPIEDAD INDUSTRIAL (marcar una opción)

☒ El trabajo **NO contiene** información que amerite confidencialidad y puede ser publicado de inmediato en repositorio con acceso abierto.

☐ El trabajo **CONTIENE** información con potenciales implicancias de propiedad industrial o intelectual y requiere un periodo de confidencialidad (**embargo**) por (marcar una opción):

☐ 6 meses ☐ 12 meses ☐ 2 años ☐ 3 años ☐ 5 años ☐ 10 años

Fundamentación de la necesidad de confidencialidad (obligatorio si se solicita embargo):

4.- FIRMAS

Profesor(a) guía o director(a) de memoria o tesis:

Fecha: 29 julio 2026

Firma:

Raúl Monge

Estudiante o Candidato(a):

Fecha: 29 julio 2026

Firma:

Vicente Perez

Este formulario debe ser insertado como página 2 de la memoria o tesis, completado y firmado por estudiante y profesor(a) antes de la entrega en portal PRISMA de Biblioteca USM.



El problema central radica en la falta de una auditoría formal de seguridad en la plataforma, lo que facilita la existencia y la permanencia de vulnerabilidades críticas no detectadas. Al gestionar información de salud protegida (PHI) y diagnósticos oncológicos estrictamente confidenciales, cualquier deficiencia de seguridad latente representa un riesgo inaceptable. Esta postergación de mecanismos de seguridad robustos en la implementación del sistema expone los datos clínicos a potenciales amenazas y vectores de ataque, tales como los documentados en el *Open Web Application Security Project* (OWASP) Top 10 [4].

En síntesis, la insuficiencia de controles defensivos y la carencia de una evaluación formal de seguridad impiden demostrar de manera reproducible el alineamiento del sistema frente a las exigencias de confidencialidad, integridad y trazabilidad requeridas para el tratamiento de información de salud protegida (PHI). En consecuencia, esta situación define la intervención como un problema de ingeniería centrado en cerrar la brecha existente entre las obligaciones normativas aplicables y la implementación de controles técnicos verificables a nivel de arquitectura.

1.3 Propuesta de solución

Se propone ejecutar la reingeniería de la arquitectura de seguridad de SIGO como un problema complejo de ingeniería. La intervención define inicialmente el alcance del estándar OWASP *Application Security Verification Standard* (ASVS) v5.0.0 [5]. Se seleccionan exclusivamente los requisitos de verificación que demuestran, de forma técnica y reproducible, el cumplimiento del marco normativo aplicable, comprendido por la legislación chilena sobre tratamiento de datos clínicos y referentes internacionales de salud. Esta convergencia entre norma y control establece un subconjunto trazable de requisitos ASVS que direcciona las fases analíticas posteriores.

Sobre esta base, se ejecuta un diagnóstico de caja blanca (*white box*) mediante análisis estático de código (SAST), análisis de composición (SCA) y modelado de amenazas (STRIDE), identificando tanto vulnerabilidades detectables de forma automatizada como defectos lógicos de diseño en la plataforma SIGO. Los hallazgos se contrastan contra el subconjunto ASVS definido a través de un *gap analysis*. Finalmente, la evidencia técnica fundamenta el diseño de una arquitectura de seguridad *to-be* anclada en el principio de defensa en profundidad, equilibrando la viabilidad operativa del sistema con sus respectivas obligaciones regulatorias.

1.4 Objetivos

Objetivo general:

- Evaluar la seguridad del sistema de gestión oncológica SIGO mediante una auditoría de código (*white box*) y rediseñar su arquitectura de seguridad bajo un enfoque de defensa en profundidad, demostrando formalmente el potencial de cumplimiento de un subconjunto trazable de controles OWASP ASVS derivado de las normativas de protección de datos de salud vigente.

Objetivos específicos:

- **Modelar los requisitos de seguridad y privacidad** aplicables al contexto oncológico, traduciendo las exigencias legales de la norma HIPAA y la Ley N.º 20.584 en un subconjunto específico de controles técnicos verificables del estándar OWASP ASVS.
- **Diagnosticar el estado actual de seguridad de SIGO** ejecutando un análisis integral de caja blanca, utilizando herramientas de SAST y SCA para identificar vulnerabilidades en el código fuente y librerías de terceros.
- **Realizar un modelado de amenazas (*threat modeling*)** sobre la arquitectura actual utilizando la metodología STRIDE, para identificar vectores de ataque lógicos y defectos de diseño que no son detectables por herramientas automatizadas.

- **Ejecutar un *gap analysis* formal** contrastando los hallazgos del diagnóstico contra el subconjunto normativo de controles ASVS previamente definido, clasificando las deficiencias según su severidad técnica.
- **Diseñar y validar teóricamente una arquitectura de seguridad *to-be***, especificando controles técnicos y de diseño que mitiguen las vulnerabilidades detectadas, demostrando su efectividad mediante una matriz de trazabilidad entre riesgos, normativa y soluciones arquitectónicas.

La ejecución secuencial de estos objetivos específicos estructura metodológicamente el trabajo propuesto. El modelado inicial de los requisitos legales establece la línea base sobre la cual se diagnostica el sistema de forma estática. Los resultados de este análisis técnico permiten contrastar el estado real de la plataforma contra el estándar esperado (*gap analysis*). Finalmente, la comprensión de estas brechas proporciona los insumos necesarios para cumplir el propósito central de diseñar la arquitectura de seguridad *to-be*.

1.5 Metodología de validación

La efectividad de la arquitectura de seguridad diseñada (*to-be*) se evalúa mediante una validación teórica basada en trazabilidad. Se construyó una matriz analítica que vincula unidireccionalmente cada vulnerabilidad identificada en el diagnóstico técnico con el control de mitigación propuesto en el rediseño y su respectiva exigencia de cumplimiento legal e industrial (ASVS). Este instrumento permite demostrar formalmente que la solución arquitectónica cubre íntegramente las brechas de seguridad iniciales y satisface los requerimientos normativos.

1.6 Introducción al marco teórico

El marco teórico articula dos pilares complementarios: las técnicas de análisis de seguridad de caja blanca (SAST, SCA y modelado STRIDE) y el marco normativo aplicable a sistemas clínicos (legislación chilena e HIPAA). Ambos convergen en OWASP ASVS como instrumento de verificación técnica, que traduce las obligaciones regulatorias en controles de software medibles y reproducibles. El Cap. 2 desarrolla en profundidad cada uno de estos pilares y establece los principios de arquitectura segura que fundamentan el rediseño *to-be*.

1.7 Organización del informe

El informe se organiza en seis capítulos de progresión lineal: el Cap. 2 establece el marco teórico y normativo; el Cap. 3 traduce ese marco en controles ASVS verificables; el Cap. 4 ejecuta el diagnóstico técnico *white box* (SAST, SCA, STRIDE); el Cap. 5 realiza el *gap analysis* formal sobre los hallazgos; y el Cap. 6 propone la arquitectura *to-be* con trazabilidad brecha–control–normativa. Se incluyen glosario y dos anexos de evidencia.



2 Marco teórico

2.1 Seguridad de aplicaciones web orientada a datos clínicos

La seguridad de aplicaciones web en salud digital exige un enfoque diferenciado respecto a dominios de menor criticidad, dado que la vulneración de la información afecta simultáneamente la privacidad del paciente, la continuidad asistencial y la confianza institucional [6]. El desarrollo de sistemas clínicos exige integrar la seguridad como principio arquitectónico fundamental y no como actividad posterior de corrección, especialmente cuando la plataforma procesa información de salud protegida (PHI) y opera mediante arquitecturas distribuidas con múltiples puntos de integración.

2.1.1 *Security by design* y deuda técnica de ciberseguridad

El paradigma *security by design* integra mecanismos defensivos desde la concepción arquitectónica [7]. Su omisión genera acumulación de deuda técnica, entendida como el conjunto de decisiones de corto plazo que aceleran la entrega funcional, pero aumentan el costo y complejidad del aseguramiento futuro [8].

En sistemas clínicos, esta deuda tiene impacto ampliado: una debilidad de autenticación, autorización o cifrado no solo constituye un defecto técnico, sino también una exposición de datos sensibles con potencial repercusión ética y regulatoria [2], [3]. Por ello, la literatura de ingeniería de seguridad recomienda tratar la deuda de ciberseguridad como riesgo arquitectónico acumulativo, priorizando su reducción mediante prácticas de trazabilidad de requisitos, revisiones de diseño y verificación sistemática de controles [9].

En el contexto de SIGO, la acumulación de deuda técnica de ciberseguridad exige sustituir los enfoques reactivos por una evaluación exhaustiva que integre el análisis de código y el modelado de amenazas arquitectónicas. Abordar esta deuda mediante un marco de evaluación formal como OWASP ASVS permite transformar las vulnerabilidades de implementación y las debilidades de diseño en un conjunto priorizado de brechas, lo que fundamenta el diseño de controles técnicos para mitigar el riesgo estructural del sistema.

2.1.2 Superficie de ataque y modelado de amenazas

La superficie de ataque de una aplicación corresponde al conjunto de interfaces, servicios, flujos y dependencias mediante las cuales un actor puede influir o comprometer el sistema [10]. En arquitecturas web modernas, dicha superficie incluye *endpoints* de API, mecanismos de sesión, integraciones entre servicios, almacenamiento de credenciales, librerías de terceros y fronteras de confianza entre componentes.

Analíticamente, este concepto permite priorizar y focalizar la evaluación defensiva. En un sistema clínico, las fronteras entre *frontend*, *backend*, bases de datos y microservicios externos constituyen zonas de especial sensibilidad, ya que allí convergen autenticación, autorización, tránsito de datos clínicos y decisiones de negocio [11].

La definición estructurada de la superficie de ataque constituye el requerimiento arquitectónico base para la ejecución de cualquier metodología de modelado de amenazas. Este enfoque integrador evita la dependencia exclusiva de diagnósticos automatizados aislados, habilitando en su lugar una evaluación sistemática de las vulnerabilidades subyacentes en el diseño de los flujos clínicos y administrativos.

2.2 Normativas de salud y protección de datos (Chile e internacional)

La gestión de información clínica exige una lectura conjunta de obligaciones regulatorias y mecanismos técnicos de seguridad, debido a que el cumplimiento en salud no puede sostenerse únicamente en declaraciones de política ni únicamente en controles aislados. Para el caso SIGO, se adopta un enfoque de convergencia

entre marco chileno, estándar internacional y verificación técnica con ASVS, con el objetivo de construir trazabilidad norma-control-evidencia.

2.2.1 Marco Chileno: confidencialidad clínica, vida privada y continuidad oncológica

En Chile, la protección de datos de salud no se sostiene de una sola norma, sino en un marco convergente que integra reglas sanitarias, reglas generales de vida privada y legislación sectorial oncológica.

La Ley N.º 20.584 [2], sobre derechos y deberes en la atención en salud, aporta el eje clínico-asistencial al establecer la reserva de la ficha clínica, los límites de acceso y los deberes de custodia. Para el diseño de sistemas como SIGO, esta norma estipula que la información clínica no constituye un dato operativo ordinario, sino un registro protegido cuyo acceso exige habilitación funcional y justificación asistencial. En términos de ingeniería, esto obliga a implementar mecanismos de autenticación fuerte, autorización por rol y contexto clínico, junto con la trazabilidad forense de las consultas y modificaciones de registros. Omitir una auditoría robusta a nivel de arquitectura genera una brecha estructural frente a este principio legal de reserva.

La Ley N.º 19.628 [12], sobre protección de la vida privada, introduce principios transversales de finalidad, proporcionalidad y seguridad, con requerimientos estrictos para datos sensibles de salud. A nivel de modelado de controles, exige minimizar la exposición de atributos, restringir el acceso a la necesidad efectiva de la función clínica y aplicar contramedidas proporcionales al riesgo de daño para el titular. Este marco legal justifica la implementación técnica de segregación de privilegios y protección de datos tanto en tránsito como en reposo.

La Ley N.º 21.258 [13], Ley Nacional del Cáncer, establece el imperativo de garantizar la trazabilidad y la continuidad de la atención oncológica. En sistemas de soporte clínico, esta exigencia requiere que la alta disponibilidad de la información coexista con controles estrictos de confidencialidad e integridad, operando de manera transparente para evitar interrupciones en el flujo asistencial.

La integración de estas tres normativas formula un marco legal y técnico sólido para el rediseño de SIGO. Esta convergencia justifica la implementación de controles técnicos verificables y mitiga el riesgo de tratar el cumplimiento regulatorio como un acto meramente declarativo.

2.2.2 Marco internacional: HIPAA §164.312 como *benchmark* técnico

Aunque HIPAA es una normativa estadounidense sin jurisdicción en Chile, en este trabajo se adopta como estándar internacional de buenas prácticas operacionales. Su valor instrumental radica en la especificación técnica de controles para proteger la información personal de salud (PHI) [4]. Se emplea como marco de evaluación estructurado en cinco dimensiones técnicas [16]:

- **Autenticación:** Mecanismos de verificación robusta de la identidad de los actores del sistema.
- **Control de acceso:** Políticas de autorización y restricción granular sobre la PHI.
- **Auditoría:** Trazabilidad inmutable de operaciones de lectura y modificación de registros.
- **Integridad:** Controles contra la alteración no autorizada de la información clínica.
- **Transmisión:** Protocolos de cifrado para la protección de los datos en tránsito por la red.

La evaluación de estas dimensiones directamente sobre el código fuente y la arquitectura permite conectar los hallazgos técnicos de manera fluida con los requisitos exigidos por el estándar ASVS.

2.3 Técnicas de análisis de seguridad en contexto *white box*



La seguridad en plataformas de salud digital exige evaluaciones exhaustivas que trasciendan los análisis superficiales. El enfoque de caja blanca (*white box*) proporciona acceso integral a la arquitectura y el código fuente del sistema, fundamentando el diagnóstico en la trazabilidad técnica de los flujos de datos clínicos.

En contextos altamente regulados, este análisis se estructura a través de tres técnicas complementarias: el análisis estático de código fuente (SAST), la revisión de dependencias externas (SCA) y el modelado de amenazas. La combinación de estas herramientas permite compensar las limitaciones que tiene cada método por separado. De esta forma, se asegura una evaluación de seguridad completa, capaz de identificar desde errores puntuales en la programación hasta debilidades en el diseño general de la arquitectura.

2.3.1 Análisis estático de código fuente (SAST)

El análisis estático de seguridad (SAST) inspecciona el código fuente sin requerir su ejecución para identificar patrones de programación inseguros y vulnerabilidades estructurales [17]. Su principal ventaja técnica es la detección temprana y automatizada de defectos en la gestión de autenticación, validación de entradas, manejo de errores y el uso de APIs criptográficas [18]. Sin embargo, los resultados generados por SAST exigen validación contextual: un hallazgo no constituye necesariamente una vulnerabilidad explotable de forma aislada, sino evidencia técnica que, al converger con el procesamiento de información de salud protegida (PHI), justifica su priorización en la fase de rediseño.

2.3.2 Análisis de composición de software (SCA)

El análisis de composición (SCA) identifica vulnerabilidades conocidas (CVE) en librerías de terceros, distinguiendo la exposición tanto en dependencias directas como transitivas. Su integración con la Lista de Materiales de Software (SBOM) [19] visibiliza la cadena de suministro, habilitando decisiones técnicas fundamentadas sobre la actualización, sustitución o mitigación de componentes. En aplicaciones clínicas, la ejecución de SCA es imperativa: una dependencia vulnerable compromete la confidencialidad y la integridad de los datos médicos, aun cuando el código nativo mantenga prácticas seguras de desarrollo. Por consiguiente, esta técnica constituye un pilar del diagnóstico estructural.

2.3.3 Modelado de amenazas STRIDE

El *framework* STRIDE proporciona una taxonomía exhaustiva para la identificación de amenazas de diseño: suplantación (*spoofing*), manipulación (*tampering*), repudio (*repudiation*), divulgación de información (*information disclosure*), denegación de servicio (*denial of service*) y elevación de privilegios (*elevation of privilege*) [20]. Metodológicamente, su aplicación requiere la construcción previa de diagramas de flujo de datos (DFD) para mapear el tránsito de la información y las fronteras de confianza de la arquitectura. Este análisis complementa directamente a las herramientas SAST y SCA al evaluar la lógica de negocio y visibilizar vectores de ataque en las interacciones entre componentes, mitigando los falsos negativos que los escáneres automatizados omiten por carecer de contexto operacional.

2.3.4 Taxonomía y cuantificación del riesgo: OWASP top 10, CWE y CVSS

La evaluación de seguridad en aplicaciones clínicas requiere estandarizar el diagnóstico técnico mediante marcos articulados de clasificación y métrica. En el nivel macro, OWASP top 10 provee una taxonomía de los riesgos más críticos en aplicaciones web, operando como un lenguaje común para contextualizar familias de debilidades recurrentes, tales como roturas en el control de acceso o fallas criptográficas [4]. Esta taxonomía aporta una clasificación inicial de exposición arquitectónica antes de aplicar estándares de verificación profundos.

Para alcanzar la granularidad técnica requerida, el estándar referencial CWE (*Common Weakness Enumeration*) categoriza la naturaleza exacta y la causa raíz del defecto subyacente en el software [21]. Una vez clasi-

ficada la debilidad, el marco CVSS (*Common Vulnerability Scoring System*) cuantifica su severidad y explotabilidad, entregando una métrica objetiva basada en el impacto directo sobre la confidencialidad, la integridad y la disponibilidad [22].

La integración conceptual de estos tres marcos (OWASP top 10, CWE y CVSS) conforma un modelo de evaluación estructurado. Esta parametrización conjunta permite transitar desde la identificación general del riesgo web hacia la cuantificación técnica objetiva del defecto. En sistemas que procesan información de salud protegida, este esquema de triple evaluación fundamenta teóricamente la priorización de mitigaciones y facilita la trazabilidad analítica hacia estándares de verificación estrictos como ASVS.

2.4 OWASP ASVS como estándar de verificación

El Estándar de Verificación de Seguridad de Aplicaciones (ASVS) de OWASP define los requisitos técnicos de seguridad verificables para aplicaciones de software [5]. A diferencia de modelos teóricos de riesgo, ASVS proporciona un conjunto estructurado de controles organizados por dominios (ej., autenticación y criptografía), cuyo nivel de exigencia se adapta según la criticidad del sistema y la sensibilidad de la información procesada.

2.4.1 Estructuras y niveles de verificación

ASVS estructura su aplicación agrupando controles en áreas temáticas. Estos dominios abarcan desde la gestión de identidad (autenticación) y autorización (control de acceso), hasta la implementación de mecanismos de criptografía para proteger la información [5]. Para adaptarse a distintos requerimientos, el estándar clasifica la exigencia en tres niveles:

- **Nivel 1 (L1):** representa la defensa mínima indispensable contra vulnerabilidades comunes y de fácil explotación (primera línea de defensa). Es apropiado únicamente para aplicaciones con un perfil de riesgo bajo que no manejan información sensible.
- **Nivel 2 (L2):** constituye la defensa recomendada contra la mayoría de los riesgos de software actuales. Es el estándar idóneo para aplicaciones que transaccionan datos sensibles B2B o B2C.
- **Nivel 3 (L3):** exige el máximo rigor técnico, demandando defensa en profundidad, controles criptográficos sofisticados y arquitecturas resilientes ante ataques dirigidos. Este nivel es mandatorio para infraestructuras críticas, tales como sistemas financieros, militares y ecosistemas informáticos de salud.

Esta categorización por niveles resulta fundamental en el sector salud. Dado que en una plataforma clínica no todos los componentes manejan información personal de salud (PHI), un módulo de agendamiento presenta un perfil de riesgo menor al de una base de datos de diagnósticos oncológicos. Esta granularidad permite concentrar los controles más estrictos exclusivamente en los vectores de mayor criticidad, optimizando los recursos técnicos.

2.4.2 ASVS como *framework* de evaluación: *gap analysis*

La aplicación de ASVS como herramienta de *gap analysis* (análisis de brechas) permite evaluar el software de forma objetiva. Alejándose de diagnósticos ambiguos, cada requisito técnico se califica con estados de cumplimiento concretos: "cumple", "no cumple", "cumplimiento parcial" o "no aplica". Para un proyecto de rediseño arquitectónico, este paso es fundamental. Transforma los errores técnicos aislados en un plan de acción claro, priorizando la remediación y definiendo cómo verificar la correcta solución.



Adicionalmente, el mapeo de los controles de ASVS con las leyes de protección de datos fortalece la solidez del proyecto. Este cruce traduce exigencias legales abstractas en configuraciones de software específicas y medibles. En consecuencia, la evaluación entrega evidencia técnica demostrable y reproducible, superando las validaciones informales de seguridad.

2.5 Principios de arquitectura segura para rediseño *to-be*

El diseño de arquitecturas clínicas seguras se sustenta en principios estables que orientan la implementación de controles estructurales, más allá de la mitigación de vulnerabilidades puntuales. En este contexto, la defensa en profundidad, el mínimo privilegio, *fail secure* y *zero trust* constituyen fundamentos complementarios para garantizar la resiliencia del sistema frente a amenazas internas y externas [11].

2.5.1 Defensa en profundidad (*defense in depth*)

La defensa en profundidad propone compartimentar controles en múltiples capas independientes (identidad, aplicación, datos, monitoreo e infraestructura), de modo que el fallo de una barrera defensiva no implique el compromiso total del sistema [11], [23]. En plataformas de salud digital, esta estrategia arquitectónica reduce el riesgo de propagación de incidentes que afecten la información de salud protegida (PHI) y habilita la contención temprana de eventos críticos.

2.5.2 Principios: mínimo privilegio, *fail secure*, *zero trust*

El principio de mínimo privilegio exige asignar de forma granular los permisos estrictamente necesarios para cada función y contexto operativo. El concepto de *fail secure* establece que, ante un error de software o condición inesperada, el sistema debe degradar y fallar hacia estados de mayor seguridad, bloqueando el acceso por defecto. Por su parte, el paradigma *zero trust* elimina los supuestos de confianza implícita entre los componentes de red y exige la verificación continua de la identidad, el contexto y la autorización para cada transacción [24]. Estos principios resultan críticos en arquitecturas web que procesan información de salud, ya que fundamentan la estructuración de controles verificables de autorización, la segmentación de la confianza y la resiliencia de las sesiones operativas entre servicios heterogéneos.

2.5.3 Criptografía aplicada en sistemas de salud

El uso de la criptografía en entornos de salud constituye una decisión arquitectónica estructural y no una mera adopción abstracta de bibliotecas de *software*. Su implementación implica seleccionar mecanismos de cifrado e integridad que estén alineados con el riesgo real de exposición, la gestión segura del ciclo de vida de las claves y los requisitos normativos de trazabilidad [25]. La concepción del sistema debe orientarse a garantizar matemáticamente la protección de los datos asistenciales tanto en tránsito como en reposo, asegurando paralelamente la solidez criptográfica de los *tokens* y mecanismos subyacentes de autenticación.

Los pilares desarrollados en este capítulo (el paradigma de *security by design*, el marco normativo convergente, las técnicas de diagnóstico *white box*, el estándar ASVS y los principios de rediseño) no operan de forma aislada. Su articulación define el instrumental analítico del trabajo: las normativas establecen el qué exigir, ASVS traduce esas exigencias en controles verificables, y las técnicas *white box* proveen la evidencia técnica para contrastarlos. El Cap. 3 operacionaliza esta convergencia, traduciendo las obligaciones legales en la línea base de controles ASVS contra la cual se evaluará SIGO.

3 Modelado de requisitos normativos

La transición desde un marco regulatorio abstracto hacia una arquitectura de *software* auditable exige la operacionalización de las normativas de salud digital en controles técnicos verificables. Este capítulo establece la línea base de seguridad para el sistema SIGO, traduciendo las obligaciones de confidencialidad, integridad y disponibilidad en el marco legal chileno (Leyes N.º 19.628, 20.584, 21.258) y el estándar internacional HIPAA (§164.312) hacia requerimientos específicos del estándar OWASP ASVS v5.0.0 [5].

Para articular este modelo, se emplea un enfoque de trazabilidad directa. Las disposiciones legales se categorizan en dominios de seguridad aplicados a contextos clínicos (autenticación, control de acceso, auditoría y protección de datos) y se vinculan al subconjunto específico de controles ASVS v5.0.0 cuyo cumplimiento es técnicamente exigible para satisfacer dichas obligaciones. El resultado es una matriz de requisitos que define el criterio de aceptación técnico para el diseño arquitectónico *to-be*.

3.1 Matriz de convergencia normativa-técnica

La Tabla 1 consolida el mapeo bidireccional entre las exigencias legales chilenas, el estándar internacional (HIPAA) y los controles de ingeniería de software de OWASP ASVS v5.0.0. Para cada requisito normativo identificado se asocia el control ASVS que demuestra, de forma técnica y reproducible, el cumplimiento de la obligación correspondiente. Esta matriz actúa como el criterio de aceptación técnico para el diagnóstico y el diseño arquitectónico.

Tabla 1. Mapeo de requerimientos legales a controles de seguridad ASVS

Fuente: Elaboración propia

ID requisito	Dominio clínico	Exigencia normativa	Control ASVS v5.0.0
REQ-001	Control de acceso estricto a PHI	Ley 20.584 (Arts. 12 y 13), HIPAA §164.312(a)(1)	V8.2.1 (L1), V8.2.2 (L1) : restricción de acceso a funciones y datos específicos (IDOR). V8.2.4 (L3) : Controles adaptativos de autorización.
REQ-002	Confidencialidad en reposo y cifrado de datos	Ley 19.628 (Art. 11), Ley 20.584 (Art. 12), HIPAA §164.312(a)(2)(iv)	V11.2.1 (L2) : uso de implementaciones criptográficas. V14.1.2 (L2) : documentación y aplicación formal de requisitos de protección y cifrado general para datos clasificados como sensibles.
REQ-003	Auditoría y trazabilidad	HIPAA §164.312(b)	V16.3.2 (L3) : registro ineludible de todas las decisiones de autorización y accesos de lectura a datos sensibles (PHI). V16.4.2 (L2) : protección absoluta de los <i>logs</i> contra accesos no autorizados y modificaciones (inmutabilidad).
REQ-004	Retención y eliminación de registros oncológicos	Ley 21.258 (Art. 8 bis)	V14.2.7 (L3) : Aplicación de clasificación de retención de datos sensibles, garantizando la eliminación automática de información obsoleta o innecesaria según plazos normativos.
REQ-005	Integridad de información	HIPAA §164.312(c)	V11.3.3 (L2) : protección de datos cifrados contra modificaciones no



			autorizadas mediante el uso de métodos de cifrado autenticado (ej. AES-GCM) o algoritmos MAC aprobados.
REQ-006a	Autenticación fuerte	HIPAA §164.312(d)	V6.3.3 (L2) : MFA basado en <i>hardware</i> para resistencia contra su-plantación.
REQ-006b	Identidad única de usuario	HIPAA §164.312(a)(2)(i)	V10.3.3 (L2) : identificación única mediante <i>tokens</i> (JWT claims 'iss' y 'sub').
REQ-007	Continuidad oncológica y alta disponibilidad	Ley 21.258 (Arts. 1 y 7)	V13.1.2 (L3), V15.2.2 (L2) : prevención de pérdida de disponibilidad por agotamiento de recursos.

El análisis del mapeo normativo indica que los dominios de control de acceso (REQ-001), autenticación (REQ-006) y auditoría (REQ-003) concentran la mayor densidad de exigencias legales concurrentes al ser referenciados simultáneamente por la legislación chilena y HIPAA §164.312. Esta convergencia los posiciona como los vectores de mayor riesgo regulatorio para SIGO. Por consiguiente, la evaluación de brechas (*gap analysis*) y el posterior rediseño (*to-be*) priorizarán la remediación de estas vulnerabilidades mediante los controles estrictos del nivel 3 de OWASP ASVS.

3.2 Segmentación de niveles de verificación y superficie de ataque

La definición del subconjunto de controles ASVS establecido en la sección anterior requiere contextualizar la superficie de ataque sobre la cual dichos controles operan. La Tabla 2 presenta el inventario de módulos funcionales de SIGO clasificados según la sensibilidad de los datos que procesan. Esta clasificación permite identificar qué componentes del sistema están directamente sujetos a las obligaciones normativas mapeadas en la Tabla 1, constituyendo así la superficie de ataque prioritaria para las fases de diagnóstico.

Tabla 2. Inventario de módulos, datos procesados y asignación de perfil ASVS

Fuente: Elaboración propia

Módulo SIGO	Datos procesados	Clasificación
Pacientes (fichas clínicas)	Información personal del paciente (RUT, nombres, fecha nacimiento, sexo), códigos CIE-10, historia clínica.	PHI
Comités médicos	Conclusiones clínicas conjuntas, escala ECOG, participantes.	PHI
Solicitudes de tratamiento	Procedimientos médicos detallados y en curso por paciente.	PHI
Bitácoras clínicas	Observación clínica (texto libre) del paciente, síntomas reportados.	PHI
Archivos médicos	Imágenes diagnósticas y reportes patológicos en formato PDF.	PHI

Exportación de datos	Funciones extractivas globales de las series de PHI.	PHI
Calendario (citas médicas)	RUT paciente, RUT tratante, temporalidad y motivo clínico.	PHI
Gestión de usuarios	PII del personal y mapeo de pacientes asignados.	PHI
<i>Dashboards</i>	Métricas de conteo, porcentajes, tasas de mortalidad.	No PHI
Síntomas (catálogo)	Metadatos referenciales y estáticos sin asociación a pacientes	No PHI

3.3 Alcance de los dominios técnicos de verificación

La Tabla 3 opera en un nivel de abstracción complementario al de la Tabla 2. Mientras la Tabla 2 clasifica los módulos funcionales de SIGO según la criticidad de los datos procesados, la Tabla 3 traslada esta jerarquía al plano técnico de verificación, estableciendo el nivel mínimo de exigencia y el alcance arquitectónico para cada dominio ASVS. Esta distinción metodológica resulta imperativa: un único módulo que procesa PHI exige la convergencia de controles de validación de entradas (V2), autorización (V8), criptografía (V11) y auditoría (V16). Cada dominio impone requerimientos técnicos verificables e independientes que deben estructurarse por separado.

Tabla 3. Dominios técnicos ASVS v5.0.0 y niveles de exigencia por alcance

Fuente: Elaboración propia

Dominio ASVS	Descripción	Nivel Mínimo	Alcance de aplicación
V2 <i>Validation</i>	Integridad de inputs y referencial	Nivel 3	Módulos PHI
V6 <i>Authentication</i>	Verificación de identidad	Nivel 3	Transversal a todos los módulos
V7 <i>Session management</i>	Gestión de JWT y <i>refresh tokens</i>	Nivel 3	Módulo de autenticación
V8 <i>Authorization</i>	RBAC y mínimo privilegio	Nivel 3	Módulos PHI
V11 <i>Cryptography</i>	AES-256-GCM y gestión de claves	Nivel 3	Módulos PHI
V12 <i>Secure communication</i>	TLS 1.2/1.3 y HSTS	Nivel 3	Toda comunicación cliente-servidor
V14 <i>Data protection</i>	Protección en memoria y reposo	Nivel 3	Módulos PHI
V16 <i>Logging</i>	<i>Audit trail</i> inmutable y forense	Nivel 3	Módulos PHI



4 Diagnóstico y modelado de amenazas

4.1 Arquitectura *as-is* de SIGO

SIGO se estructura bajo una arquitectura de tres capas: una *single page application* (SPA) en React para la capa de presentación, una API REST en Spring Boot como capa de lógica de negocio y acceso a datos, y una capa de persistencia mediante dos bases de datos MongoDB alojadas en MongoDB Atlas. La Tabla 4 detalla los componentes del sistema y sus roles arquitectónicos.

4.1.1 Componentes del sistema

Tabla 4. Componentes de la arquitectura *as-is* de SIGO

Fuente: Elaboración propia

Componente	Tecnología y versión	Rol en el sistema
<i>Frontend</i> SPA	React 18.3.1, React Router v6.23.1, Axios 1.7.2, Material-UI 6.1.4	Interfaz de usuario para personal médico y administrativo. Gestiona el token JWT en <i>localStorage</i> del navegador y aplica control de acceso por rol exclusivamente en el cliente.
<i>Backend</i> API REST	Spring Boot 3.3.2, Java 17, Spring Security, JWT 0.11.5, Jasypt 1.9.3	Punto de integración central: autenticación, lógica de negocio, acceso a datos clínicos y gestión de archivos médicos. Expone 12 controladores REST más el módulo de autenticación.
Base de datos principal	MongoDB Atlas, colección <i>sigo_database</i>	Persistencia de datos clínicos: pacientes, diagnósticos, bitácoras, solicitudes, <i>tickets</i> de comité médico, calendarios y usuarios del sistema.
Almacenamiento de archivos	MongoDB Atlas, GridFS, colección <i>imagenologia</i>	Almacenamiento de archivos médicos binarios (imágenes diagnósticas e informes en PDF) mediante el protocolo GridFS, con soporte de hasta 200 MB por archivo.

4.1.2 Flujos de datos principales

La Figura 1 presenta el diagrama de flujo de datos (DFD) de nivel 1 de la arquitectura *as-is* de SIGO. El modelo identifica actores, procesos, almacenes de datos y fronteras de confianza, y resume los tres flujos de mayor criticidad para la evaluación de seguridad. Este DFD y la descripción *as-is* se elaboraron sobre el estado del código en el repositorio GitHub (rama *audit/white-box-analysis*, *commit a882192*), referencia adoptada para la auditoría de caja blanca del presente informe.

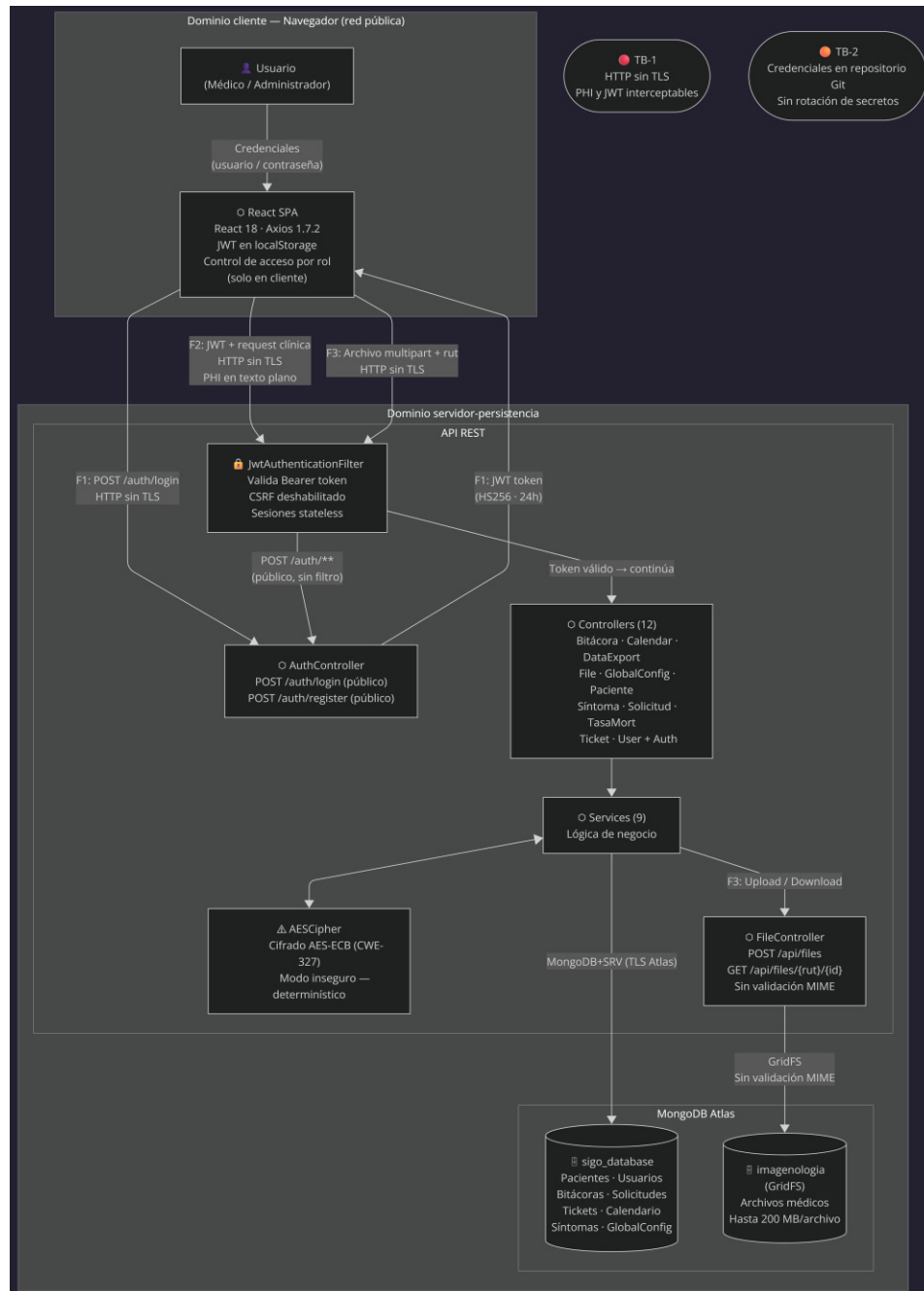


Figura 1. DFD Nivel 1

Fuente: Elaboración propia con Mermaid



En correspondencia con la Figura 1, a continuación, se describen los flujos F1 a F3.

Flujo F1: Autenticación y gestión de sesión. El cliente envía credenciales al *endpoint* público *POST /auth/login* sobre HTTP sin TLS en el tramo entre navegador y API REST. El *backend* valida los datos contra la colección de usuarios en *sigu_database* y devuelve un *token* JWT firmado (algoritmo HS256, vigencia 24 horas). El cliente almacena el *token* en *localStorage*. Las solicitudes posteriores incluyen el *token* en la cabecera *Authorization: Bearer <token>*, interceptada por *JwtAuthenticationFilter* en un esquema *stateless*. La configuración de Spring Security tiene la protección CSRF deshabilitada; dado que el modelo predominante es el *token* en cabecera y no una sesión por *cookie* sobre la API, la superficie típica de CSRF es menor que en aplicaciones basadas exclusivamente en *cookies*, pero la opción permanece como punto débil de endurecimiento si en el futuro se combinan *cookies* de autenticación o envíos tipo formulario HTML con los mismos *endpoints*.

Flujo F2: Operaciones sobre PHI. Este flujo agrupa las transacciones clínicas representativas del núcleo transaccional, incluyendo la gestión de pacientes, bitácoras, solicitudes, comités, citas y exportación de datos. El análisis arquitectónico de este flujo se estructura en tres capas complementarias:

- **Transporte:** La PHI se transfiere en texto plano a nivel aplicación en ese tramo HTTP sin TLS.
- **Autenticación y autorización en aplicación:** El componente *JwtAuthenticationFilter* comprueba únicamente la validez del *token* (*anyRequest().authenticated()*), sin controles de acceso basados en roles (RBAC) ni validación de propiedad o necesidad de conocer a nivel de registro. En consecuencia, cualquier usuario autenticado puede operar sobre la totalidad de las colecciones clínicas expuestas por la API en este modelo.
- **Persistencia:** La lógica de negocio emplea, para datos sensibles en reposo, cifrado AES en modo ECB (CWE-327); es decir, un modo determinístico que no proporciona confidencialidad semántica adecuada.

Flujo F3: Archivos médicos (GridFS): El componente *FileController* recibe archivos *multipart* mediante *POST /api/files*, con un tamaño máximo declarado de 200 MB, sobre el mismo tramo HTTP sin TLS entre cliente y API. Los archivos se almacenan en la base *imagenologia* mediante GridFS. La descarga se realiza con *GET /api/files/{rut}/{solicitudId}*: la respuesta se construye confiando en el tipo MIME registrado en la base de datos, sin validación técnica del contenido binario subyacente (por ejemplo, mediante firmas de tipo o *magic numbers*), lo que dificulta detectar sustitución de tipo o contenido no clínico presentado como documentación médica.

4.1.3 Fronteras de confianza

El análisis de la arquitectura *as-is* identifica dos fronteras de confianza (*trust boundaries*), definidas como los límites entre componentes con distintos niveles de privilegio o distintos dominios de control.

Tabla 5. Fronteras de confianza de la arquitectura *as-is*

Fuente: Elaboración propia

ID	Frontera	Protocolo	Mecanismo de control existente	Debilidad estructural identificada
TB-1	Navegador (red pública) → API REST	HTTP sin TLS	<i>Token</i> JWT en encabezado <i>Authorization</i> ; política CORS configurada en el servidor	Sin cifrado en tránsito: <i>tokens</i> JWT y PHI viajan en texto plano y son interceptables en cualquier entorno de red, incluyendo producción.
TB-2	API REST → MongoDB Atlas	MongoDB+SRV (TLS nativo de Atlas)	TLS gestionado por Atlas; autenticación mediante cadena de conexión	Ausencia de gestión segura de secretos: las credenciales de conexión no están protegidas mediante un meca-

nismo de rotación ni almacenadas en un gestor de secretos externo.

4.2 Análisis estático de código fuente (SAST)

4.2.1 Configuración y ejecución de herramientas

El análisis SAST comprendió un total de 16716 líneas de código no comentado (NCLOC), desglosadas en 4559 NCLOC para el *backend* desarrollado en Java y 12157 NCLOC para el *frontend* en React. El diagnóstico técnico se realizó mediante el uso complementario de SonarQube *Community Edition* (v26.4.0.121862 desplegado en contenedor Docker) y Semgrep OSS (v1.159.0). La Tabla 6 detalla la configuración técnica aplicada por componente.

Tabla 6. Configuración de herramientas SAST por componente

Fuente: Elaboración propia

Herramienta	Componente	Rule sets / Reglas activas	Archivos escaneados
SonarQube Community	<i>Backend</i>	Reglas Java: S6418, S5542, S4684, S4502, S4507 + perfil de seguridad activo	66 archivos Java
SonarQube Community	<i>Frontend</i>	Reglas JavaScript: S6418, S7013, S4822, S3923, S3403, S1534, S930 + perfil de seguridad activo	97 archivos JS/JSX
Semgrep OSS	<i>Backend</i>	p/java, p/owasp-top-ten, p/secrets	66 archivos Java
Semgrep OSS	<i>Frontend</i>	p/javascript, p/react, p/secrets	97 archivos JS/JSX

4.2.2 Resultados consolidados

Tabla 7. Resumen de resultados SonarQube por componente

Fuente: Elaboración propia

Componente	NCLOC	Vulnerabilidades	<i>Security hotspots</i>	<i>Code smells</i>
<i>Backend</i>	4559	11	39	218
<i>Frontend</i>	12157	2	2	585
Total	16716	13	41	803

Los 41 *security hotspots* identificados por SonarQube no constituyen vulnerabilidades confirmadas, sino patrones de diseño que requieren revisión manual para evaluar su explotabilidad en el contexto del sistema. La inspección del código fuente permitió caracterizar los hallazgos del *backend* (39 instancias) en dos riesgos principales. Primero, una instancia (regla java:S4502 en *Config/SecurityConfig.java*) deshabilita la protección CSRF sin medidas compensatorias documentadas (CWE-352). Aunque el impacto actual es bajo por el uso de *tokens Bearer*, esto constituye deuda de endurecimiento si el sistema incorpora *cookies* a futuro. Segundo, 38 instancias (regla java:S4507) utilizan *Throwable.printStackTrace()*. En entornos de producción, esto expone la jerarquía de clases y facilita la enumeración de la arquitectura interna al derivar las trazas a la salida estándar (CWE-489).

Para sintetizar el inventario de vulnerabilidades confirmadas por SonarQube (detalladas entre las filas S-01 y S-13 de la Tabla A-1 del anexo), conviene agrupar por CWE, tal como muestra la Tabla 8.

Tabla 8. Distribución de vulnerabilidades SonarQube por CWE

Fuente: Elaboración propia

CWE	Familia	N.º de hallazgos
CWE-798	Credenciales estáticas en código fuente	3
CWE-327	Uso de criptografía débil (AES-ECB)	2
CWE-915	<i>Mass assignment</i> / entidad expuesta como cuerpo de petición	8

La Tabla 9 documenta hallazgos *backend* asociados a AES-ECB en *AESCipher.java*, en coherencia con las filas S-04 y S-05 de SonarQube. La concordancia entre herramientas independientes refuerza la criticidad del hallazgo criptográfico.

Tabla 9. Hallazgos Semgrep *backend*

Fuente: Elaboración propia

Severidad	Regla	Archivo: Línea	Descripción	CWE
WARNING	ecb-cipher	AESCipher.java:18	AES-ECB — misma entrada produce misma salida cifrada	CWE-327
WARNING	use-of-aes-ecb	AESCipher.java:18	AES-ECB no garantiza confidencialidad del mensaje	CWE-327
WARNING	ecb-cipher	AESCipher.java:25	AES-ECB — segunda instancia en método decrypt	CWE-327
WARNING	use-of-aes-ecb	AESCipher.java:25	AES-ECB — segunda instancia (regla redundante)	CWE-327

Los cuatro registros Semgrep se reducen a dos hallazgos únicos (dos reglas redundantes por ubicación). El análisis Semgrep ejecutado sobre el *frontend* con los rulesets p/javascript, p/react y p/secrets no produjo hallazgos con severidad $\geq$ WARNING. Esta ausencia es coherente con el perfil de cobertura de la edición OSS. El resultado de 0 hallazgos no indica ausencia de riesgo en el *frontend*, sino el límite de cobertura técnico de la herramienta en su edición gratuita.

4.3 Análisis de composición de *software* (SCA)

El análisis de composición de *software* (SCA) evalúa la cadena de suministro mediante la inspección de dependencias externas directas y transitivas para identificar vulnerabilidades conocidas en repositorios públicos (*National Vulnerability Database*, GitHub Advisory, Google OSV). A diferencia del SAST, que analiza el código fuente escrito por el equipo de desarrollo, el SCA audita exclusivamente los componentes de terceros integrados en la arquitectura. Como artefacto de trazabilidad y monitoreo continuo, se generó una Lista de Materiales de Software (SBOM) bajo el estándar CycloneDX 1.6.

4.3.1 Configuración y ejecución de herramientas

Se ejecutaron tres herramientas SCA con roles diferenciados:

Tabla 10. Herramientas SCA seleccionadas

Fuente: Elaboración propia

Herramienta	Versión	Ecosistema cubierto	Rol en el análisis
Trivy (Aqua Security)	0.58.1	Maven (pom.xml + JARs compilados)	Detección de CVEs y generación de SBOM

npm audit	11.6.2	JavaScript (package-lock.json)	Detección de CVEs en dependencias Node.js
OWASP Dependency-Track	4.12.0	Multi-ecosistema (vía SBOM CycloneDX)	Monitoreo continuo; fuente de datos: Google OSV

Trivy fue elegido sobre las alternativas OWASP Dependency-Check y Snyk por tres razones concretas: (i) genera SBOM nativo en formato CycloneDX sin *plugins* adicionales, lo que permite importarlo directamente en Dependency-Track para monitoreo continuo sin transformaciones intermedias; (ii) su modelo de despliegue como contenedor Docker aísla el entorno de análisis del sistema local, garantizando reproducibilidad independientemente de la versión de JDK instalada; (iii) a diferencia de Snyk, no requiere registro en un servicio externo ni conectividad de red a una plataforma propietaria durante el análisis. El formato CycloneDX 1.6 fue elegido sobre SPDX 2.3 por alineación con el ecosistema OWASP: Dependency-Track implementa soporte nativo de primera clase para CycloneDX.

4.3.2 Dependencias *backend* – Trivy (Maven)

Trivy procesó 73 dependencias Maven del proyecto *backend*, identificando 53 CVEs:

Tabla 11. Distribución de CVEs por severidad (*backend*)

Fuente: Elaboración propia

Severidad	Cantidad
CRITICAL	4
HIGH	16
MEDIUM	24
LOW	9
Total	53

Las CVEs críticas y altas se concentran en tomcat-embed-core:10.1.26, spring-security-web:6.3.1 y spring-security-crypto:6.3.1. Su criticidad se amplifica por el acceso directo que estas librerías tienen sobre todos los *endpoints* que procesan PHI.

Tabla 12. CVEs críticas y altas seleccionadas (*backend*)

Fuente: Elaboración propia

CVE	Dependencia	Ver-sión	CVSS	Descripción	Módulo SIGO	Fix disponible
CVE-2025-24813	tomcat-embed-core	10.1.26	9.8	RCE potencial + divulgación de información	Todo el <i>backend</i>	10.1.35+
CVE-2024-50379	tomcat-embed-core	10.1.26	9.8	RCE vía condición de carrera TOCTOU en JSP	Todo el <i>backend</i>	10.1.34+
CVE-2026-29145	tomcat-embed-core	10.1.26	9.1	<i>Bypass</i> de autenticación CLIENT_CERT	Todo el <i>backend</i>	10.1.53+
CVE-2024-38821	spring-security-web	6.3.1	9.1	<i>Bypass</i> de autorización en recursos estáticos	<i>Endpoints</i> API	6.3.4+
CVE-2026-22732	spring-security-web	6.3.1	9.1	<i>Bypass</i> de política de seguridad	<i>Endpoints</i> API	6.5.9+
CVE-2024-56337	tomcat-embed-core	10.1.26	8.1	<i>Fix</i> incompleto de CVE-2024-50379	Todo el <i>backend</i>	11.0.2+
CVE-2025-22228	spring-security-crypto	6.3.1	7.4	<i>BCryptPasswordEncoder</i> sin límite de longitud	Autenticación	6.3.8+



CVE-2024-38816	spring-webmvc	6.1.11	7.5	Path traversal en aplicaciones Spring MVC	FileController	6.1.13+
CVE-2024-38819	spring-webmvc	6.1.11	7.5	Path traversal funcional (variante)	FileController	6.1.14+
CVE-2025-22235	spring-boot	3.3.2	7.3	Bypass de EndpointRequest.to() en Actuator	Actuator	3.3.11+

El SBOM generado por Trivy contiene 73 componentes Maven en formato CycloneDX 1.6, importados en OWASP Dependency-Track para análisis continuo:

Tabla 13. Resumen de vulnerabilidades en OWASP Dependency-Track

Fuente: Elaboración propia

Severidad	Cantidad
Critical	7
High	15
Medium	25
Low	8
Total	55

La diferencia entre las 53 CVEs de Trivy y las 55 de Dependency-Track obedece a la fuente de datos: Trivy utiliza Trivy-DB + GitHub Advisory, mientras Dependency-Track consultó exclusivamente Google OSV. Ambas fuentes son complementarias y la intersección confirma los hallazgos de mayor prioridad.

La evidencia visual de la pestaña de auditoría de vulnerabilidades de OWASP Dependency-Track se encuentra documentada en la Figura A-3 (disponible en el anexo). Este reporte detalla los niveles de severidad asociados a los componentes logback-core:1.5.6, jackson-core:2.17.2, tomcat-embed-core:10.1.26 y la suite de spring-security, ratificando las dependencias vulnerables identificadas previamente mediante el análisis con Trivy.

4.3.3 Dependencias frontend – npm Audit

El análisis npm procesó el grafo de dependencias del frontend (React 18), identificando 60 vulnerabilidades:

Tabla 14. Distribución de vulnerabilidades por severidad (frontend)

Fuente: Elaboración propia

Severidad	Cantidad
CRITICAL	1
HIGH	31
MODERATE	14
LOW	14
Total	60

Tabla 15. Vulnerabilidades npm críticas y altas seleccionadas
Fuente: Elaboración propia

Paquete	Rango vulnerable	CVSS	Descripción	Tipo de dependencia
form-data	3.0.0–3.0.3, 4.0.0–4.0.3	CRITICAL	Función <i>random</i> insegura en generación de <i>boundary</i>	Transitiva
serialize-javascript	≤7.0.4	8.1	RCE vía expresión regular en serialización	Transitiva (<i>build</i>)
@remix-run/router	≤1.23.1	8.0	XSS vía <i>Open Redirect</i> en React Router	Transitiva
axios	1.0.0–1.14.0	HIGH	<i>Server-Side Request Forgery</i> (SSRF)	Directa — axios ^1.7.2
body-parser	≤1.20.3	7.5	DoS por URL malformada	Transitiva
cross-spawn	7.0.0–7.0.4	7.5	ReDoS en procesamiento de argumentos	Transitiva
nth-check	<2.0.1	7.5	ReDoS en procesamiento CSS	Transitiva
path-to-regexp	≤0.1.12	7.5	ReDoS vía <i>backtracking</i>	Transitiva
react-router-dom	6.0.0–6.30.2	6.5	<i>Open Redirect</i> inesperado	Directa — react-router-dom ^6.23.1
lodash	≤4.17.23	6.5	<i>Prototype pollution</i>	Transitiva

El análisis de dependencias del entorno cliente identifica vulnerabilidades de alta severidad en *axios* versión 1.7.2 y *react-router-dom* versión 6.23.1, ambas dependencias directas declaradas en la configuración del proyecto. La vulnerabilidad SSRF en *axios* representa un riesgo arquitectónico crítico, dado que esta librería centraliza la totalidad de las peticiones HTTP desde el cliente hacia la API REST. Asimismo, el defecto de severidad crítica identificado en la librería *form-data* compromete la estructuración técnica en las peticiones *multipart*, un fallo que impacta directamente el flujo F3 de carga de archivos médicos y habilita vectores de ataque sobre la información de salud protegida durante su tránsito.

4.4 Modelado de Amenazas STRIDE

El modelado de amenazas se aplica sobre el DFD de Nivel 1 presentado en la Figura 1 de la sección anterior, extendiendo el análisis a cada flujo de datos (F1, F2 y F3) y a las fronteras de confianza identificadas (TB-1 y TB-2). La metodología STRIDE actúa como técnica complementaria a SAST y SCA: mientras estos últimos detectan defectos puntuales en el código fuente y en las dependencias externas, STRIDE evalúa la lógica de negocio y las interacciones entre componentes desde la perspectiva del atacante, identificando vectores de amenaza que los escáneres automatizados omiten por carecer de contexto operacional.

Para cada elemento del modelo se identifican las categorías STRIDE aplicables, se describe el impacto diferencial sobre la arquitectura y se vincula, cuando corresponde, con hallazgos técnicos previos referenciados por su identificador SAST (S-XX) o SCA (CVE). Los hallazgos sin antecedente en secciones anteriores representan amenazas de diseño detectadas exclusivamente por este análisis. Los identificadores TM-XX y TA-XX son propios de esta sección. El inventario completo de las 22 instancias identificadas se detalla en la Tabla A-2 del Anexo.

4.4.1 Perfiles de actores de amenaza

El modelo de amenazas define cuatro perfiles de atacante derivados de la arquitectura de roles de SIGO y del contexto de despliegue del sistema. La Tabla 16 caracteriza cada actor según su posición respecto a las fronteras de confianza, sus capacidades técnicas asumidas y el vector de acceso inicial disponible. Esta clasificación opera como criterio de priorización: una amenaza ejecutable por un actor externo no autenticado desde la



red pública tiene mayor exposición regulatoria que un vector que requiere credenciales internas válidas y acceso físico a la red hospitalaria.

Tabla 16. Perfiles de actores de amenaza sobre la arquitectura *as-is* de SIGO

Fuente: Elaboración propia

ID	Actor	Posición	Capacidades asumidas	Vector de acceso inicial
TA-01	Atacante externo no autenticado	Fuera de TB-1 (red pública)	Intercepción de tráfico de red (MITM), escaneo de <i>endpoints</i> públicos, explotación de CVEs conocidas	HTTP sin TLS sobre TB-1; <i>endpoints</i> públicos <i>POST /auth/login</i> , <i>POST /auth/register</i>
TA-02	Usuario clínico interno – rol: MEDICO	Dentro de TB-1, autenticado	<i>Token</i> JWT válido, conocimiento del dominio clínico, acceso a <i>endpoints</i> API mediante cliente React	Credenciales válidas; acceso irrestricto a la API por ausencia de RBAC en servidor
TA-03	Usuario privilegiado interno – rol: GESTOR/ADMIN	Dentro de TB-1, autenticado	<i>Token</i> JWT válido, acceso a funciones administrativas y de exportación, conocimiento de la estructura de datos	Credenciales válidas; acceso a módulos de exportación masiva de PHI
TA-04	Actor interno con acceso al repositorio	Dentro del perímetro de desarrollo	Lectura del código fuente, acceso a variables de entorno y cadenas de conexión fijadas estáticamente.	Acceso al repositorio GitHub (rama <i>audit/white-box-analysis</i>)

TA-04 representa el vector de mayor impacto estructural del sistema: la exposición del repositorio compromete simultáneamente el secreto JWT (S-01) y las credenciales de conexión a MongoDB, colapsando las dos fronteras de confianza sin requerir interacción con el sistema en ejecución.

4.4.2 Flujo F1: Autenticación y gestión de sesión

La presencia de la clave secreta del JWT incrustada en el código fuente (S-01, CWE-798) habilita la amenaza de **suplantación** de mayor impacto estructural del sistema (TM-01): **TA-04**, con acceso al repositorio, puede forjar *tokens* con identidad arbitraria sin requerir credenciales válidas, anulando por completo el mecanismo de autenticación desde el interior. Este vector es indetectable por análisis de comportamiento externo y solo visible desde la auditoría *white box*.

La transmisión de credenciales y *token* sobre HTTP sin TLS en TB-1 constituye un vector de **divulgación** explotable por **TA-01** (TM-04): la interceptación pasiva en cualquier entorno de red expone la sesión completa sin requerir autenticación previa. El almacenamiento del *token* en *localStorage* amplía esta superficie al hacerlo accesible para *scripts* arbitrarios ante inyección (TM-05), combinando dos debilidades de diseño independientes en un único vector de compromiso de sesión.

La ausencia de controles de tasa en *POST /auth/login* habilita ataques de fuerza bruta por **TA-01** (**denegación**, TM-06). CVE-2025-22228 en *spring-security-crypto:6.3.1* (CVSS 7.4) amplifica este vector al eliminar el límite de longitud del codificador BCrypt, habilitando el agotamiento computacional del servidor mediante contraseñas de longitud arbitraria.

4.4.3 Flujo F2: Operaciones sobre PHI

La ausencia de RBAC en servidor documentada en §4.1.2 (F2) habilita **suplantación** de contexto clínico explotable por **TA-02** y **TA-03** (TM-07): cualquier credencial válida se convierte en acceso irrestricto sobre la totalidad de la colección PHI sin validación de propiedad ni necesidad de conocer. La distinción entre actores es relevante para la priorización: TA-02 representa el escenario de credencial médica comprometida con acceso lateral a registros de pacientes no asignados; TA-03 representa el abuso de privilegio interno con acceso a funciones de exportación masiva de PHI. Ambos vectores tienen alcance transversal sobre los ocho módulos PHI identificados en la Tabla 2.

La carencia de registros persistentes para accesos de lectura y escritura sobre PHI constituye la amenaza de **repudio** de mayor impacto normativo (TM-10), explotable por cualquier actor autenticado (TA-02, TA-03): ante la ausencia de *audit trail*, ninguna acción sobre PHI es atribuible de forma forense, incumpliendo V16.3.2 de ASVS v5.0.0 y REQ-003 de la Tabla 1.

CVE-2024-38821 y CVE-2026-22732 en spring-security-web:6.3.1 (CVSS 9.1 ambas) materializan amenazas activas de **elevación de privilegios** explotables por **TA-01** desde la red pública (TM-15): habilitan el *bypass* de la capa de autorización sobre *endpoints* de la API sin requerir credenciales válidas, convirtiendo un vector externo en acceso equivalente al de TA-03 sobre módulos PHI.

4.4.4 Flujo F3: Archivos médicos (GridFS)

La ausencia de validación binaria documentada en §4.1.2 (F3) habilita **suplantación** de tipo de contenido explotable por **TA-02** o **TA-03** con *token* válido (TM-16): sin verificación mediante *magic numbers*, un archivo no clínico puede inyectarse como documentación diagnóstica válida, comprometiendo la integridad de la ficha clínica sin modificar sus metadatos. Este vector opera en la capa semántica del dato y es indetectable por análisis estático.

La admisión de archivos de hasta 200 MB sin limitación de tasa configura un vector de **denegación de servicio** por agotamiento de recursos explotable por **TA-01** sin autenticación (TM-20), cuya explotabilidad se amplifica por CVE-2025-24813 y CVE-2024-50379 en tomcat-embed-core:10.1.26 (CVSS 9.8 ambas).

4.4.5 Fronteras de confianza

El TLS nativo de Atlas protege el canal de transporte, pero la cadena de conexión con credenciales no está gestionada por un servicio de secretos externo ni sujeta a rotación (Tabla 5). En convergencia con S-01 (CWE-798), **TA-04** con acceso al repositorio compromete simultáneamente el mecanismo de autenticación JWT y el acceso directo a la colección PHI (TM-21), sin interacción con el sistema en ejecución y sin dejar trazas en ningún *log* de aplicación.

4.4.6 Síntesis consolidada

Tabla 17. Distribución de amenazas STRIDE por categoría y actor principal

Fuente: Elaboración propia

Categoría STRIDE	Instancias	Actor principal	Flujos / Fronteras afectados
Suplantación (S)	3	TA-01, TA-02, TA-04	F1, F2, F3
Manipulación (T)	6	TA-01, TA-02, TA-03, TA-04	F1/TB-1, F2, F3, TB-2
Repudio (R)	2	TA-02, TA-03	F1, F2
Divulgación (I)	6	TA-01, TA-04	F1, F2/TB-1, F3/TB-1, TB-2
Denegación (D)	3	TA-01	F1, F2, F3
Elevación (E)	2	TA-01, TA-02	F2



Total **22**

4.5 Consolidación del diagnóstico

El diagnóstico de caja blanca integra tres técnicas complementarias aplicadas sobre la arquitectura *as-is* de SIGO: 13 vulnerabilidades confirmadas por SAST (S-01 a S-13), vulnerabilidades de terceros identificadas mediante SCA (4 de severidad crítica y 16 de severidad alta en el *backend*; 1 crítica y 31 de severidad alta en el *frontend*) y 22 amenazas de diseño modeladas a través de STRIDE (TM-01 a TM-22). La complementariedad entre los métodos seleccionados no es redundante. El análisis SAST detecta defectos puntuales en el código fuente escrito por el equipo de desarrollo. El SCA audita los componentes de terceros integrados en la cadena de suministro. STRIDE evalúa la lógica de negocio y los flujos completos para identificar vectores que los escáneres automatizados omiten por carecer de contexto operacional.

El patrón de mayor criticidad estructural radica en la amplificación entre hallazgos de diversa naturaleza. El secreto JWT almacenado en el código fuente (S-01, CWE-798) expone la clave de firma que valida la identidad de los actores del sistema. En convergencia con la cadena de conexión de MongoDB también expuesta en el repositorio (TM-21, TM-22), el actor TA-04 puede comprometer simultáneamente el mecanismo de autenticación (V13.3.1) y el acceso directo a la colección PHI en Atlas. Este compromiso se produciría sin interacción con el sistema en ejecución y sin dejar trazas en los registros de aplicación. Esta convergencia entre un defecto de código detectado por SAST y dos amenazas de diseño identificadas por STRIDE genera un impacto que ninguna técnica aislada hubiera caracterizado íntegramente.

La distribución de hallazgos por dominio técnico ASVS se presenta en la Tabla 17. Esta evidencia indica que los ocho dominios de verificación definidos en la Tabla 3 registran al menos un hallazgo confirmado. Los dominios de autorización (V8), comunicación segura (V12) y auditoría (V16) concentran la mayor densidad de amenazas activas. Dicha distribución guarda coherencia con la priorización regulatoria establecida en la Tabla 1. Los requisitos REQ-001, REQ-003 y REQ-006 son los que concentran las exigencias concurrentes de la legislación chilena y HIPAA §164.312(a)(1), §164.312(b) y §164.312(d). La convergencia entre la evidencia técnica y la priorización normativa previa confirma la pertinencia del subconjunto de controles ASVS seleccionado en el Capítulo 3 y define el criterio de entrada para el análisis de brechas del Capítulo 5.

Tabla 18. Distribución consolidada de hallazgos por dominio ASVS

Fuente: Elaboración propia

Dominio ASVS	Hallazgos SAST	Amenazas STRIDE	REQ asociado
V2 Validación	S-06–S-13 (CWE-915)	TM-08, TM-16, TM-18	—
V6 Autenticación	S-01–S-03 (CWE-798)	TM-01, TM-06	REQ-006a/b
V7 Gestión de sesión	—	TM-05	REQ-006b
V8 Autorización	—	TM-07, TM-14, TM-15	REQ-001
V11 Criptografía	S-04, S-05 (CWE-327)	TM-09, TM-12	REQ-002, REQ-005
V12 Comunicación segura	—	TM-02, TM-04, TM-11, TM-13, TM-19, TM-20	REQ-002, REQ-007
V14 Protección de datos	—	TM-21, TM-22	REQ-002, REQ-004
V16 Auditoría	—	TM-03, TM-10	REQ-003

5 Evaluación de brechas de seguridad: cumplimiento ASVS y priorización de riesgos

Este capítulo documenta la evaluación formal de brechas (*gap analysis*), contrastando el estado de seguridad de SIGO, diagnosticado en el capítulo anterior, contra el subconjunto trazable de controles OWASP ASVS v5.0.0 definido en el Capítulo 3. El análisis recorre los ocho dominios técnicos estructurados en la Tabla 3 y los siete requisitos normativos de la Tabla 1, evaluando cada control según su estado de implementación efectiva en la arquitectura *as-is* (*commit a882192*). Los resultados obtenidos en esta fase constituyen el insumo técnico directo para fundamentar el rediseño arquitectónico del Capítulo 6.

5.1 Método de evaluación

5.1.1 Criterios de cumplimiento

Cada control ASVS evaluado recibe uno de los siguientes estados operacionales:

- **Cumple (C):** Implementación técnica completa, efectiva y verificable en la arquitectura actual.
- **Parcial (P):** El mecanismo existe, pero posee debilidades estructurales o primitivas criptográficas obsoletas que comprometen el nivel exigido.
- **No Cumple (NC):** Ausencia de control o implementación ineficaz, vinculada obligatoriamente a un hallazgo (S-XX, CVE o TM-XX) del Capítulo 4.
- **No Aplicable (N/A):** Control arquitectónicamente irrelevante para el patrón de implementación de SIGO, con justificación técnica explícita.

5.1.2 Cuantificación del riesgo y regla de elevación normativa

El nivel de riesgo final para cada brecha se calcula mediante el cruce de la severidad técnica internacional (CVSS v3.1) y la consecuencia regulatoria dentro del entorno oncológico (PHI), parametrizado analíticamente en la Tabla 19.

Tabla 19. Matriz de riesgo ordinal: Severidad Técnica × Impacto Clínico (PHI)

Fuente: Elaboración propia

Severidad Técnica (CVSS v3.1)	PHI Alto	PHI Medio	PHI Bajo
Crítica (≥ 9.0)	Crítico	Crítico	Alto
Alta (7.0–8.9)	Crítico	Alto	Medio
Media (4.0–6.9)	Crítico	Medio	Bajo
Baja (0.1–3.9)	Crítico	Bajo	Bajo

Los dos ejes que parametrizan la matriz de riesgo se definen operacionalmente bajo los siguientes criterios:

- **Severidad Técnica:** Corresponde a la puntuación base del estándar *Common Vulnerability Scoring System* (CVSS v3.1) [22] del hallazgo indexado en el diagnóstico del Capítulo 4. Para las debilidades de diseño o fallas de código fuente (SAST) que carecen de un identificador CVE formal asignado por un proveedor de *software*, se emplea la puntuación de referencia histórica de la vulnerabilidad raíz (*Common Weakness Enumeration - CWE*) registrada en la base de datos de la *National Vulnerability Database* (NVD) [26] (véase la metodología detallada en la Tabla B-3 del Anexo B). Los rangos se categorizan en: **Baja** (0.1–3.9), **Media** (4.0–6.9), **Alta** (7.0–8.9) y **Crítica** (≥ 9.0).
- **Impacto Clínico (PHI):** Clasificación jerárquica derivada del inventario de flujos y tipologías de datos estructurado en la Tabla 2 del Capítulo 3:



- **Alto:** La brecha técnica intercepta o se ejecuta sobre componentes de *software* que procesan de manera directa información de salud protegida (*Protected Health Information*), tales como diagnósticos clínicos o decisiones de comités médicos. Su exposición activa un escenario de incumplimiento legal transgresor de las leyes chilenas N.º 20.584 y N.º 19.628.
- **Medio:** Componentes orientados al procesamiento de metadatos clínicos indirectos, trazas de auditoría de infraestructura o datos de identificación personal (*Personally Identifiable Information*) de los usuarios internos del sistema.
- **Bajo:** Módulos lógicos funcionalmente desacoplados de la persistencia de datos clínicos de pacientes, restringidos exclusivamente a servicios de presentación agregada (como *dashboards* gerenciales) o catálogos estáticos de uso público.

Como se observa en la matriz, la presencia de atributos clasificados como **PHI Alto** actúa como un operador de umbral absoluto que anula la variabilidad de la probabilidad técnica de explotación, homologando todas las brechas no conformes en la categoría de **Riesgo Crítico**.

Esta decisión metodológica modela de manera exacta la concurrencia legal de las leyes chilenas N.º 20.584 y N.º 19.628, junto al estándar internacional HIPAA §164.312. Ninguno de estos marcos normativos supedita las sanciones institucionales o el daño ético al paciente a la complejidad del vector informático utilizado: la mera existencia de una vía de exposición de datos oncológicos protegidos constituye un quiebre legal inmediato de consecuencia máxima. Debido a que la arquitectura *as-is* de SIGO está totalmente acoplada y carece de segmentación funcional o de red en el *backend*, la condición por PHI Alto se despliega transversalmente sobre el ecosistema. Dentro de este espectro homogeneizado en nivel Crítico, la puntuación base CVSS v3.1 opera como el criterio técnico secundario de desempate y ordenamiento para fijar las prioridades físicas de reingeniería.

5.2 Matriz densa de *gap analysis*

La Tabla 20 consolida la distribución cuantitativa de las brechas detectadas en el sistema SIGO, agrupadas por dominios de verificación OWASP ASVS v5.0.0 y vinculadas a sus respectivos identificadores de brecha (*Gap ID*). La definición completa de cada brecha (control ASVS, hallazgo vinculado, CVSS v3.1 y nivel de riesgo $R(g)$) se registra en la Tabla B-1 del Anexo B, que constituye la fuente primaria de trazabilidad del análisis.

Tabla 20. Estado de cumplimiento y volumetría de brechas por dominio ASVS v5.0.0 Fuente: Elaboración propia

Fuente: Elaboración propia

Dominio ASVS v5.0.0	Alcance Operacional	Cumple	Parcial	No Cumple	Brechas Asociadas (GAP-ID)	Nivel de Riesgo Base
V2: Validación	Módulos PHI	0	0	3	GAP-18, GAP-19, GAP-20	Crítico
V6: Autenticación	Transversal	0	0	4	GAP-01, GAP-02, GAP-03, GAP-04	Crítico
V7: Gestión de Sesión	Módulo Autenticación	0	0	1	GAP-05	Crítico (elevado)
V8: Autorización	Módulos PHI	0	0	3	GAP-06, GAP-07, GAP-08	Crítico
V11: Criptografía	Módulos PHI	0	0	2	GAP-09, GAP-10	Crítico
V12: Comunicación	Cliente-Servidor-DB	1	0	3	GAP-11, GAP-12 (Cumple), GAP-13, GAP-14	Crítico
V14: Protección de Datos	Módulos PHI	0	1	2	GAP-15 (parcial), GAP-16, GAP-17	Crítico

V16: Auditoría	Módulos PHI	0	0	2	GAP-21, GAP-22	Crítico (elevado)
Totales		1	1	20		21 críticos / 0 altos

5.3 Análisis de brechas estructurales por vectores de convergencia

El análisis cruzado de los datos de la Tabla 20 demuestra un colapso sistémico de los controles de seguridad exigidos por el nivel de verificación 3 (L3) de OWASP ASVS. La deuda técnica acumulada en SIGO no se manifiesta a través de vulnerabilidades aisladas, sino mediante vectores de convergencia donde fallas de código fuente (SAST) y dependencias desactualizadas (SCA) activan de manera simultánea amenazas lógicas de diseño arquitectónico (STRIDE) mapeadas en el Capítulo 4.

En el eje de **Autenticación y Gestión de Sesión (V6, V7)**, la presencia de la clave secreta de JWT incrustada en el código fuente de *JwtService.java* (S-01) junto con la exposición de la cadena de conexión de MongoDB Atlas en el repositorio público (TM-21) anula completamente la frontera de confianza TB-2. Un atacante externo con perfil de amenaza de red (TA-04) posee la capacidad técnica de forjar de manera autónoma *tokens* con identidades y privilegios arbitrarios, accediendo directamente al almacén de persistencia sin interactuar con la lógica del *backend*. Esta condición se ve agravada por la ausencia de un segundo factor de autenticación (GAP-02) y la falta de mecanismos de atenuación de tasa (*rate-limiting*) en el *endpoint POST/auth/login* (GAP-03), vulnerabilidad expuesta por el algoritmo de *hash* síncrono BCrypt sujeto a denegación de servicio computacional (CVE-2025-22228). Asimismo, el almacenamiento del JWT en el *LocalStorage* del navegador del cliente (GAP-05), expone el identificador de sesión a la extracción pasiva mediante ataques de inyección de scripts (XSS).

El vector de **Autorización (V8) y Validación de Entradas (V2)** evidencia una ausencia absoluta de validaciones en el lado del servidor. Al delegar el control de acceso de manera exclusiva a las rutas de la SPA en React, la API REST expone 8 controladores clínicos (S-06 a S-13) que carecen de verificación de políticas sobre los parámetros de entrada. Esto faculta escenarios de asignación masiva de parámetros (*Mass Assignment* / CWE-915), donde cualquier usuario autenticado de nivel administrativo (TA-03) o médico (TA-02) puede mutar atributos sensibles, tales como la asignación de oncólogos, el estado de solicitudes del comité o estadios tumorales, alterando directamente el *payload* JSON. Esta insuficiencia de validación se extiende al subsistema de almacenamiento GridFS (GAP-20), donde la omisión de validaciones semánticas sobre el encabezado real de los archivos binarios (*magic numbers*) faculta la inyección de *payloads* maliciosos en la colección *imagenologia*, utilizando tipos MIME falsificados.

A nivel de **Protección de Datos, Criptografía e Integridad (V11, V14, V16)**, el uso del algoritmo AES en modo de operación ECB (S-04, S-05) destruye la confidencialidad semántica de los datos en reposo. Al tratarse de un esquema de cifrado determinístico, bloques de texto claro idénticos (como códigos de diagnóstico CIE-10 recurrentes) generan bloques de texto cifrado idénticos. Un atacante pasivo con acceso a la base de datos puede inferir diagnósticos oncológicos e historiales clínicos mediante análisis de frecuencia estadística, rompiendo el mandato de secreto estipulado en el Artículo 12 de la Ley N.º 20.584 y las salvaguardas técnicas de HIPAA §164.312(c). Finalmente, esta condición estructural se vuelve irreversible ante la falta de trazas de auditoría inmutables (GAP-21, GAP-22): la API no registra los accesos de lectura (*read logs*) sobre los recursos PHI, produciendo un escenario de no-repudio negativo que impide auditorías forenses externas.

5.4 Ranking priorizado de quiebres estructurales para la reingeniería

La Tabla 21 condensa los cinco quiebres estructurales de la arquitectura *as-is* que determinan, de forma directa y restrictiva, las decisiones de diseño obligatorias del Capítulo 6. La selección aplica dos criterios jerárquicos: (1) cobertura máxima de capas arquitectónicas distintas (infraestructura, control de acceso, gestión de identidad, transporte y trazabilidad normativa) y (2) CVSS(*g*) descendente como desempate dentro de la



misma capa. Las 16 brechas restantes, todas en nivel Crítico, se abordan como controles derivados en la propuesta arquitectónica del Capítulo 6.

Tabla 21. Quiebres estructurales de la arquitectura *as-is*: vectores de entrada al diseño *to-be*

Fuente: Elaboración propia

P.	GAP-ID	Capa arquitectónica	CVSS v3.1	REQ	Hallazgo vinculado	Impacto estructural
1	GAP-13	Servidor de aplicaciones	9.8	REQ-007	TM-13, TM-20; CVE-2025-24813, CVE-2024-50379	RCE y DoS sin autenticación sobre Tomcat embebido; compromete la disponibilidad total del <i>backend</i> y la continuidad de la atención.
2	GAP-06	Control de acceso (RBAC)	9.1	REQ-001	TM-07, TM-15; CVE-2024-38821	Ausencia de RBAC en servidor: credencial válida $\equiv$ acceso irrestricto a totalidad de colecciones PHI.
3	GAP-01	Gestión de identidad	8.8	REQ-006a	S-01 (CWE-798); TM-01	JWT <i>secret</i> estático: forja de <i>tokens</i> con identidad arbitraria desde repositorio; anula autenticación.
4	GAP-07	Lógica de negocio	8.1 (est.)	REQ-001	TM-07; CWE-639	Ausencia de validación de propiedad de registros: usuario clínico legítimo puede extraer/modificar PHI ajena.
5	GAP-11	Capa de transporte	7.5 (est.)	REQ-002	TM-02, TM-04, TM-11, TM-19	HTTP sin TLS en TB-1: PHI, credenciales e imágenes diagnósticas transmitidas en texto plano sobre todos los flujos.

Se establece metodológicamente que la mitigación de los componentes de prioridad 1, 2 y 3 mediante patrones arquitectónicos estructurales es mandatoria para asegurar la viabilidad operativa y la supervivencia legal mínima de la plataforma SIGO antes de proceder con el endurecimiento de los controles de auditoría y almacenamiento secundario.

5.5 Síntesis y criterio de cumplimiento *gap analysis*

El *gap analysis* evaluó 22 controles ASVS v5.0.0 en los ocho dominios técnicos de la Tabla 3; la especificación completa de cada brecha (GAP-01 a GAP-22) se detalla en la Tabla B-1 del Anexo B. La distribución cuantitativa de estados arrojó 20 controles No Cumple, 1 control Parcial (GAP-15, AES-ECB sin confidencialidad semántica) y 1 control Cumple (GAP-12, TLS nativo en TB-2 Atlas). Al aplicarse la parametrización de riesgo de la Tabla 19, la ausencia de fronteras lógicas de aislamiento provoca que el riesgo regulatorio absorbe la severidad transaccional, situando las 21 brechas no conformes en nivel Crítico.

Ninguno de los ocho dominios ASVS evaluados alcanza el estado de cumplimiento completo en la plataforma actual. Los dominios V8 (Autorización), V12 (Comunicación) y V6 (Autenticación) concentran la mayor densidad de vulnerabilidades estructurales activas, en estricta coherencia con la carga normativa acumulada en los requisitos REQ-001, REQ-002 y REQ-006 de la Tabla 1. Los cinco quiebres arquitectónicos priorizados en la Tabla 21 determinan el subconjunto de restricciones técnicas de entrada que el Capítulo 6 resolverá obligatoriamente mediante patrones de diseño seguros para alcanzar el perfil institucional *to-be*, dando cumplimiento formal al cuarto objetivo específico de este trabajo de memoria.

6 Arquitectura de seguridad *to-be*: controles de mitigación por defensa en profundidad y validación de cumplimiento normativo en SIGO.

Este capítulo presenta el diseño de la arquitectura de seguridad *to-be* para la plataforma SIGO, derivado directamente de las 21 brechas de nivel Crítico identificadas en el Capítulo 5. El diseño propuesto opera bajo el principio de defensa en profundidad y especifica, para cada dominio de verificación ASVS v5.0.0 evaluado, los controles técnicos capaces de cerrar las brechas detectadas y satisfacer los siete requisitos normativos definidos en la Tabla 1. La efectividad del diseño se demuestra mediante una matriz de trazabilidad que conecta cada brecha (GAP-ID) con el control propuesto, el control ASVS v5.0.0 satisfecho y la norma legal cumplida, dando cumplimiento formal al quinto objetivo específico de este trabajo de memoria.

6.1 Principios arquitectónicos del rediseño

La fragilidad arquitectónica diagnosticada en el Capítulo 5 no es el resultado de fallas aisladas sino de la ausencia de principios de diseño seguro desde la concepción del sistema. La arquitectura *to-be* se fundamenta en cinco principios que responden directamente a los vectores de convergencia identificados en el análisis de brechas estructurales.

1. **Defensa en profundidad:** ningún control único protege la totalidad de la superficie PHI. El diseño propone capas de control independientes (perímetro de red, autenticación, autorización, cifrado y auditoría) de modo que la falla de cualquier capa individual no comprometa los datos clínicos. Este principio contrarresta directamente el acoplamiento total de la arquitectura *as-is*, donde la exposición del secreto JWT (GAP-01) anula simultáneamente autenticación, autorización y trazabilidad.
2. **Zero trust:** toda solicitud entrante es tratada como no confiable hasta que se verifican de forma independiente: la identidad del actor (autenticación), el rol que posee sobre el recurso solicitado (autorización RBAC) y la propiedad del recurso accedido (validación de *ownership*). Este principio responde a GAP-06 y GAP-07, donde un *token* válido otorgaba acceso irrestricto a la totalidad de las colecciones PHI sin validación de contexto.
3. **Mínimo privilegio:** cada componente opera con los permisos estrictamente necesarios para su función. Los controladores de negocio no tienen acceso de escritura a la colección *audit_logs*; los servicios de autenticación no tienen acceso a colecciones clínicas; la clave privada de firma JWT no es accesible desde el código de aplicación. Este principio elimina estructuralmente los escenarios de desplazamiento lateral documentados en la Tabla 21.
4. **Fail secure:** ante condiciones anómalas o fallo de un componente, el sistema deniega el acceso y registra el evento, sin exponer información de diagnóstico. Este principio contrarresta la ausencia de *audit trail* documentada en GAP-21 y GAP-22, donde la inexistencia de registros de acceso producía escenarios de no-repudio negativo.
5. **Cifrado en múltiples capas:** La PHI se cifra tanto en tránsito como en reposo mediante primitivas criptográficas aprobadas. TLS 1.3 protege todos los flujos de la frontera de confianza TB-1 (cliente-API); AES-256-GCM con vector de inicialización (IV) aleatorio por operación reemplaza el modo ECB documentado en GAP-04, GAP-09 y GAP-10. Este principio elimina el riesgo de análisis de frecuencia estadística sobre diagnósticos oncológicos repetidos.

6.2 DFD de la arquitectura *to-be*

La Figura 2 presenta el diagrama de flujo de datos (DFD) de nivel 1 de la arquitectura *to-be*, al mismo nivel de abstracción que la Figura 1 del Capítulo 4. La comparación directa entre ambos diagramas evidencia los cambios estructurales introducidos por el rediseño.

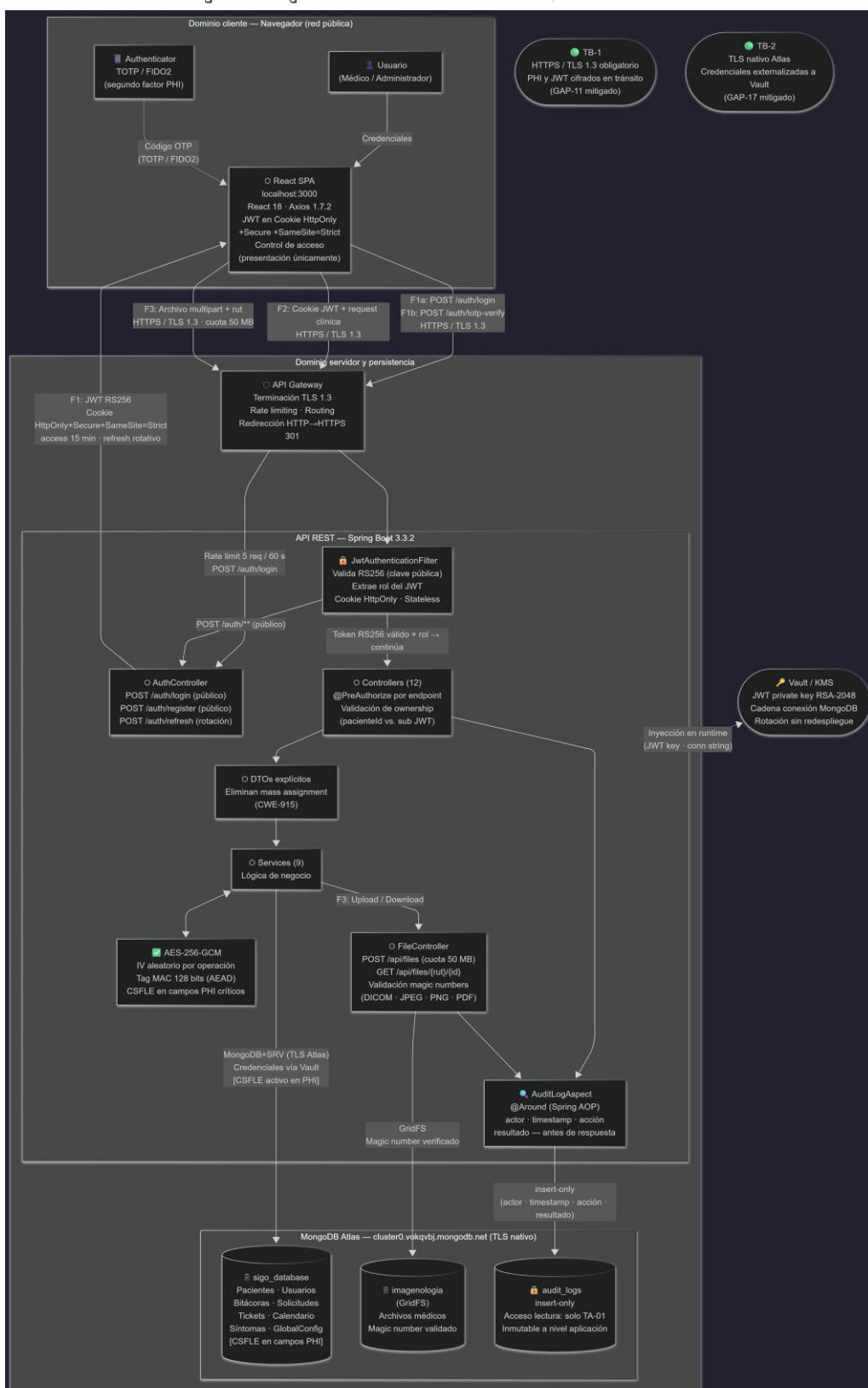


Figura 2. DFD arquitectura to-be SIGO
Fuente: Elaboración propia con Mermaid

6.3 Cambios estructurales respecto a la arquitectura *as-is*

Los cambios principales respecto al DFD *as-is* son los siguientes:

- **API Gateway como punto único de entrada:** se introduce un *API Gateway* delante del *backend* Spring Boot. Centraliza la terminación TLS 1.3 para la frontera TB-1 (elimina GAP-11), *rate limiting* sobre los *endpoints* de autenticación y carga de archivos (elimina GAP-03 y GAP-14), y enrutamiento hacia los controladores internos. El *backend* deja de ser accesible directamente desde el cliente.
- **Gestor de secretos externo (Vault/KMS):** El par de claves RSA para la firma JWT y la cadena de conexión de MongoDB Atlas se migran a HashiCorp Vault o AWS KMS. Se elimina la persistencia de credenciales en el código fuente y en las variables de entorno del repositorio (corrige GAP-01, GAP-17 y las vulnerabilidades CWE-798 asociadas a S-01, S-02 y S-03).
- **Aislamiento de la colección *audit_logs*:** Se añade una colección en MongoDB separada de los datos clínicos, configurada con política *insert-only* a nivel de aplicación. El componente *AuditLogAspect* (Spring AOP) intercepta de forma síncrona cada operación sobre recursos PHI y graba el log antes de retornar la respuesta al cliente (elimina GAP-21 y GAP-22).
- **Cifrado en la frontera TB-1:** La comunicación entre el cliente React y el *API Gateway* se restringe exclusivamente a TLS 1.3. La frontera TB-2 (*API Gateway* hacia MongoDB Atlas) mantiene el TLS nativo de Atlas, el cual ya cumplía con los requisitos (GAP-12).

6.4 Especificación técnica de controles por dominio ASVS

Para cada dominio se especifica la brecha *as-is*, el control *to-be* propuesto y las brechas cerradas.

6.4.1 V6 – Autenticación (GAP-01, GAP-02, GAP-03, GAP-4)

Brecha *as-is*: La clave secreta de JWT (HS256) se encuentra codificada de forma rígida en *JwtService.java* (S-01, CWE-798, CVSS 8.8). La autenticación carece de segundo factor para acceso a módulos PHI (GAP-02). El *endpoint* *POST /auth/login* no implementa control de tasa, exponiéndose a ataques de fuerza bruta amplificados por CVE-2025-22228 (GAP-03). El algoritmo AES-ECB utilizado en el módulo de autenticación no provee confidencialidad semántica (GAP-04).

Control *to-be*:

- **Migración a JWT RS256 asimétrico:** reemplazar el secreto simétrico por un par de claves RSA-2048. La clave privada reside exclusivamente en el gestor de secretos externo (Vault/KMS) y se inyecta en tiempo de ejecución mediante variable de entorno cifrada. La clave pública se distribuye a los servicios verificadores. La rotación de claves se ejecuta con frecuencia configurable sin despliegue de código.
- **MFA para módulos PHI:** implementar un segundo factor de autenticación (TOTP - RFC 6238 o FIDO2) obligatorio para los roles que acceden a F1 (fichas), F2 (diagnósticos) y F3 (comités). El acceso a módulos no-PHI no requiere segundo factor.
- **Rate limiting en API gateway:** configurar un límite de intentos sobre *POST /auth/login* (máx. 5 intentos por IP en ventana de 60 segundos, bloqueo progresivo). El *API Gateway* absorbe la presión antes de que alcance al *backend*.
- **Eliminación de AES-ECB en flujos de autenticación:** reemplazado por AES-256-GCM en el control V11 (ver más adelante), que cubre la misma clase de debilidad criptográfica.



6.4.2 V7 – Gestión de sesión (GAP-05)

Brecha as-is: El *token* JWT se almacena en el *localStorage* del navegador (TM-05, CVSS 5.9 est.), exponiéndolo a extracción mediante inyección de scripts (XSS). Cualquier script arbitrario ejecutado en el contexto del navegador puede leer y exfiltrar el identificador de sesión.

Control to-be:

- **Cookie *HttpOnly+Secure+SameSite=Strict*:** el *token* JWT se almacena en una cookie con los atributos *HttpOnly* (inaccesible desde JavaScript), *Secure* (transmisión solo sobre HTTPS/TLS) y *SameSite=Strict* (mitigación de CSRF). El tiempo de vida del *token* de acceso se reduce a 15 minutos. Un *refresh token* de larga duración, almacenado con los mismos atributos, permite renovar el acceso sin reautenticación del usuario.
- **Rotación de *refresh tokens*:** cada uso del *refresh token* emite un nuevo token de refresco e invalida el anterior (detección de reuso ante robo de cookie).

6.4.3 V8 – Autorización (GAP-06, GAP-07, GAP-08)

Brecha as-is: El *backend* Spring Boot no implementa control de acceso basado en roles a nivel de *endpoint* (TM-07, TM-15; CVE-2024-38821, CVSS 9.1). El control de acceso se delega exclusivamente a las rutas de la SPA React, que es un mecanismo del lado del cliente eludible trivialmente. Adicionalmente, no existe validación de propiedad del recurso solicitado (GAP-07), permitiendo que un usuario clínico legítimo acceda o modifique PHI de pacientes no asignados.

Control to-be:

- **RBAC en Spring Security:** anotación *@PreAuthorize* en cada controlador de dominio PHI, evaluada contra el rol extraído del JWT verificado. Los roles se mapean a los actores definidos en la Tabla 4 del Capítulo 4 (TA-01 Administrador, TA-02 Médico Oncólogo, TA-03 Administrativo). Ningún *endpoint* PHI es accesible sin autenticación válida y rol autorizado.
- **Validación de *ownership*:** para operaciones de lectura y modificación de registros individuales (fichas, diagnósticos, solicitudes de comité), el *backend* verifica que el *patientId* del recurso solicitado esté en el conjunto de pacientes asignados al actor autenticado. La verificación se ejecuta antes de consultar MongoDB, no después.
- **Actualización de Spring Security:** migración a versión remediada de spring-security-web que cierra CVE-2024-38821 y CVE-2026-22732.

6.4.4 V11 – Autorización (GAP-09, GAP-10)

Brecha as-is: *AESCipher.java* implementa AES en modo de operación ECB (S-04, S-05, CWE-327, CVSS 7.4 est.). Al ser un esquema determinístico sin vector de inicialización, bloques de texto plano idénticos producen bloques cifrados idénticos. Un atacante pasivo con acceso a la base de datos puede realizar análisis de frecuencia estadística sobre diagnósticos CIE-10 repetidos e inferir el historial clínico de los pacientes. Adicionalmente, no se implementa cifrado autenticado (GAP-10), lo que impide detectar modificaciones no autorizadas del contenido cifrado.

Control to-be:

- **Migración a AES-256-GCM:** reemplazar el modo ECB por AES-256-GCM (Galois/Counter Mode). GCM genera un vector de inicialización (IV) aleatorio de 96 bits por cada operación de cifrado, garantizando que el mismo texto plano produzca criptogramas distintos. GCM incluye además un tag de autenti-

cación (MAC) de 128 bits que permite detectar cualquier modificación del *ciphertext* (cifrado autenticado - AEAD).

- **Client-Side Field Level Encryption (CSFLE):** para los campos PHI de mayor sensibilidad (diagnósticos, estadios tumorales, decisiones de comité), se habilita el cifrado a nivel de campo nativo de MongoDB Atlas (CSFLE). El cifrado y descifrado ocurre en el *driver* Java antes de que los datos salgan del proceso de la aplicación, garantizando que MongoDB Atlas nunca procese PHI en texto plano.

6.4.5 V12 – Comunicación segura (GAP-11, GAP-13, GAP-14)

Brecha as-is: La frontera de confianza TB-1 (cliente-API) opera sobre HTTP sin cifrado (TM-02, TM-04, TM-11, TM-19; CVSS 7.5 est.), transmitiendo PHI, credenciales e imágenes diagnósticas en texto plano. Adicionalmente, el servidor Tomcat embebido presenta dos vulnerabilidades críticas de RCE (CVE-2025-24813, CVE-2024-50379; CVSS 9.8) que permiten ejecución remota de código y denegación de servicio sin autenticación. El *endpoint* de carga de archivos no implementa cuota de tamaño (GAP-14), amplificando la superficie de ataque DoS.

Control to-be:

- **TLS 1.3 obligatorio en TB-1:** el API *Gateway* termina TLS 1.3 para toda comunicación cliente-API. Las conexiones HTTP en texto plano son redirigidas con código 301 hacia HTTPS. El certificado TLS se gestiona mediante una autoridad certificadora reconocida.
- **Actualización de Apache Tomcat:** migración a versión $\geq 10.1.34$ (rama 10.x) o $\geq 9.0.99$ (rama 9.x), que incorporan los parches de CVE-2025-24813 y CVE-2024-50379. Esta actualización es una corrección de dependencia *upstream* y no implica cambios en el código de la aplicación.
- **Cuota de tamaño en FileController:** configurar `spring.servlet.multipart.max-file-size` y `spring.servlet.multipart.max-request-size` en `application.properties` para limitar la carga de archivos médicos (límite recomendado: 50 MB por archivo, 100 MB por solicitud). El API *Gateway* aplica adicionalmente un límite de tasa sobre `POST /files/upload`.

6.4.6 V14 – Protección de datos (GAP-15, GAP-16, GAP-17)

Brecha as-is: El cifrado AES-ECB implementado no provee confidencialidad semántica (GAP-15, Parcial). No existe mecanismo de retención ni eliminación automatizada de registros PHI (GAP-16), incumpliendo el derecho al olvido oncológico establecido en el Art. 8 bis de la Ley N.º 21.258. La cadena de conexión de MongoDB Atlas se almacena como credencial estática en el repositorio público (TM-21, TM-22; GAP-17, CVSS 8.8 est.).

Control to-be:

- **Migración completa a AES-256-GCM (coordinar con control V11):** resuelve GAP-15 completamente, elevando el estado de Parcial a Cumple.
- **Política de retención y eliminación automatizada:** implementar un *job* programado (`@Scheduled` de Spring) que, con periodicidad diaria, identifique registros PHI que hayan superado el plazo de retención definido por la Ley N.º 21.258 Art. 8 bis (cinco años desde alta definitiva). La eliminación adopta el patrón *soft delete* (campo `deletedAt` + flag `anonymized`) para mantener trazabilidad estadística sin retener datos personales identificables.
- **Externalización de credenciales:** la cadena de conexión de MongoDB Atlas, junto con todas las credenciales de infraestructura, se migran al gestor de secretos externo (Vault/KMS) definido en el control V6. El repositorio no contiene ningún secreto en texto plano.

6.4.7 V16 – Auditoría y trazabilidad (GAP-21, GAP-22)



Brecha as-is: La API no registra los accesos de lectura ni escritura sobre recursos PHI (TM-03, TM-10; GAP-21, CVSS 5.5 est.). Los registros de sistema existentes carecen de inmutabilidad y control de acceso (GAP-22, CVSS 6.5 est.), produciendo un escenario de no-repudio negativo que impide auditorías forenses externas requeridas por HIPAA §164.312(b).

Control to-be:

- **Spring AOP Audit Aspect:** implementar un aspecto transversal (*@Aspect*) que intercepta toda operación sobre controladores PHI (F1, F2, F3). Por cada operación registra: actor (sub del JWT verificado), *timestamp* UTC, tipo de recurso, identificador del recurso, acción (READ / WRITE / DELETE) y resultado (SUCCESS / DENIED). El aspecto actúa como *advice @Around*: persiste el registro en *audit_logs* antes de devolver la respuesta, garantizando que ninguna operación PHI quede sin traza incluso ante excepción.
- **Colección *audit_logs insert-only*:** el repositorio de auditoría expone únicamente el método *save()*. Los métodos *delete()*, *update()* y *deleteAll()* no están implementados en la capa de aplicación. El acceso de lectura a *audit_logs* se restringe al rol Administrador (TA-01). Esta política de acceso garantiza la inmutabilidad a nivel de aplicación requerida por HIPAA §164.312(b) y ASVS V16.4.2.

6.4.8 V2 – Validación de entradas (GAP-18, GAP-19, GAP-20)

Brecha as-is: Los ocho controladores de dominio PHI (S-06 a S-13, CWE-915) exponen directamente las entidades JPA como parámetros de entrada, permitiendo asignación masiva de atributos (*mass assignment*). El componente Spring WebMVC presenta vulnerabilidades de *path traversal* (CVE-2024-38816, CVE-2024-38819; CVSS 7.5). El servicio de almacenamiento GridFS no valida el tipo real del archivo cargado, permitiendo la inyección de *payloads* maliciosos mediante tipos MIME falsificados.

Control to-be:

- **Data Transfer Objects (DTOs) explícitos:** reemplazar las entidades JPA como parámetro de los métodos *@RequestBody* por DTOs que declaran exclusivamente los campos que el cliente está autorizado a modificar. Esta medida elimina estructuralmente el vector CWE-915 sin requerir listas de exclusión ni anotaciones *@JsonIgnore* frágiles.
- **Actualización de Spring WebMVC:** migración a versión ≥6.1.14 que incorpora los parches de CVE-2024-38816 y CVE-2024-38819. Actualización de dependencia *upstream* sin cambios en código de aplicación.
- **Validación de *magic numbers* en *FileController*:** antes de persistir un archivo en GridFS, el *FileController* lee los primeros *bytes* del *stream* (*magic number*) y verifica que correspondan al tipo MIME declarado en el encabezado de la solicitud. Los archivos cuyo *magic number* no coincida con un tipo permitido (DICOM, JPEG, PNG, PDF) son rechazados con código 415 antes de acceder al sistema de almacenamiento.

6.5 Matriz de trazabilidad: GAP-ID – normativa – control to-be

La Tabla 22 conecta cada una de las 22 brechas identificadas en el Capítulo 5 con el control *to-be* propuesto, el requisito normativo satisfecho y el estado de mitigación resultante. Esta matriz constituye el instrumento de validación teórica del quinto objetivo específico del trabajo de memoria y demuestra que el diseño propuesto satisface íntegramente los siete requisitos normativos de la Tabla 1.

Tabla 22. Matriz de trazabilidad: brechas de seguridad → controles *to-be* → cumplimiento normativo

Fuente: Elaboración propia

GAP-ID	Control ASVS v5.0.0	REQ bloqueado	Base legal	Control <i>to-be</i> propuesto
--------	---------------------	---------------	------------	--------------------------------

GAP-01	V13.3.1 - Secretos no en código fuente	REQ-006a	HIPAA §164.312(d)	JWT RS256 asimétrico; clave privada en Vault/KMS
GAP-02	V6.3.3 - MFA en módulos PHI	REQ-006a	HIPAA §164.312(d)	TOTP / FIDO2 obligatorio para roles con acceso PHI
GAP-03	V6.1.1 - <i>Rate limiting</i> en autenticación	REQ-006a	HIPAA §164.312(d)	<i>Rate limiting</i> en API Gateway sobre POST /auth/login
GAP-04	V6.2.5 - Algoritmos criptográficos aprobados	REQ-002, REQ-005	Ley N.º 19.628 Art. 11; HIPAA §164.312(a)(2)(iv)	Migración a AES-256-GCM (coordina con GAP-09)
GAP-05	V7.3.2 - JWT fuera de <i>localStorage</i>	REQ-006a	HIPAA §164.312(d)	<i>Cookie HttpOnly+Secure+SameSite=Strict; refresh token</i> rotativo
GAP-06	V8.2.1 - RBAC en servidor	REQ-001	Ley N.º 20.584 Arts. 12-13; HIPAA §164.312(a)(1)	@PreAuthorize por endpoint en Spring Security
GAP-07	V8.2.2 - Validación de propiedad del recurso	REQ-001	Ley N.º 20.584 Arts. 12-13; HIPAA §164.312(a)(1)	Verificación de <i>ownership</i> antes de consulta MongoDB
GAP-08	V8.2.4 - Controles adaptativos de autorización	REQ-001	Ley N.º 20.584 Arts. 12-13; HIPAA §164.312(a)(1)	Actualización Spring Security (parche CVE-2024-38821, CVE-2026-22732)
GAP-09	V11.2.1 - Implementaciones criptográficas aprobadas	REQ-002	Ley N.º 19.628 Art. 11; HIPAA §164.312(a)(2)(iv)	AES-256-GCM con IV aleatorio por operación
GAP-10	V11.3.3 - Cifrado autenticado (AES-GCM o MAC)	REQ-005	HIPAA §164.312(c)	AES-256-GCM (tag MAC de 128 bits incluido) + CSFLE MongoDB
GAP-11	V12.2.1 - TLS en TB-1 (cliente-API)	REQ-002	Ley N.º 19.628 Art. 11; HIPAA §164.312(e)(1)	TLS 1.3 obligatorio en API Gateway; redirección 301 HTTP a HTTPS
GAP-12	V12.2.1 - TLS en TB-2 (API-MongoDB)	—	—	Control existente mantenido (TLS nativo Atlas)
GAP-13	V13.1.2 - Prevención DoS/RCE en servidor	REQ-007	Ley N.º 21.258 Arts. 1 y 7	Actualización Tomcat ≥10.1.34 / 9.0.99; WAF perimetral
GAP-14	V15.2.2 - <i>Rate limiting</i> en carga de archivos	REQ-007	Ley N.º 21.258 Arts. 1 y 7	Cuota <i>multipart</i> + <i>rate limiting</i> en API Gateway sobre /files/upload
GAP-15	V14.1.2 - Cifrado formal de datos sensibles	REQ-002	Ley N.º 19.628 Art. 11; HIPAA §164.312(a)(2)(iv)	Migración a AES-256-GCM (coordina con GAP-09); CSFLE MongoDB
GAP-16	V14.2.7 - Retención y eliminación de PHI	REQ-004	Ley N.º 21.258 Art. 8 bis	<i>Job @Scheduled</i> de retención + <i>soft delete</i> con anonimización
GAP-17	V13.3.1 - Credenciales de BD no en repositorio	REQ-002	Ley N.º 19.628 Art. 11; HIPAA §164.312(a)(2)(iv)	Cadena de conexión externalizada a Vault/KMS
GAP-18	V5.1.2 - Prevención <i>mass assignment</i>	REQ-005	HIPAA §164.312(c)	DTOs explícitos como <i>request body</i> en controladores PHI
GAP-19	V5.2.1 - Prevención <i>path traversal</i>	REQ-002	Ley N.º 19.628 Art. 11; HIPAA §164.312(e)(1)	Actualización Spring WebMVC ≥6.1.14 (parche CVE-2024-38816/38819)



GAP-20	V5.1.2 - Validación de tipo MIME en archivos	REQ-005	HIPAA §164.312(c)	Validación de <i>magic numbers</i> en <i>FileController</i> antes de GridFS
GAP-21	V16.3.2 - <i>Audit trail</i> sobre accesos PHI	REQ-003	HIPAA §164.312(b)	Spring AOP <i>@Aspect insert-only</i> en colección <i>audit_logs</i>
GAP-22	V16.4.2 - Protección de registros de auditoría	REQ-003	HIPAA §164.312(b)	Repositorio <i>audit_logs</i> sin métodos <i>delete/update</i> ; acceso restringido a TA-01

6.6 Límites del diseño propuesto

El diseño propuesto da cumplimiento formal al quinto objetivo específico del trabajo de memoria: especificar controles técnicos que mitiguen las vulnerabilidades detectadas y demostrar su efectividad mediante trazabilidad entre riesgos, normativa y soluciones arquitectónicas.

6.6.1 Limitaciones declaradas

El presente diseño está sujeto a las siguientes limitaciones que deben considerarse en la interpretación de los resultados:

Validación teórica: los controles propuestos no han sido implementados en el repositorio auditado (*commit a882192*). La validación se basa en trazabilidad entre brechas y controles, conforme a la metodología declarada en §1.4. La efectividad real de los controles depende de su implementación correcta, que constituye trabajo futuro.

Dependencias *upstream*: tres brechas (GAP-08, GAP-13, GAP-19) requieren actualizaciones de versiones parcheadas de componentes de terceros (Spring Security, Apache Tomcat, Spring WebMVC). La disponibilidad de estos parches está fuera del control del equipo de desarrollo de SIGO.

Commit de referencia fijo. el análisis se realizó sobre el *commit a882192* de la rama principal. Cambios posteriores en el repositorio pueden alterar la superficie de ataque analizada.

7 Conclusiones

El objetivo general del trabajo (evaluar la seguridad de SIGO mediante una auditoría de código y rediseñar su arquitectura bajo defensa en profundidad, demostrando formalmente el potencial de cumplimiento de un subconjunto trazable de controles ASVS) se cumplió en su dimensión de diseño, no en su dimensión de implementación. La evidencia que sostiene esta afirmación es la Tabla 22: una matriz de trazabilidad que conecta cada una de las 22 brechas críticas identificadas con un control *to-be* y el requisito normativo que satisface, cerrando formalmente el ciclo diagnóstico-normativa-control que definían los cinco objetivos específicos. Esta validación es teórica (basada en trazabilidad documental) y no empírica, distinción que es necesario mantener explícita en esta instancia y no solo en el capítulo de diseño: el trabajo demuestra que existe una solución arquitectónica coherente y justificada normativamente para cada brecha, no que esa solución ya opera de forma verificada sobre el sistema productivo.

La principal contribución del trabajo no es SIGO en sí, sino el método: la construcción de un subconjunto trazable de ASVS derivado directamente de normativa aplicable (Ley N.º 20.584, Ley N.º 19.628, Ley N.º 21.258 y HIPAA §164.312) es un procedimiento reproducible para cualquier sistema de información clínico chileno que deba justificar sus controles técnicos ante un marco regulatorio, más allá del caso oncológico particular de este informe. En la misma línea, la combinación de SAST, SCA y modelado de amenazas STRIDE mostró ser complementaria y no redundante: Semgrep no reportó hallazgos con severidad $\geq$ *WARNING* en el *frontend* pese a cubrirlo con *rulesets* específicos, mientras que STRIDE identificó defectos lógicos de diseño

(como la ausencia de auditoría sobre operaciones PHI) que ningún escáner detecta por construcción. Esa combinación es aplicable como patrón de auditoría a otros sistemas de salud digital con restricciones de tiempo y presupuesto similares a las que originaron la deuda técnica de SIGO.

El alcance de los resultados está acotado por limitaciones que condicionan su interpretación. La principal es la ya señalada: los controles *to-be* no fueron implementados ni re-testeados contra los hallazgos SAST/SCA originales, por lo que su efectividad real queda pendiente de verificación empírica. El diagnóstico, además, está anclado a un commit fijo (a882192) del repositorio, de modo que cambios posteriores en SIGO pueden desalinear el estado real del sistema respecto de lo aquí documentado. Tres brechas (GAP-08, GAP-13, GAP-19) dependen de parches *upstream* fuera del control del equipo de desarrollo, y la cobertura de Semgrep en su edición OSS es menor que la de sus versiones comerciales, por lo que la ausencia de hallazgos adicionales no puede interpretarse como ausencia de riesgo. Finalmente, SIGO fue desarrollado y posteriormente auditado por la misma persona, y el sistema no se encuentra actualmente implementado en un entorno hospitalario real: tanto el *gap analysis* como la arquitectura *to-be* fueron construidos y validados sin revisión independiente por parte de un tercero ajeno al desarrollo ni retroalimentación de personal clínico operando el sistema en producción. Esta doble condición (desarrollo y auditoría a cargo de la misma persona, y ausencia de un contexto de uso real) constituye una limitación de validez de contenido que un trabajo futuro debería subsanar mediante revisión por pares independiente y, eventualmente, mediante el despliegue piloto de SIGO en un entorno clínico controlado.

Estas limitaciones definen directamente las líneas de trabajo futuro más inmediatas: implementar y re-testear los controles *to-be* sobre el repositorio real para cerrar empíricamente el ciclo hoy solo teórico; resolver las tres dependencias *upstream* pendientes verificando que el parche efectivamente elimina el CVE asociado. A una escala mayor, este trabajo se ubica casi enteramente dentro de las funciones de *Verification* y, parcialmente, *Design* del modelo de madurez OWASP SAMM [7], sin abordar *Governance* ni *Operations*; en consecuencia, se recomienda adoptar SAMM como hoja de ruta de mejora continua para SIGO (usando esta memoria como línea base medible) si el sistema avanza hacia un despliegue operativo real, en lugar de repetir auditorías puntuales como esta. De modo análogo, la matriz de riesgo ordinal empleada (Tabla 19, CVSS × impacto clínico PHI) es una forma simplificada de evaluación de riesgo comparable en espíritu a ISO/IEC 27005 [27] y NIST SP 800-30 [28], pero sin adoptar sus procesos formales completos de identificación de activos y tratamiento del riesgo; su evolución natural es migrar hacia un modelo cuantitativo como FAIR [29] una vez que existan datos operacionales históricos de frecuencia de incidentes, de los que este diagnóstico de caja blanca puntual careció.

Desde una perspectiva personal, el mayor aprendizaje de este trabajo fue metodológico más que técnico: traducir una exigencia legal (como la confidencialidad exigida por la Ley N.º 20.584) en un control ASVS verificable exige resolver ambigüedad normativa con criterio propio, y ese ejercicio de *scoping* resultó más exigente que la ejecución de las herramientas de análisis en sí. Igualmente, formativo fue constatar en la práctica los límites de la automatización en seguridad: un análisis SAST/SCA sin hallazgos no es sinónimo de un sistema seguro, y solo la combinación con modelado de amenazas manual permitió exponer defectos de diseño que las herramientas, por su naturaleza, no están concebidas para encontrar.



Referencias

- [1] International Agency for Research on Cancer, Chile - Globocan 2022: Cancer Observatory. World Health Organization, 2022. [Online]. Available: <https://gco.iarc.fr/>
- [2] Superintendencia de Salud, Ley N.º 20.584: Regula los derechos y deberes que tienen las personas en relación con acciones vinculadas a su atención en salud, 2012. [Online]. Available: <https://www.superdesalud.gob.cl/tax-materias-prestadores/ley-dederechos-y-deberes-4185/>
- [3] U.S. Department of Health & Human Services, HIPAA Security Rule: 45 CFR Part 164 Subpart C, 2003. [Online]. Available: <https://www.hhs.gov/hipaa/for-professionals/security/index.html>
- [4] OWASP Foundation, OWASP Top 10 – 2024: The Ten Most Critical Web Application Security Risks, 2024. [Online]. Available: <https://owasp.org/www-project-top-ten/>
- [5] OWASP Foundation, OWASP Application Security Verification Standard (ASVS) 5.0.0, 2025. [Online]. Available: <https://owasp.org/www-project-application-security-verification-standard/>
- [6] S. T. Argaw *et al.*, "Cybersecurity of Hospitals: discussing the challenges and working towards mitigating the risks," *BMC Medical Informatics and Decision Making*, vol. 20, no. 1, p. 146, 2020.
- [7] OWASP Foundation, "OWASP Software Assurance Maturity Model (SAMM) v2.0," 2020. [Online]. Available: <https://owasp.org/samm/>
- [8] Kruchten, R. L. Nord and I. Ozkaya, "Technical Debt: From Metaphor to Theory and Practice," in *IEEE Software*, vol. 29, no. 6, pp. 18-21, Nov.-Dec. 2012, doi: 10.1109/MS.2012.167
- [9] M. Aydos, Ç. Aldan, E. Coşkun, y A. Soydan, "Security testing of web applications: A systematic mapping of the literature," *Journal of King Saud University - Computer and Information Sciences*, vol. 34, pp. 6775-6792, 2022, doi: 10.1016/j.jksuci.2021.09.018.
- [10] Manadhata, Pratyusa & Wing, Jeannette. (2011). An Attack Surface Metric. Software Engineering, IEEE Transactions on. 37. 371-386. 10.1109/TSE.2010.60.
- [11] Joint Task Force, "Security and Privacy Controls for Information Systems and Organizations", National Institute of Standards and Technology, Gaithersburg, MD, NIST Special Publication (SP) 800-53, Rev. 5, Sep. 2020. [Online]. Available: <https://doi.org/10.6028/NIST.SP.800-53r5> [Accessed: Mar. 16, 2026].
- [12] Biblioteca del Congreso Nacional de Chile, "Ley 19628: Sobre protección de la vida privada", 1999. [Online]. Available: <https://www.bcn.cl/leychile/navegar?idNorma=141599> [Accessed: Mar. 18, 2026].
- [13] Biblioteca del Congreso Nacional de Chile, "Ley 21258: Crea la Ley Nacional del Cáncer, que rinde homenaje póstumo al doctor Claudio Mora", 02-sep-2020. [Online]. Available: <https://www.bcn.cl/leychile/navegar?idNorma=1149004> [Accessed: Mar. 18, 2026].
- [14] Biblioteca del Congreso Nacional de Chile, "Ley 21668: Modifica la ley N.º 20.584 con el objeto de establecer la interoperabilidad de las fichas clínicas", 23-may-2024. [Online]. Available: <https://www.bcn.cl/leychile/navegar?idNorma=1203827> [Accessed: Mar. 19, 2026].
- [15] Superintendencia de Seguridad Social (SUSESO), "Ley 20.584, artículo 13", SUSESO, 2012. [Online]. Available: <https://www.suseso.cl/612/w3-propertyvalue-130630.html> [Accessed: Mar. 19, 2026].
- [16] U.S. Dept. of Health and Human Services, "HHS implementation guidance for technical safeguards," Office for Civil Rights, Washington, DC, USA, Tech. Rep., [Online]. Available: <https://www.hhs.gov/sites/default/files/ocr/privacy/hipaa/administrative/securityrule/techsafeguards.pdf>
- [17] National Institute of Standards and Technology (NIST), "Secure Software Development Framework (SSDF) Version 1.1: Mitigating the Risk of Software Vulnerabilities," NIST Special Publication 800-218, Feb. 2022.
- [18] OWASP Foundation, "OWASP Secure Coding Practices - Quick Reference Guide" OWASP, 2023. [Online]
- [19] Cybersecurity and Infrastructure Security Agency (CISA), "2025 Minimum Elements for a Software Bill of Materials (SBOM)," Dept. of Commerce, Washington, DC, 2025. Available: <https://www.cisa.gov/resources-tools/resources/2025-minimum-elements-software-bill-materials-sbom>
- [20] OWASP, "Threat Modeling Process," OWASP Foundation, [En línea]. Disponible en: https://owasp.org/www-community/Threat_Modeling_Process#stride. [Accessed: Apr. 9, 2026].
- [21] The MITRE Corporation, "CVE Top 25 Most Dangerous Software Weaknesses", MITRE. [Online]. Available: <https://cwe.mitre.org/top25/> [Accessed: Mar. 16, 2026].
- [22] Forum of Incident Response and Security Teams (FIRST), "Common Vulnerability Scoring System v3.1: Specification Document," FIRST.org. [Online]. Available: <https://www.first.org/cvss/v3.1/specification-document>
- [23] European Union Agency for Cybersecurity (ENISA), "Health Threat Landscape," *ENISA*, Jul. 2023. [Online]. Available: <https://www.enisa.europa.eu/publications/health-threat-landscape>. [Accessed: Apr. 9, 2026].
- [24] S. Rose, O. Borchert, S. Mitchell, and S. Connelly, "Zero Trust Architecture," National Institute of Standards and Technology (NIST), Gaithersburg, MD, NIST Special Publication (SP) 800-207, Aug. 2020. [Online]. Available: <https://nvl-pubs.nist.gov/nistpubs/specialpublications/NIST.SP.800-207.pdf>. [Accessed: Apr. 9, 2026].



- [25] National Institute of Standards and Technology (NIST), "Cryptographic Standards and Guidelines," *NIST Computer Security Resource Center*. [Online]. Available: <https://csrc.nist.gov/projects/cryptographic-standards-and-guidelines>. [Accessed: Apr. 9, 2026].
- [26] National Institute of Standards and Technology (NIST), "National Vulnerability Database (NVD)," U.S. Department of Commerce. [Online]. Available: <https://nvd.nist.gov/>
- [27] International Organization for Standardization, "ISO/IEC 27005:2022 — Information security, cybersecurity and privacy protection — Guidance on managing information security risks," ISO, Oct. 2022.
- [28] R. Ross, P. Viscuso, G. Guissanie, K. Dempsey, and M. Riddle, "Guide for Conducting Risk Assessments," National Institute of Standards and Technology, Gaithersburg, MD, NIST Special Publication (SP) 800-30, Rev. 1, Sep. 2012. [Online]. Available: <https://doi.org/10.6028/NIST.SP.800-30r1>
- [29] The Open Group / FAIR Institute, "Open FAIR Body of Knowledge — Factor Analysis of Information Risk," 2025. [Online]. Available: <https://www.fairinstitute.org/what-is-fair>



Glosario

1. Marco normativo chileno

Ley N.º 20.584 — Derechos y deberes en acciones de salud

- **Artículo 12 (reserva de la ficha clínica):** Establece que la ficha clínica es estrictamente confidencial. El acceso se limita al paciente, los profesionales directamente a cargo de su atención y terceros con autorización expresa.
- **Artículo 13 (custodia de la información):** Obliga a los prestadores de salud a custodiar las fichas clínicas, exigiendo medidas de seguridad técnicas para su acceso, conservación y uso.

Ley N.º 19.628 — Protección de la vida privada

- **Artículo 11 (tratamiento de datos sensibles):** Regula el uso de datos de salud. Exige que su tratamiento cumpla con los principios de finalidad (uso exclusivo para la atención médica), proporcionalidad (acceso solo a los datos estrictamente necesarios) y seguridad.

Ley N.º 21.258 — Ley Nacional del Cáncer

- **Artículo 1:** Establece el deber de garantizar la continuidad, oportunidad y calidad de la atención oncológica en la red asistencial.
- **Artículo 2, letra b:** Obliga a mantener el secreto y la confidencialidad absoluta de los datos personales de los pacientes.
- **Artículo 7:** Garantiza la continuidad del proceso de atención médica a lo largo del tiempo.
- **Artículo 8:** Refuerza la obligatoriedad de cumplir las leyes de privacidad y derechos del paciente en el Registro Nacional de Cáncer.
- **Artículo 8 bis (derecho al olvido oncológico):** Permite al paciente exigir que su información oncológica no sea consultada en procesos administrativos (selección, seguros) tras cinco años desde su alta definitiva.

2. Estándar Internacional: HIPAA §164.312

- **Control de acceso [§164.312(a)(1)]:** Obliga a restringir el acceso a la PHI exclusivamente a las personas o programas de software que cuenten con autorización funcional.
- **Identificador único [§164.312(a)(2)(i)]:** Exige asignar un identificador exclusivo a cada usuario para permitir la trazabilidad individual de sus acciones en el sistema.
- **Cifrado [§164.312(a)(2)(iv)]:** Requiere la implementación de mecanismos criptográficos para proteger la información almacenada (datos en reposo).
- **Controles de auditoría [§164.312(b)]:** Obliga a registrar, almacenar y examinar de forma inmutable la actividad de cualquier usuario dentro del sistema.
- **Integridad [§164.312(c)]:** Exige proteger la información médica contra cualquier alteración o destrucción no autorizada.
- **Autenticación [§164.312(d)]:** Requiere procedimientos técnicos robustos para verificar y confirmar la identidad real de quien solicita acceso a los datos.
- **Seguridad en transmisión [§164.312(e)(1)]:** Obliga a proteger la PHI mediante cifrado para evitar accesos no autorizados mientras los datos viajan a través de la red.

3. Clasificación de Debilidades (CWE)

- **CWE-798:** Uso de credenciales estáticas en código fuente
- **CWE-327:** Uso de criptografía débil o riesgosa
- **CWE-915:** Modificación incontrolada de atributos (*Mass Assignment*)
- **CWE-22:** *Path Traversal* (Salto de directorio)
- **CWE-400:** Consumo incontrolado de recursos

4. Vulnerabilidades de Componentes (CVE)

- **CVE-2025-24813 / CVE-2024-50379 (Apache Tomcat) | Severidad: Crítica (9.8)**



- **Descripción:** Fallas lógicas (condición de carrera y procesamiento PUT) que habilitan la escritura arbitraria y la ejecución remota de código (RCE) en el servidor.
- **CVE-2024-38821 / CVE-2026-22732** (Spring Security) | **Severidad: Crítica (9.1)**
 - **Descripción:** Vulnerabilidades de evasión (*bypass*) que permiten a un atacante saltarse la capa de autorización *web* y acceder a recursos sin estar autenticado.
- **CVE-2024-38816 / CVE-2024-38819** (Spring WebMVC) | **Severidad: Alta (7.5)**
 - **Descripción:** Defectos de *Path* Traversal al procesar rutas URL o recursos estáticos, permitiendo acceso no autorizado a archivos del servidor.
- **CVE-2025-22228** (Spring Security Crypto) | **Severidad: Alta (7.4)**
 - **Descripción:** Falla en el algoritmo BCrypt que ignora su límite de longitud, truncando contraseñas largas de forma silenciosa y debilitando la autenticación.

**Anexos A Evidencia del diagnostico de caja blanca****Tabla A-1. Vulnerabilidades SonarQube ordenadas por severidad**

Fuente: Elaboración propia

ID	Severidad	Regla	Archivo: Línea	Descripción	CWE	ASVS
S-01	BLOCKER	java:S6418	JwtService.java:21	JWT <i>secret</i> estático en código fuente	CWE-798	V13.3.1
S-02	BLOCKER	secrets:S7013	BitacoraModal.jsx:106	OpenAI API key estática en <i>frontend</i>	CWE-798	V13.3.1
S-03	BLOCKER	javascript:S6418	BitacoraModal.jsx:106	apiKey estática detectada (misma instancia que S-02)	CWE-798	V13.3.1
S-04	CRITICAL	java:S5542	AESCipher.java:18	Cifrado AES en modo ECB — sin confidencialidad semántica	CWE-327	V6.2.5
S-05	CRITICAL	java:S5542	AESCipher.java:25	Cifrado AES en modo ECB (segunda instancia)	CWE-327	V6.2.5
S-06	CRITICAL	java:S4684	PacienteController.java:48	Entidad JPA como <i>request body</i> — <i>mass assignment</i>	CWE-915	V5.1.2
S-07	CRITICAL	java:S4684	SolicitudController.java:47	Entidad JPA como <i>request body</i> — <i>mass assignment</i>	CWE-915	V5.1.2
S-08	CRITICAL	java:S4684	TicketController.java:46	Entidad JPA como <i>request body</i> — <i>mass assignment</i>	CWE-915	V5.1.2
S-09	CRITICAL	java:S4684	TasaMortController.java:37	Entidad JPA como <i>request body</i> — <i>mass assignment</i>	CWE-915	V5.1.2
S-10	CRITICAL	java:S4684	BitacoraController.java:40	Entidad JPA como <i>request body</i> — <i>mass assignment</i>	CWE-915	V5.1.2
S-11	CRITICAL	java:S4684	CalendarController.java:41	Entidad JPA como <i>request body</i> — <i>mass assignment</i>	CWE-915	V5.1.2
S-12	CRITICAL	java:S4684	GlobalConfigController.java:31	Entidad JPA como <i>request body</i> — <i>mass assignment</i>	CWE-915	V5.1.2
S-13	CRITICAL	java:S4684	SintomaController.java:35	Entidad JPA como <i>request body</i> — <i>mass assignment</i>	CWE-915	V5.1.2

Tabla A-2: Inventario completo de amenazas STRIDE sobre la arquitectura *as-is* de SIGO

Fuente: Elaboración propia

ID	Categoría STRIDE	Actor(es)	Flujo / Frontera	Vector de amenaza	Referencia técnica	Control ASVS vulnerado
TM-01	Suplantación (S)	TA-04	F1	Secreto JWT estático en código fuente: forja de <i>tokens</i> con identidad arbitraria sin requerir credenciales válidas	S-01 (CWE-798)	V13.3.1 (L2)
TM-02	Manipulación (T)	TA-01	F1 / TB-1	Credenciales transmitidas en texto plano sobre HTTP: modificación o inyección de contenido por MITM en el tramo navegador → API	TB-1 (§4.1.3)	V12.2.1 (L1)
TM-03	Repudio (R)	TA-02, TA-03	F1	Ausencia de registro de eventos de autenticación (<i>login</i> , <i>logout</i> , <i>expiración</i>)	REQ-003 / HIPAA §164.312(b)	V16.3.2 (L3)

				ración de <i>token</i>): ninguna acción de sesión es atribuible de forma forense		
TM-04	Divulgación (I)	TA-01	F1 / TB-1	<i>Token</i> JWT interceptable en tránsito sobre HTTP sin TLS: compromiso de sesión sin interacción con el servidor	TB-1 (§4.1.3)	V12.2.1 (L1)
TM-05	Divulgación (I)	TA-01	F1	<i>Token</i> JWT almacenado en <i>localStorage</i> : accesible mediante scripts arbitrarios ante inyección de contenido en el cliente	§4.1.2, F1	V7.3.2 (L3)
TM-06	Denegación (D)	TA-01	F1	Ausencia de <i>rate limiting</i> en <i>POST /auth/login</i> : fuerza bruta y agotamiento computacional BCrypt amplificado por CVE-2025-22228	CVE-2025-22228 (CVSS 7.4)	V6.1.1 (L1)
TM-07	Suplantación (S)	TA-02, TA-03	F2	Ausencia de RBAC en servidor: credencial válida equivale a acceso irrestricto sobre la totalidad de la colección PHI sin validación de propiedad ni necesidad de conocer	§4.1.2, F2	V8.2.1 (L1), V8.2.2 (L1)
TM-08	Manipulación (T)	TA-02, TA-03	F2	<i>Mass assignment</i> sobre entidades clínicas JPA: sobrescritura arbitraria de campos PHI no previstos en los controladores REST, incluyendo atributos de clasificación diagnóstica	S-06 a S-13 (CWE-915)	V5.1.2 (L1)
TM-09	Manipulación (T)	TA-02, TA-03, TA-04	F2	AES-ECB determinístico: inferencia de patrones estructurales sobre PHI cifrada mediante análisis diferencial sobre campos de baja entropía semántica (códigos CIE-10, estados de tratamiento)	S-04 / S-05 (CWE-327)	V11.3.3 (L2)
TM-10	Repudio (R)	TA-02, TA-03	F2	Ausencia de registros persistentes de accesos de lectura y escritura sobre PHI: ninguna operación clínica es atribuible de forma forense ni auditable ante requerimiento regulatorio	REQ-003 / HIPAA §164.312(b)	V16.3.2 (L3), V16.4.2 (L2)
TM-11	Divulgación (I)	TA-01	F2 / TB-1	PHI transmitida en texto plano sobre HTTP sin TLS en el tramo navegador → API: interceptación pasiva de datos clínicos en tránsito sin requerir autenticación	TB-1 (§4.1.3)	V12.2.1 (L1)
TM-12	Divulgación (I)	TA-04	F2	AES-ECB revela patrones estructurales de datos clínicos cifrados en reposo: análisis de volcados de base de datos permite inferir contenido PHI sin clave de descifrado	S-04 / S-05 (CWE-327)	V11.2.1 (L2)
TM-13	Denegación (D)	TA-01	F2	RCE y DoS sobre la totalidad del <i>backend</i> mediante vulnerabilida-	CVE-2025-24813 / CVE-2024-50379 (CVSS 9.8)	V13.1.2 (L3)



				des críticas en el servidor de aplicaciones embebido		
TM-14	Elevación (E)	TA-02	F2	Ausencia de RBAC en servidor: usuario con rol MEDICO accede y opera sobre módulos y registros PHI fuera de su asignación clínica sin mecanismo de control que lo impida	§4.1.2, F2	V8.2.4 (L3)
TM-15	Elevación (E)	TA-01	F2	Bypass de la capa de autorización sobre endpoints de la API sin requerir credenciales válidas: acceso externo equivalente al de TA-03 sobre módulos PHI	CVE-2024-38821 (CVSS 9.1), CVE-2026-22732 (CVSS 9.1)	V8.2.1 (L1)
TM-16	Suplantación (S)	TA-02, TA-03	F3	Ausencia de validación binaria mediante magic numbers: archivo no clínico inyectado como documentación diagnóstica válida comprometiendo la integridad de la ficha clínica sin modificar metadatos	§4.1.2, F3	V5.1.2 (L1)
TM-17	Manipulación (T)	TA-02, TA-03	F3	Ausencia de controles de integridad sobre archivos almacenados en GridFS: sustitución de contenido binario no detectable sin huellas criptográficas sobre los archivos	§4.1.2, F3	V11.3.3 (L2)
TM-18	Manipulación (T)	TA-01, TA-02	F3	Path traversal en GET /api/files/{rut}/{solicitudId}: construcción de rutas maliciosas para acceder a recursos fuera del ámbito autorizado del controlador	CVE-2024-38816 / CVE-2024-38819 (CVSS 7.5)	V5.2.1 (L1)
TM-19	Divulgación (I)	TA-01	F3 / TB-1	Imágenes diagnósticas e informes PDF transmitidos en texto claro sobre HTTP sin TLS: interceptación pasiva de archivos médicos binarios en tránsito	TB-1 (§4.1.3)	V12.2.1 (L1)
TM-20	Denegación (D)	TA-01	F3	Carga de archivos de hasta 200 MB sin limitación de tasa ni controles de cuota: agotamiento de recursos del servidor amplificado por RCE activo en Tomcat	CVE-2025-24813 (CVSS 9.8)	V15.2.2 (L2)
TM-21	Divulgación (I)	TA-04	TB-2	Cadena de conexión MongoDB con credenciales expuesta en repositorio: acceso directo a la colección PHI sin interacción con el sistema en ejecución y sin trazas en logs de aplicación	S-01 (CWE-798), Tabla 5	V13.3.1 (L2)
TM-22	Manipulación (T)	TA-04	TB-2	Ausencia de rotación de credenciales de base de datos: reutilización indefinida de secretos expuestos habilita ventana de compromiso ilimitada sobre MongoDB Atlas	Tabla 5	V13.3.1 (L2)

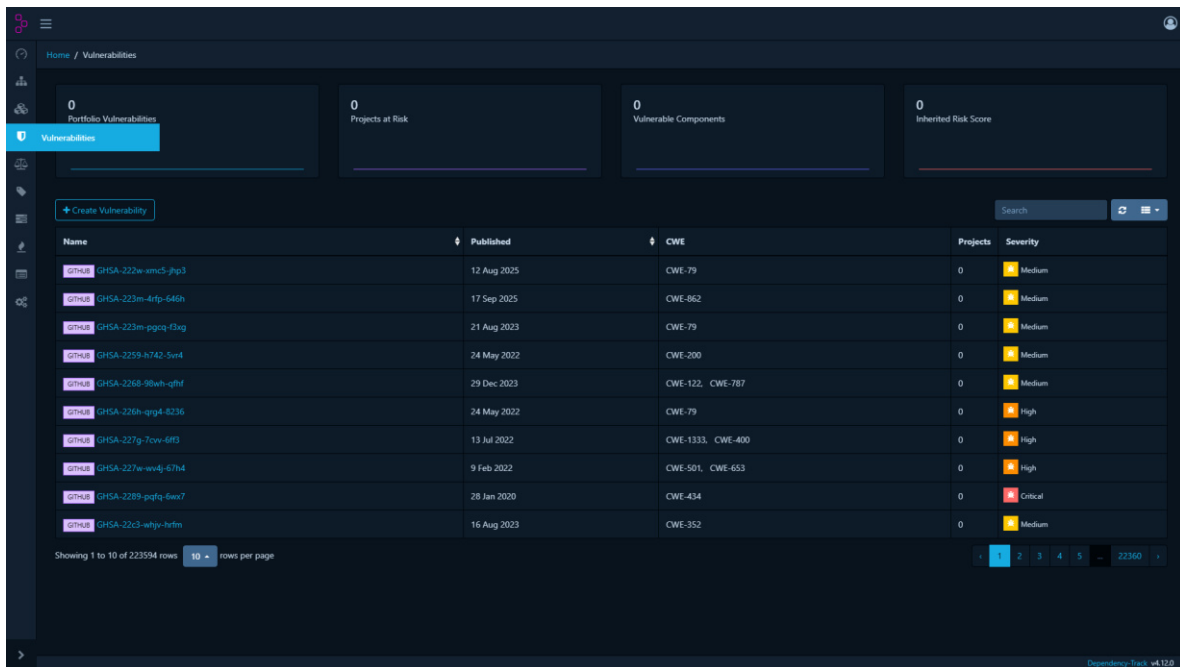


Figura A-1. Vista del proyecto "SIGO Backend" en OWASP Dependency-Track, pestaña "Overview"
Fuente: Elaboración propia con Dependency-Track



Anexos B Matrices del gap analysis

Tabla B - 1. Matriz de brechas de seguridad: controles ASVS v5.0.0 evaluados sobre la arquitectura *as-is* de SIGO

Fuente: elaboración propia

GAP-ID	Domini- nio ASVS	Control	Ni- vel	Es- tado	Hallazgo vinculado	CVSS v3.1	Im- pacto PHI	$R(g)$
GAP-01	V6 Au- tenti- cación	V13.3.1 — Secre- tos de autentica- ción no almace- nados en código fuente	L2	No Cum- ple	S-01 (JWT <i>secret</i> , CWE-798), S-02/S-03 (API key OpenAI, CWE- 798); TM-01	8.8 (est. CWE-798)	Alto	Crí- tico
GAP-02	V6 Au- tenti- cación	V6.3.3 — MFA para acceso a módulos PHI	L3	No Cum- ple	TM-06 (ausencia de segundo factor en ac- ceso PHI; amplificado por CVE-2025-22228)	7.5 (est. STRIDE- TM01)	Alto	Crí- tico
GAP-03	V6 Au- tenti- cación	V6.1.1 — <i>Rate li- miting</i> en <i>endpoint</i> de au- tentificación	L1	No Cum- ple	TM-06; CVE-2025- 22228 (7.4)	7.4	Alto	Crí- tico
GAP-04	V6 Au- tenti- cación	V6.2.5 — Algorit- mos criptográfi- cos aprobados (sin AES-ECB)	L2	No Cum- ple	S-04/S-05 (AES-ECB, CWE-327)	7.4 (est. CWE-327)	Alto	Crí- tico
GAP-05	V7 Se- sión	V7.3.2 — <i>Token</i> JWT no debe al- macenarse en <i>lo- calStorage</i>	L3	No Cum- ple	TM-05 (<i>token</i> ex- puesto a <i>scripts</i> arbi- trarios)	5.9 (est. STRIDE- TM05)	Alto	Crí- tico
GAP-06	V8 Au- toriza- ción	V8.2.1 — Control de acceso basado en roles en el servidor	L1	No Cum- ple	TM-07, TM-15; CVE- 2024-38821 (9.1)	9.1	Alto	Crí- tico
GAP-07	V8 Au- toriza- ción	V8.2.2 — Valida- ción de propie- dad del recurso solicitado	L1	No Cum- ple	TM-07 (acceso irres- tricto a PHI de cual- quier paciente)	8.1 (est. STRIDE- TM07)	Alto	Crí- tico
GAP-08	V8 Au- toriza- ción	V8.2.4 — Contro- les adaptativos de autorización	L3	No Cum- ple	TM-14; CVE-2026- 22732 (9.1)	9.1	Alto	Crí- tico
GAP-09	V11 Cripto- grafía	V11.2.1 — Uso de implementa- ciones criptográ- ficas aprobadas	L2	No Cum- ple	S-04/S-05 (AES-ECB), TM-12 (análisis dife- rencial sobre PHI ci- frada)	7.4 (est. CWE-327)	Alto	Crí- tico
GAP-10	V11 Cripto- grafía	V11.3.3 — Ci- frado autenti- cado (AES-GCM o MAC aprobado)	L2	No Cum- ple	S-04/S-05, TM-09 (in- ferencia de patrones CIE-10 en PHI cifrada)	7.4 (est. CWE-327)	Alto	Crí- tico

GAP-11	V12 Comu- nica- ción	V12.2.1: TLS obli- gatorio en comu- nicación cliente- API (TB-1)	L1	No Cum- ple	TM-02, TM-04, TM-11, TM-19 (PHI en texto plano sobre HTTP)	7.5 (est. STRIDE- TM02)	Alto	Crí- tico
GAP-12	V12 Comu- nica- ción	V12.2.1: TLS en comunicación API-MongoDB Atlas (TB-2)	L1	Cum- ple	TLS nativo gestionado por Atlas	—	—	—
GAP-13	V12 Comu- nica- ción	V13.1.2: Preven- ción de pérdida de disponibilidad por DoS	L3	No Cum- ple	TM-13, TM-20; CVE- 2025-24813 (9.8), CVE-2024-50379 (9.8)	9.8	Alto	Crí- tico
GAP-14	V12 Comu- nica- ción	V15.2.2: <i>Rate li- miting</i> en carga de recursos (ar- chivos)	L2	No Cum- ple	TM-20 (<i>upload</i> 200 MB sin cuota); CVE- 2025-24813 (9.8)	9.8	Alto	Crí- tico
GAP-15	V14 Datos	V14.1.2: Requisi- tos formales de cifrado para da- tos sensibles	L2	Par- cial	S-04/S-05 (AES-ECB no provee confidenciali- dad semántica)	7.4 (est. CWE-327)	Alto	Crí- tico
GAP-16	V14 Datos	V14.2.7: Clasifi- cación y elimina- ción de registros con plazos nor- mativos	L3	No Cum- ple	§4.1.2 — sin meca- nismo de retención ni eliminación automati- zada de PHI	5.0 (est. STRIDE- §4.1.2)	Alto	Crí- tico
GAP-17	V14 Datos	V13.3.1: Creden- ciales de base de datos no expues- tas en repositorio	L2	No Cum- ple	TM-21, TM-22 (cadena de conexión MongoDB estática, S-01)	8.8 (est. CWE-798)	Alto	Crí- tico
GAP-18	V2 Va- lida- ción	V5.1.2: Preven- ción de <i>mass as- signment</i> en con- troladores REST	L1	No Cum- ple	S-06 a S-13 (CWE-915, 8 controladores PHI expuestos); TM-08	7.5 (est. CWE-915)	Alto	Crí- tico
GAP-19	V2 Va- lida- ción	V5.2.1: Preven- ción de <i>path</i> tra- versal en servicio de archivos	L1	No Cum- ple	TM-18; CVE-2024- 38816 (7.5), CVE- 2024-38819 (7.5)	7.5	Alto	Crí- tico
GAP-20	V2 Va- lida- ción	V5.1.2: Valida- ción de tipo MIME en carga de archivos mé- dicos	L1	No Cum- ple	TM-16 (archivo no clí- nico inyectable como documento médico)	6.5 (est. STRIDE- TM16)	Alto	Crí- tico
GAP-21	V16 Auditoría	V16.3.2: Registro de accesos de lectura y escri- tura sobre PHI	L3	No Cum- ple	TM-03, TM-10 (sin au- dit trail; HIPAA §164.312(b))	5.5 (est. STRIDE- TM10)	Alto	Crí- tico
GAP-22	V16 Auditoría	V16.4.2: Protec- ción de registros de auditoría con- tra modificación	L2	No Cum- ple	TM-10 (<i>logs</i> sin inmu- tabilidad ni control de acceso)	6.5 (est. STRIDE- TM10)	Alto	Crí- tico



Tabla B - 2. Matriz de trazabilidad: GAP-ID → Requisito normativo → Base legal

Fuente: elaboración propia

GAP-ID	Control ASVS	REQ bloqueado	Base legal
GAP-01	V13.3.1 — Secretos en código fuente	REQ-006a	HIPAA §164.312(d) — Autenticación de persona o entidad
GAP-02	V6.3.3 — MFA en módulos PHI	REQ-006a	HIPAA §164.312(d) — Autenticación de persona o entidad
GAP-03	V6.1.1 — <i>Rate limiting</i> en autenticación	REQ-006a	HIPAA §164.312(d) — Autenticación de persona o entidad
GAP-04	V6.2.5 — Algoritmos criptográficos aprobados	REQ-002, REQ-005	Ley N.º 19.628 Art. 11; HIPAA §164.312(a)(2)(iv); HIPAA §164.312(c)
GAP-05	V7.3.2 — JWT fuera de <i>localStorage</i>	REQ-006a	HIPAA §164.312(d) — Gestión de sesión autenticada
GAP-06	V8.2.1 — RBAC en servidor	REQ-001	Ley N.º 20.584 Arts. 12 y 13; HIPAA §164.312(a)(1)
GAP-07	V8.2.2 — Validación de propiedad del recurso	REQ-001	Ley N.º 20.584 Arts. 12 y 13; HIPAA §164.312(a)(1)
GAP-08	V8.2.4 — Controles adaptativos de autorización	REQ-001	Ley N.º 20.584 Arts. 12 y 13; HIPAA §164.312(a)(1)
GAP-09	V11.2.1 — Implementaciones criptográficas aprobadas	REQ-002	Ley N.º 19.628 Art. 11; Ley N.º 20.584 Art. 12; HIPAA §164.312(a)(2)(iv)
GAP-10	V11.3.3 — Cifrado autenticado (AES-GCM)	REQ-005	HIPAA §164.312(c) — Integridad
GAP-11	V12.2.1 — TLS en TB-1	REQ-002	Ley N.º 19.628 Art. 11; Ley N.º 20.584 Art. 12; HIPAA §164.312(a)(2)(iv)
GAP-12	V12.2.1 — TLS en TB-2	— (Cumple)	—
GAP-13	V13.1.2 — Prevención DoS/RCE	REQ-007	Ley N.º 21.258 Arts. 1 y 7 — Continuidad oncológica
GAP-14	V15.2.2 — <i>Rate limiting</i> en carga de archivos	REQ-007	Ley N.º 21.258 Arts. 1 y 7 — Continuidad oncológica
GAP-15	V14.1.2 — Cifrado formal de datos sensibles	REQ-002	Ley N.º 19.628 Art. 11; Ley N.º 20.584 Art. 12; HIPAA §164.312(a)(2)(iv)
GAP-16	V14.2.7 — Retención y eliminación de PHI	REQ-004	Ley N.º 21.258 Art. 8 bis — Retención de registros oncológicos
GAP-17	V13.3.1 — Credenciales de BD en repositorio	REQ-002	Ley N.º 19.628 Art. 11; HIPAA §164.312(a)(2)(iv)
GAP-18	V5.1.2 — <i>Mass assignment</i> en controladores REST	REQ-005	HIPAA §164.312(c) — Integridad de datos clínicos
GAP-19	V5.2.1 — <i>Path</i> traversal en servicio de archivos	REQ-002	Ley N.º 19.628 Art. 11; Ley N.º 20.584 Art. 12
GAP-20	V5.1.2 — Validación de tipo MIME	REQ-005	HIPAA §164.312(c) — Integridad del repositorio clínico
GAP-21	V16.3.2 — <i>Audit trail</i> sobre accesos PHI	REQ-003	HIPAA §164.312(b) — Controles de auditoría
GAP-22	V16.4.2 — Protección de registros de auditoría	REQ-003	HIPAA §164.312(b) — Controles de auditoría

Las brechas sin CVE asignado corresponden a deficiencias de diseño o ausencias de control sin vulnerabilidad explotable directa registrada en NVD. Para estas, la puntuación CVSS v3.1 se estima mediante dos métodos mutuamente excluyentes, aplicados en orden de prioridad:

- **(a) Score de referencia CWE en NVD (prioridad alta):** cuando la brecha puede atribuirse a una clase de debilidad catalogada, se utiliza el Base Score publicado por NVD para el CWE correspondiente. Este método garantiza trazabilidad a una fuente objetiva externa reproducible por cualquier evaluador.
- **(b) Caracterización de amenaza STRIDE (prioridad secundaria):** cuando la brecha no tiene un CWE con Base Score disponible en NVD, típicamente ausencias de control sin debilidad técnica directa catalogada, se construye el vector CVSS estimando cada componente a partir del tipo de amenaza STRIDE documentada en la Tabla A-2, conforme a la metodología de NIST SP 800-30 Rev. 1 §3.2 (*threat characterization*).

Notación del vector CVSS v3.1

El vector CVSS v3.1 expresa la severidad de una vulnerabilidad mediante ocho métricas codificadas, agrupadas en dos dimensiones: **capacidad de explotación** (cómo llega el atacante) e **impacto** (qué daño produce).

Dimensión	Código	Métrica	Indicador analítico	Valores y pesos específicos
Explotabilidad	AV	<i>Attack Vector</i>	Dominios de red requeridos para alcanzar el componente vulnerable.	N: Red pública (Internet) A: Red adyacente (local/WiFi) L: Acceso local (CLI/OS) P: Acceso físico
	AC	<i>Attack Complexity</i>	Condiciones de entorno o factores externos requeridos para la explotación.	L: Baja (sin precondiciones) H: Alta (requiere condiciones de carrera o configuraciones específicas)
	PR	<i>Privileges Required</i>	Nivel de autenticación y autorización previo a la ejecución del ataque.	N: Ninguno L: Privilegios básicos (usuario) H: Privilegios elevados (administrador)
	UI	<i>User Interaction</i>	Requerimiento de acciones conscientes por parte de un usuario legítimo.	N: No requiere interacción R: Requiere interacción (clic, descarga)
Alcance	S	<i>Scope</i>	Capacidad de propagación del compromiso hacia componentes de software ajenos.	U: Sin cambio (daño contenido) C: Alcance modificado (afecta otros servicios)
Impacto	C	<i>Confidentiality</i>	Magnitud del acceso no autorizado a la lectura de datos del componente.	N: Sin impacto L: Impacto parcial (exposición limitada) H: Impacto total (extracción irrestricta)
	I	<i>Integrity</i>	Capacidad de alteración, inserción o modificación no autorizada de datos.	N: Sin impacto L: Impacto parcial (mutación acotada) H: Impacto total (corrupción masiva)
	A	<i>Availability</i>	Nivel de degradación, interrupción o denegación de servicios del sistema.	N: Sin impacto L: Impacto parcial (degradación)



H: Impacto total (caída completa)

Tabla B - 3. Metodología de estimación CVSS v3.1 para brechas sin CVE asignado
Fuente: elaboración propia

GAP-ID	Score est.	Método	Vector CVSS v3.1 (Estimado)	Criterio lógico de derivación e impacto estructural
GAP-01	8.8	(a) CWE	AV:N/AC:L/PR:L/UI:N/S:U/C:H/I:H/A:N	CWE-798 (hard-coded credentials): Acceso por red pública; complejidad baja; explotación mediante el uso de privilegios obtenidos del código fuente (PR:L); impacto total en confidencialidad e integridad sobre el <i>backend</i> completo (S-01).
GAP-02	7.5	(b) STRIDE	AV:N/AC:H/PR:N/UI:N/S:U/C:H/I:H/A:N	Spoofing de tokens JWT: Requiere una condición previa compleja (obtención del secreto criptográfico expuesto en el repositorio; AC:H); una vez forjado el <i>token</i> , no requiere autenticación intermedia (PR:N), otorgando acceso e impacto masivo (C:H/I:H) sobre las colecciones PHI.
GAP-04	7.4	(a) CWE	AV:N/AC:H/PR:N/UI:N/S:U/C:H/I:H/A:N	CWE-327 (broken cryptographic algorithm): Explotación con complejidad alta sobre el canal cifrado en reposo; permite el quiebre de la confidencialidad (C:H) mediante análisis diferencial estadístico de bloques idénticos (S-04/S-05).
GAP-05	5.9	(b) STRIDE	AV:N/AC:H/PR:L/UI:R/S:U/C:H/I:L/A:N	Information disclosure (LocalStorage): Exposición del JWT en el lado del cliente; requiere una precondition compleja de inyección de código (XSS, AC:H) e interacción obligatoria del usuario (UI:R); impacto total en confidencialidad de la sesión activa (C:H).
GAP-07	8.1	(b) STRIDE	AV:N/AC:L/PR:L/UI:N/S:U/C:H/I:H/A:N	CWE-639 (insecure direct object reference - IDOR): Acceso por red pública; complejidad baja; requiere rol autenticado básico (PR:L); al omitirse la validación de propiedad en la capa de negocio <i>backend</i> , el impacto es total sobre la integridad y confidencialidad clínica (C:H/I:H).
GAP-09	7.4	(a) CWE	AV:N/AC:H/PR:N/UI:N/S:U/C:H/I:H/A:N	Misma raíz estructural que GAP-04: Referenciado bajo el dominio técnico de criptografía (V11.2.1); uso de algoritmos no aprobados por la indus-

				tria que anulan el secreto clínico (CWE-327).
GAP-10	7.4	(a) CWE	AV:N/AC:H/PR:N/UI:N/S:U/C:H/I:H/A:N	Misma raíz estructural que GAP-04: Referenciado bajo la verificación de integridad criptográfica (V11.3.3); ausencia de cifrado autenticado que habilita la manipulación pasiva de bloques en base de datos (CWE-327).
GAP-11	7.5	(b) STRIDE	AV:N/AC:L/PR:N/UI:N/S:U/C:H/I:N/A:N	Information disclosure en Transporte (TB-1): Interceptación pasiva de red pública sin condiciones previas (AV:N/AC:L/PR:N); impacto crítico sobre la confidencialidad de los datos médicos transmitidos en texto plano en los flujos F1, F2 y F3 (C:H).
GAP-15	7.4	(a) CWE	AV:N/AC:H/PR:N/UI:N/S:U/C:H/I:H/A:N	Estado parcial (protección de datos): El cifrado existe formalmente en persistencia pero implementa la primitiva insegura AES-ECB (CWE-327), repitiendo la severidad técnica de GAP-04 por falta de confidencialidad semántica.
GAP-16	5.0	(b) STRIDE	AV:N/AC:L/PR:H/UI:N/S:U/C:L/I:L/A:N	Brecha lógica de cumplimiento normativo (compliance gap): Ausencia de políticas automatizadas de retención/purga; no existe vector de explotación técnica activa; requiere rol de administración avanzado (PR:H) para constatar la vulneración legal del ciclo de vida del dato.
GAP-17	8.8	(a) CWE	AV:N/AC:L/PR:L/UI:N/S:U/C:H/I:H/A:N	CWE-798 (credenciales expuestas de base de datos): Reitera la severidad del fallo analítico de GAP-01; credenciales estáticas de MongoDB Atlas en repositorio público; impacto total directo sobre persistencia (C:H/I:H).
GAP-18	7.5	(a) CWE	AV:N/AC:L/PR:L/UI:N/S:U/C:N/I:H/A:H	CWE-915 (improperly controlled mass assignment): Acceso por red; requiere rol autenticado básico (PR:L); complejidad baja; impacto total en integridad (I:H) y disponibilidad por corrupción de entidades clínicas críticas en 8 controladores concurrentes.
GAP-20	6.5	(b) STRIDE	AV:N/AC:L/PR:L/UI:N/S:U/C:N/I:H/A:N	Tampering / manipulación del repositorio GridFS: Inyección de ejecutables o <i>payloads</i> falsificados como documentos clínicos binarios; complejidad baja; requiere rol básico (PR:L); compromiso directo de la integridad del repositorio clínico (I:H).



GAP-21	5.5	(b) STRIDE	AV:N/AC:L/PR:H/UI:N/S:U/C:N/I:N/A:H	Repudio (ausencia de registros sobre PHI): No habilita un ataque destructivo directo sobre la información; requiere privilegios elevados (PR:H) para explotar la falta de trazabilidad forense; impacto crítico en disponibilidad de auditoría (A:H), violando HIPAA §164.312(b).
GAP-22	6.5	(b) STRIDE	AV:N/AC:L/PR:H/UI:N/S:U/C:N/I:H/A:N	Tampering de registros de auditoría (Logs): Variante del vector anterior; la ausencia de inmutabilidad en la capa de persistencia faculta a un rol privilegiado malicioso (PR:H) a alterar o suprimir trazas existentes, destruyendo la integridad forense (I:H).