

UNIVERSIDAD TÉCNICA FEDERICO SANTA MARÍA
DEPARTAMENTO DE INFORMÁTICA
VALPARAÍSO - CHILE



“ALMACENAMIENTO Y TRATAMIENTO SEGURO DE DATOS DE SALUD EN EL MARCO DE UNA APLICACIÓN DE TELEMEDICINA”

JOSÉ FOLLERT MÜLLER

MEMORIA PARA OPTAR AL TÍTULO DE
INGENIERO CIVIL EN INFORMÁTICA

Profesor Guía: Gabriel Torres
Profesor Correferente: Carolina Saavedra

Julio - 2025



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: ALMACENAMIENTO Y TRATAMIENTO SEGURO DE DATOS DE SALUD EN EL MARCO DE UNA APLICACIÓN DE TELEMEDICINA

Nombre del candidato(a): José Bastián Ignacio Follert Müller

Carrera / Grado: Ingeniería Civil Informática

Campus: Casa Central Valparaíso ; **Departamento:** Informática

2.- VALIDACIÓN DEL PROFESOR GUÍA/DIRECTOR DE TESIS

Yo, Gabriel Torres, 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

☒ 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:

☐ 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: 29/07/2025

; Firma:

Estudiante o Candidato(a):

Fecha: 29/07/2025

; Firma:

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.

DEDICATORIA

A quienes han caminado conmigo en este viaje.
A mi madre, faro incansable cuya fe me sostuvo hasta el final.
A mi familia y amistades, por su afecto y paciencia.
A ustedes dedico este logro.

AGRADECIMIENTOS

A la Universidad Técnica Federico Santa María, por ofrecer el entorno y los recursos que hicieron posible esta memoria.

Al Departamento de Informática, por la sólida formación y los conocimientos que hoy sustentan mi trabajo.

Al Jefe de Carrera, Pedro Godoy, por sus gestiones y apoyo constante a lo largo del proceso.

Y a mi profesor guía, Gabriel Torres, por su paciencia, orientación y permanente disposición para resolver dudas y encaminarme hacia la meta.

A todos ustedes, mi más sincero agradecimiento.

RESUMEN

Resumen— El trabajo presenta el diseño e implementación de un mecanismo que protege la información clínica manejada por una plataforma chilena de tele-otorrinolaringología alojada en AWS bajo arquitectura serverless. Primero se analizó la infraestructura existente y se detectaron brechas: el cifrado no cubría todos los flujos y la gestión de claves era manual. Para remediarlo, se incorporó un esquema de “cifrado en sobre” que genera claves de datos efímeras y las protege con AWS KMS, automatizando la rotación y aplicando políticas de mínimo privilegio. El módulo se integró a las funciones Lambda y a los almacenes DynamoDB y S3 sin alterar la lógica de negocio. Las pruebas funcionales y de seguridad demostraron que los datos permanecen cifrados desde el origen, no se degrada la experiencia de usuario y se dificulta la exfiltración incluso ante accesos ilícitos. Con ello, la aplicación cumple la Ley 21.663 de Chile y se alinea con los marcos internacionales HIPAA y GDPR, reduciendo significativamente los riesgos legales y reputacionales.

Palabras Clave— Cifrado; Telemedicina; Serverless; AWS; Seguridad

ABSTRACT

Abstract— This thesis describes the design and implementation of a mechanism that safeguards clinical information handled by a Chilean tele-otolaryngology platform running on AWS in a serverless architecture. An initial assessment revealed gaps: encryption did not cover every data flow and key management was manual. To address this, an envelope-encryption scheme was introduced, generating ephemeral data keys protected by AWS KMS, with automated key rotation and least-privilege policies. The module was integrated into Lambda functions and the DynamoDB and S3 stores without altering business logic. Functional and security tests showed that data remain encrypted from the point of origin, user experience is unaffected, and exfiltration is thwarted even if illicit access occurs. Consequently, the application now complies with Chile’s Law 21.663 and aligns with international frameworks such as HIPAA and GDPR, markedly reducing legal and reputational risk.

Keywords— Encryption; Telemedicine; Serverless; AWS; Security

GLOSARIO

AC3E: Centro Avanzado de Ingeniería Eléctrica y Electrónica.
ACM: AWS Certificate Manager.
AES: Advanced Encryption Standard.
API: Application Programming Interface.
ARN: Amazon Resource Name.
ATA: American Telemedicine Association.
AWS: Amazon Web Services.
CDK: Cloud Development Kit.
CI/CD: Continuous Integration / Continuous Delivery.
CLI: Command Line Interface.
CMK: Customer Master Key.
DDD: Domain-Driven Design.
EHR: Electronic Health Records.
ECC: Elliptic Curve Cryptography.
EDK: Encryption Data Key.
GDPR: General Data Protection Regulation.
HIPAA: Health Insurance Portability and Accountability Act.
HSM: Hardware Security Module.
IaaS: Infrastructure as a Service.
IAM: Identity and Access Management.
IaC: Infrastructure as Code.
JSON: JavaScript Object Notation.
JWT: JSON Web Token.
KMS: Key Management Service.
NIST: National Institute of Standards and Technology.
NoSQL: No Structured Query Language.
PaaS: Platform as a Service.
PHI: Protected Health Information.
PYME: Pequeña y Mediana Empresa.
S3: Simple Storage Service.
SaaS: Software as a Service.
SDK: Software Development Kit.
SSE: Server-Side Encryption.
TI: Tecnología de la Información.
TIC: Tecnologías de la Información y la Comunicación.
TLS: Transport Layer Security.
UTFSM: Universidad Técnica Federico Santa María.
XaaS: Everything as a Service.

ÍNDICE DE CONTENIDOS

RESUMEN	IV
ABSTRACT	IV
GLOSARIO	V
ÍNDICE DE FIGURAS	IX
ÍNDICE DE TABLAS	IX
INTRODUCCIÓN	1
1 DEFINICIÓN DEL PROBLEMA	3
1.1 CONTEXTO GENERAL	3
1.2 APLICACIÓN DE TELEMEDICINA	4
1.2.1 OBJETIVOS Y FUNCIONALIDADES	4
1.2.2 PROCESO Y FLUJO DE TRABAJO	4
1.2.3 TECNOLOGÍAS, ARQUITECTURA E INFRAESTRUCTURA	5
1.3 DESAFÍOS DE LA SEGURIDAD DE LA INFORMACIÓN	6
1.3.1 CUMPLIMIENTO CON ESTÁNDARES INTERNACIONALES	7
1.3.2 MEDIDAS DE SEGURIDAD ESPECÍFICAS	8
1.3.3 IMPACTO DE UNA EXFILTRACIÓN DE DATOS	8
1.4 SEGURIDAD DE DATOS EN CLOUD SERVERLESS	9
1.4.1 VENTAJAS DE LA ARQUITECTURA SERVERLESS	9
1.4.2 CONSIDERACIONES DE SEGURIDAD EN ARQUITECTURA SERVERLESS	10
1.4.3 MINIMIZACIÓN DE RESPONSABILIDADES SOBRE CAPAS DE SEGURIDAD	10
1.5 ENFRENTANDO LOS DESAFÍOS DE SEGURIDAD EN LA TELEMEDICINA	11
1.5.1 PRIORIZACIÓN DEL CIFRADO DE DATOS	11
1.5.2 DESARROLLO SOSTENIBLE Y SEGURIDAD A LARGO PLAZO	12
1.6 OBJETIVOS DEL TRABAJO	12
1.6.1 OBJETIVO GENERAL	13
1.6.2 OBJETIVOS ESPECÍFICOS	13
CAPÍTULO 2: MARCO CONCEPTUAL	14
2.1 TELESALUD Y TELEMEDICINA	14
2.2 SEGURIDAD DE LA INFORMACIÓN EN SALUD	14
2.3 CLOUD COMPUTING	15
2.4 AMENAZAS DE SEGURIDAD EN TELEMEDICINA	16
2.5 MODELO DE RESPONSABILIDAD COMPARTIDA EN ARQUITECTURAS SERVERLESS	17
2.6 RETOS DE LA SEGURIDAD PARA TELEMEDICINA EN CLOUD	18
2.7 PROTECCIÓN DE DATOS Y TECNOLOGÍAS DE CIFRADO	19
2.7.1 ALGORITMOS DE CIFRADO	19

2.7.2	CIFRADO SIMÉTRICO	19
2.7.3	GESTIÓN DE CLAVES DE CIFRADO	20
2.7.4	HARDWARE SECURITY MODULE (HSM)	20
2.8	REGULACIÓN INTERNACIONAL EN PROTECCIÓN DE DATOS	21
2.8.1	HIPAA (HEALTH INSURANCE PORTABILITY AND ACCOUNTABILITY ACT)	21
2.8.2	GDPR (GENERAL DATA PROTECTION REGULATION)	22
2.8.3	OTROS ESTÁNDARES Y DIRECTRICES INTERNACIONALES	22
2.8.4	COMPARACIÓN HIPAA VS GDPR Y BUENAS PRÁCTICAS PARA TELEMEDICINA	22
2.9	REGULACIÓN CHILENA DE PROTECCIÓN DE DATOS	23
2.9.1	LEY 19.628 SOBRE PROTECCIÓN DE LA VIDA PRIVADA (1999)	23
2.9.2	LEY 21.663 QUE FORTALECE LA PROTECCIÓN DE DATOS PERSONALES (2024)	24
2.9.3	COMPATIBILIDAD CON EL GDPR Y OTROS ESTÁNDARES	25
2.9.4	IMPACTO EN EL SECTOR SALUD Y LA TELEMEDICINA	25
CAPÍTULO 3: PROPUESTA DE SOLUCIÓN		26
3.1	ANÁLISIS DE LA ARQUITECTURA ACTUAL	26
3.1.1	ARQUITECTURA A NIVEL DE INFRAESTRUCTURA	26
3.1.2	ARQUITECTURA A NIVEL DE CÓDIGO	27
3.1.3	PUNTOS CRÍTICOS DEL USO DE DATOS	28
3.2	DISEÑO DE LA SOLUCIÓN	31
3.2.1	ESTRATEGIA DE CIFRADO DE DATOS	32
3.2.2	GESTIÓN Y ROTACIÓN DE CLAVES CRIPTOGRÁFICAS	33
3.2.3	INTEGRACIÓN DE SEGURIDAD Y PRIVACIDAD POR DISEÑO	35
3.2.4	FLUJOS DE CIFRADO Y DESPLIEGUE	36
3.2.5	ESCALABILIDAD Y CONTINUIDAD DE LA SOLUCIÓN	37
3.2.6	ASPECTOS CLAVES DEL CUMPLIMIENTO NORMATIVO	37
3.3	INTEGRACIÓN CON LA ARQUITECTURA EXISTENTE	39
3.3.1	ADAPTACIÓN DE MÓDULOS EN LA CAPA DE INFRAESTRUCTURA	39
3.3.2	MANTENIMIENTO DE LOS FLUJOS EXISTENTES Y MIGRACIÓN PROGRESIVA	40
3.3.3	ALINEAMIENTO CON PRINCIPIOS DE CLEAN ARCHITECTURE Y DDD	40
3.3.4	PASOS TÉCNICOS PARA LA IMPLEMENTACIÓN	42
3.3.5	CONSIDERACIONES DE DESEMPEÑO Y CONFIABILIDAD	42
CAPÍTULO 4: VALIDACIÓN DE LA SOLUCIÓN		44
4.1	IMPLEMENTACIÓN DEL MÓDULO DE CIFRADO	44
4.1.1	CIFRADO EN EL LADO DEL SERVIDOR	44
4.1.2	CIFRADO EN EL LADO DE LA APLICACIÓN	46
4.1.3	GESTIÓN DE PERMISOS Y CLAVES	48
4.2	VALIDACIÓN DEL CIFRADO DE DATOS EN TRÁNSITO	50
4.2.1	ESCENARIO 1: ENDPOINT POR DEFECTO DE API GATEWAY	51
4.2.2	ESCENARIO 2: DOMINIO PERSONALIZADO	51

4.3 PRUEBAS DE SEGURIDAD Y FUNCIONALIDAD	53
4.3.1 PRUEBAS DE INTEGRACIÓN FUNCIONAL (HAPPY-PATH)	53
4.3.2 PRUEBAS DE CONTROL DE ACCESO Y SIMULACIÓN DE AMENAZAS (NEGATIVE-PATHS)	56
4.4 ANÁLISIS Y DISCUSIÓN DE RESULTADOS	60
CAPÍTULO 5: CONCLUSIONES	61
ANEXOS	67
5.1 CÓDIGO FUENTE DE LA SOLUCIÓN DE CIFRADO	67
5.1.1 Repositorio de Código Fuente	67
5.1.2 Cifrado en el Lado del Servidor con Aspects de AWS CDK	67
5.1.3 Cifrado en el Lado de la Aplicación con AWS Encryption SDK	70
REFERENCIAS BIBLIOGRÁFICAS	73

ÍNDICE DE FIGURAS

1	Flujo de información manejado por la aplicación de telemedicina.	5
2	Individuos afectados por brechas de seguridad en el sector de la salud entre 2009 y 2023.	6
3	Modelo de responsabilidad compartida en arquitectura serverless.	11
4	Diferencias entre IaaS, SaaS, PaaS y XaaS.	16
5	Siete áreas relacionadas con las amenazas a la seguridad de la telemedicina . .	17
6	Comparación entre cifrado simétrico y asimétrico.	20
7	Arquitectura de la Infraestructura de la Aplicación de Telemedicina en Estudio .	27
8	Arquitectura del código de la Aplicación de Telemedicina en Estudio	29
9	Flujo del cifrado en sobre	32
10	Resultados de ejecución del Comando 1. Error de Handshake	52
11	Resultados de ejecución del Comando 2. Conexión Exitosa	53
12	Formulario de Información del Paciente.	54
13	Casos Clínicos Registrados en DynamoDB	55
14	Respuesta de API con Casos Clínicos	55
15	Resultados Simulador de Permisos - UnauthorizedRole.	57
16	Consulta de Caso Clínico Directa a DynamoDB	57

ÍNDICE DE TABLAS

1	Matriz de Pruebas de Validación de la Solución	59
---	--	----

INTRODUCCIÓN

La pandemia de COVID-19 aceleró un cambio que ya se veía venir: hoy la atención médica remota es parte del día a día. Plataformas de telemedicina permiten que un paciente sea evaluado a cientos de kilómetros de un especialista y, en muchos casos, reciba diagnóstico en horas en lugar de semanas. Sin embargo, esa comodidad trae un desafío enorme: la protección de los datos de salud. La información clínica —diagnósticos, exámenes, antecedentes— es extremadamente sensible; en las manos equivocadas puede usarse para discriminar, extorsionar o suplantar identidades. No es casualidad que los ciberataques al sector salud vengán aumentando sin pausa en los últimos años [HIPAA Journal, 2023a].

En Chile, la *Ley 19.628* clasifica estos datos como “sensibles” y la nueva *Ley 21.663* refuerza su resguardo convirtiendo la ciberseguridad en un tema país. A nivel internacional, el *GDPR* europeo y la *HIPAA* estadounidense fijan multas que pueden arruinar una empresa si se produce una fuga de información. El mensaje es claro: o protegemos los datos o asumimos riesgos legales, económicos y reputacionales muy serios.

CONTEXTO DEL CASO DE ESTUDIO

Este trabajo se desarrolla en torno a una *startup* de la Región de Valparaíso que ofrece un servicio de tele-otorrinolaringología. Su aplicación permite a técnicos de centros de salud capturar videos endoscópicos y antecedentes del paciente, almacenarlos en la nube y ponerlos a disposición de especialistas que entregan el diagnóstico a distancia. La plataforma corre íntegramente en AWS usando una arquitectura *serverless*: funciones *Lambda*, API Gateway, DynamoDB y S3, todo orquestado con AWS CDK. Este enfoque hace que la solución escale casi sola y que el equipo se concentre en mejorar la atención; pero también implica que los datos clínicos viajan y se guardan en múltiples servicios gestionados por terceros.

Una revisión inicial detectó puntos débiles: el cifrado no cubría todos los flujos de información y la gestión de claves era manual y propensa a errores. Cualquier atacante que lograra acceso a la base de datos o a los respaldos podía leer datos de salud en texto claro. Con sanciones potenciales que llegan a millones de dólares [Commission, 2018, HIPAA Journal, 2023b], la empresa no puede permitirse incurrir en ese riesgo.

PROPÓSITO DEL TRABAJO

El objetivo principal es diseñar e implementar un mecanismo que proteja los datos clínicos contra filtraciones, con el menor impacto posible sobre la arquitectura existente y sin sacrificar la experiencia de usuario. Para lograrlo se propone:

1. Adoptar *cifrado en sobre (envelope encryption)* de extremo a extremo usando claves simétricas gestionadas por AWS KMS [Weissbaum y Lugin, 2023].
2. Automatizar la rotación y el aislamiento de claves, minimizando la exposición y el tra-

bajo operativo [Krähenbühl y Perrig, 2023a].

3. Integrar el cifrado tanto al almacenar como al transferir la información, de modo transparente para médicos y técnicos.

Con esta estrategia, aunque un atacante obtuviera acceso a la base o a los archivos, solo encontraría datos cifrados con claves que no controla.

METODOLOGÍA

Este trabajo se desarrolló en cuatro pasos:

1. *Análisis de arquitectura*: se modeló el flujo de datos y se identificaron los puntos críticos de exposición.
2. *Diseño de la solución*: se definieron las capas de cifrado, los servicios de AWS a utilizar y los cambios mínimos en la lógica de negocio.
3. *Implementación*: se codificó un módulo transversal en TypeScript que aplica el cifrado, gestiona claves mediante KMS y se despliega con CDK sin afectar los tiempos de respuesta.
4. *Validación*: se verificó, mediante pruebas funcionales y de seguridad, que los datos viajan y se almacenan cifrados y que las funciones siguen cumpliendo con los requisitos de la empresa y las normativas citadas.

ESTRUCTURA DEL DOCUMENTO

El **Capítulo 1** contextualiza el problema y expone los objetivos. El **Capítulo 2** resume los conceptos clave: telemedicina, seguridad de la información, modelos *cloud* y regulación vigente. El **Capítulo 3** detalla la arquitectura actual y las brechas encontradas, además de describir la solución propuesta y su integración con AWS. El **Capítulo 4** presenta la implementación y los resultados de las pruebas. Finalmente, el **Capítulo 5** recoge las conclusiones y sugiere líneas de trabajo futuro.

En síntesis, este trabajo entrega un camino práctico para que una *startup* de telemedicina adopte cifrado robusto sin perder agilidad, garantizando la confidencialidad de los datos de salud y cumpliendo con las exigencias legales tanto chilenas como internacionales. El resultado es un sistema más seguro, preparado para crecer y —lo más importante— confiable para los pacientes y profesionales que lo utilizan.

DEFINICIÓN DEL PROBLEMA

1.1. CONTEXTO GENERAL

La era de la transformación digital en el campo de la salud ha marcado un punto de inflexión en cómo se manejan y se accede a los servicios sanitarios. La adopción de sistemas electrónicos de información clínica y la integración de tecnologías de información en los procesos médicos han logrado avances significativos en la eficiencia y seguridad del manejo de datos de pacientes. Estos desarrollos han revolucionado la forma de almacenar y consultar la información clínica, permitiendo un acceso más rápido, seguro y confiable a datos críticos en el momento en que se necesitan, algo especialmente crucial en situaciones donde un paciente requiere atención urgente o especializada.

La **disponibilidad de información clínica completa y actualizada** para los profesionales de la salud mejora la toma de decisiones clínicas y reduce la duplicación de pruebas y procedimientos, optimizando así los recursos médicos y la calidad de la atención. Una atención más coordinada y centrada en el paciente es posible cuando los datos del paciente están disponibles de forma integral en sistemas digitales; esto se traduce en mejores resultados de salud y una mayor satisfacción del paciente. Sin embargo, este progreso digital no está exento de desafíos. La necesidad de cumplir con estrictos estándares de seguridad y con regulaciones en constante evolución incrementa la complejidad del desarrollo y mantenimiento de los sistemas de información en salud, especialmente al introducir tecnologías avanzadas para la protección de datos.

En este panorama, **la telemedicina** emerge como un componente vital de la medicina moderna. Aprovechando las tecnologías de información y comunicación, la telemedicina se enfoca en superar barreras geográficas y optimizar los recursos médicos para proporcionar servicios de salud de manera más accesible, oportuna y efectiva. Ofrece una alternativa prometedora para llegar a poblaciones en ubicaciones remotas o con acceso limitado a especialistas médicos. Aunque la telemedicina ha mostrado un enorme potencial, no está libre de desafíos. La seguridad y la confidencialidad de la información médica del paciente son preocupaciones constantes. De hecho, la telemedicina ha tenido que enfrentar problemas de privacidad de datos, integridad de la información y autenticación de usuarios desde sus inicios.

Con el avance tecnológico y el aumento en la sofisticación de los ciberataques, la **ciberseguridad** se ha convertido en un aspecto crítico que requiere atención continua en el desarrollo y despliegue de soluciones de telemedicina [Iqbal *et al.*, 2022]. Este entorno de medicina digital y de servicios a distancia proporciona el marco para el desarrollo de la aplicación en estudio. La necesidad de abordar estos retos de manera efectiva y segura resalta la importancia de proponer una solución orientada a mejorar la accesibilidad y la calidad de los servicios de salud mediante la innovación tecnológica, garantizando al mismo tiempo la protección de los datos sensibles de los pacientes.

1.2. APLICACIÓN DE TELEMEDICINA

1.2.1. OBJETIVOS Y FUNCIONALIDADES

El objetivo principal de la aplicación de telemedicina es **incrementar la eficiencia en la atención de otorrinolaringología**, respondiendo a la necesidad de reducir las listas de espera y optimizar el tiempo de respuesta en el diagnóstico y tratamiento de los pacientes. La aplicación facilita una conexión efectiva entre pacientes y especialistas, permitiendo interacciones remotas rápidas y efectivas.

Un aspecto clave de la aplicación es su capacidad para **filtrar casos según su complejidad**. En muchas ocasiones, las consultas de otorrinolaringología involucran afecciones que son relativamente sencillas de abordar. Por ejemplo, un paciente que consulta por una disminución en la audición causada por un tapón de cera puede ser fácilmente diagnosticado y abordado de forma remota. Del mismo modo, casos que requieren exámenes adicionales, como la audiometría para la prescripción de audífonos, pueden ser identificados y derivados adecuadamente sin la necesidad de una consulta presencial inmediata.

Este enfoque de filtración de casos permite una **asignación más eficiente de los recursos médicos** y asegura que los pacientes que requieren atención especializada y presencial de un otorrinolaringólogo la reciban de manera oportuna. Al proporcionar a los especialistas acceso a información detallada sobre cada caso, incluyendo datos clínicos, historiales médicos y resultados de pruebas preliminares, la aplicación mejora significativamente el proceso de triaje y contribuye a una gestión más eficaz de los casos.

1.2.2. PROCESO Y FLUJO DE TRABAJO

El proceso de la aplicación comienza con la atención de pacientes con consultas específicas de otorrinolaringología en centros de salud afiliados. Aquí, un examinador capacitado realiza una recopilación exhaustiva de datos, que incluye:

- **Datos Demográficos:** Información básica del paciente como edad, género y otros detalles relevantes para su identificación y contexto.
- **Historial Médico (Mórbidos):** Detalles sobre condiciones médicas previas, diagnósticos y tratamientos anteriores.
- **Datos Farmacéuticos:** Información sobre medicamentos que el paciente está tomando o ha tomado, incluyendo posibles alergias a medicamentos.
- **Hábitos y Estilo de Vida:** Datos sobre los hábitos de vida del paciente que pueden influir en su condición de salud, como fumar, dieta y ejercicio.

- **Videos Diagnósticos:** Imágenes y videos capturados con el otoendoscopio digital, proporcionando una visión detallada de las áreas afectadas.

Una vez recopilada, esta información se carga en la plataforma para su revisión por especialistas a distancia. Ellos pueden acceder a un perfil completo del paciente, incluyendo visualizaciones detalladas a través de los videos diagnósticos, permitiéndoles proporcionar diagnósticos precisos y recomendaciones de tratamiento específicas. Este enfoque no solo mejora la eficiencia del proceso diagnóstico, sino que también asegura que los casos que requieren atención presencial más compleja reciban la prioridad necesaria. En la figura 1 se puede observar de manera general el flujo de información que maneja la aplicación en estudio.



Figura 1: Flujo de información manejado por la aplicación de telemedicina.

Fuente: Elaboración Propia.

La capacidad de la aplicación para procesar y compartir de manera segura y eficiente esta variedad de datos es fundamental para el éxito del modelo de telemedicina implementado, representando un avance significativo en la calidad y accesibilidad de la atención médica especializada.

1.2.3. TECNOLOGÍAS, ARQUITECTURA E INFRAESTRUCTURA

La plataforma de telemedicina, concebida como un Software como Servicio (SaaS), se fundamenta en una infraestructura tecnológica diseñada para enfrentar los retos emergentes en el ámbito de la salud digital. Esta solución SaaS está desplegada en AWS, aprovechando las ventajas de una infraestructura principalmente serverless, lo cual no solo facilita una gestión eficiente de los recursos y una notable escalabilidad, sino que también plantea desafíos específicos en términos de seguridad de la información.

La elección de AWS y la adopción de una arquitectura serverless reflejan un enfoque estratégico dirigido a optimizar el rendimiento y la disponibilidad del servicio, al tiempo que se abordan de manera proactiva los desafíos inherentes a la seguridad y privacidad de los datos de los pacientes. Este modelo de despliegue implica una colaboración estrecha con el proveedor de servicios en la nube, en este caso AWS, para asegurar una gestión compartida de la seguridad que capitaliza las fortalezas del modelo serverless sin comprometer la protección de la información sensible.

Aunque los detalles técnicos y las implicaciones de seguridad de la arquitectura serverless y la plataforma serán detallados en las secciones subsecuentes, es importante destacar desde ahora la relevancia de estas decisiones arquitectónicas. El compromiso con la seguridad de la información, en un entorno tecnológico que evoluciona rápidamente, es un pilar esencial de nuestra propuesta, buscando no solo cumplir con las expectativas de accesibilidad y calidad en los servicios de salud, sino también garantizar la confidencialidad y la integridad de los datos clínicos.

1.3. DESAFIOS DE LA SEGURIDAD DE LA INFORMACIÓN

La digitalización de los servicios de salud, especialmente en el ámbito de la telemedicina, plantea retos significativos en cuanto a la seguridad y la protección de la información personal de los pacientes. La aplicación de telemedicina descrita anteriormente maneja datos sensibles, incluyendo información demográfica, historiales médicos, datos farmacéuticos, hábitos de vida y registros audiovisuales diagnósticos. La protección de esta información es crítica no solo para mantener la confianza del paciente y cumplir con las regulaciones legales, sino también para prevenir riesgos significativos asociados con la exfiltración de datos, como la pérdida de reputación, sanciones legales y financieras, y el impacto negativo en la atención al paciente.

El creciente número de brechas de seguridad en el sector sanitario subraya la magnitud de este problema. El gráfico que podemos observar en la Figura 2 proporcionado por HIPAA Journal ilustra el alarmante incremento en el número de individuos afectados por brechas de seguridad en el sector de la salud desde 2009 hasta 2023, con un pico notable en 2015 y una tendencia ascendente en años recientes, lo que enfatiza la creciente amenaza de estos incidentes y una clara indicación de la urgencia con la que la industria de la salud debe abordar las vulnerabilidades de seguridad [HIPAA Journal, 2023a].

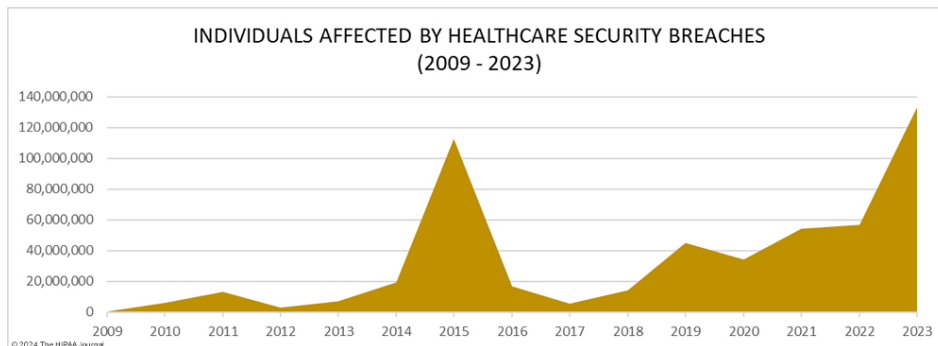


Figura 2: Individuos afectados por brechas de seguridad en el sector de la salud entre 2009 y 2023.

Fuente: HIPAA Journal

La complejidad de las causas detrás de estas brechas es multifacética. Factores comunes

incluyen tácticas de phishing dirigidas a empleados desprevenidos, la presencia de configuraciones de seguridad deficientes y la pérdida o robo de dispositivos que contienen información sensible. Además, los ataques de malware y ransomware han evolucionado, volviéndose más sofisticados y difíciles de detectar y neutralizar. Estos ataques no solo buscan el lucro económico sino que también pueden ser motivados por espionaje o sabotaje, lo que amplifica la necesidad de protecciones robustas y una vigilancia constante [HIPAA Journal, 2023a].

La educación continua sobre la seguridad y la implementación de políticas de gestión de datos son imperativas para mitigar los riesgos asociados con las brechas de seguridad. La industria de la salud debe enfocarse no solo en la protección de datos sino también en la resiliencia operativa, preparando al personal para responder eficazmente en caso de una violación de datos. Las estrategias de protección avanzadas, como la encriptación de extremo a extremo y la autenticación multifactor, deben ser estándares en la protección de información de salud para prevenir accesos no autorizados y para asegurar la continuidad de los servicios críticos de atención médica.

1.3.1. CUMPLIMIENTO CON ESTÁNDARES INTERNACIONALES

En el contexto de la creciente digitalización de los servicios de salud, el cumplimiento con estándares internacionales de protección de datos como la Ley de Portabilidad y Responsabilidad de Seguros de Salud (HIPAA) de los Estados Unidos y el Reglamento General de Protección de Datos (GDPR) de la Unión Europea se ha convertido en un pilar fundamental para la implementación exitosa de aplicaciones de telemedicina. Estos marcos legales no solo establecen directrices para el manejo seguro y responsable de la información de salud de los pacientes, sino que también definen las obligaciones de las entidades que procesan dichos datos. La adopción de estas normativas garantiza la integridad y confidencialidad de la información de salud, imponiendo requisitos rigurosos para la protección de datos, el consentimiento informado del paciente, la transparencia en el manejo de la información y la implementación de medidas técnicas y organizativas que prevengan el acceso no autorizado, la alteración o la pérdida de datos.

El incumplimiento de estándares internacionales como HIPAA y GDPR puede resultar en graves consecuencias financieras y de reputación para entidades de telemedicina. Las sanciones bajo HIPAA pueden alcanzar \$50,000 USD por violación, con un límite de \$1.5 millones USD anuales, mientras que el GDPR puede imponer multas de hasta el 4 % del ingreso global o 20 millones de euros. Además, el no cumplir con estas normativas puede dañar la confianza del paciente y afectar negativamente la relación con los proveedores de salud, subrayando la importancia de la seguridad y privacidad de la información del paciente. La reciente tendencia al alza en las multas, como se evidencia en el aumento del total recaudado por infracciones a HIPAA en 2023 [HIPAA Journal, 2023a], destaca la necesidad crítica de adherirse estrictamente a estas regulaciones.

Por lo tanto, es esencial que las aplicaciones de telemedicina no solo adopten estos estándares

res internacionales desde el inicio de su desarrollo, sino que también incorporen un enfoque proactivo y continuo hacia la seguridad de la información. Esto implica realizar auditorías regulares de seguridad, actualizar las políticas de privacidad y protección de datos conforme a las legislaciones vigentes y en evolución, y asegurar que todo el personal esté adecuadamente capacitado en las mejores prácticas de manejo de datos de salud. Asimismo, el diseño y la implementación de la infraestructura tecnológica deben contemplar mecanismos avanzados de seguridad, como la encriptación de datos, la autenticación multifactor y los controles de acceso basados en roles, para cumplir efectivamente con los requisitos establecidos por estos marcos legales.

1.3.2. MEDIDAS DE SEGURIDAD ESPECÍFICAS

Para cumplir con estos estándares y garantizar la seguridad de la información, la aplicación deberá implementar una serie de medidas de seguridad, que incluyen:

- **Encriptación de Datos:** Toda la información sensible, incluidos los videos diagnósticos, debe ser encriptada tanto en tránsito como en reposo, utilizando algoritmos de cifrado robustos.
- **Gestión de Accesos y Autenticación:** Implementar controles de acceso estrictos basados en roles para asegurar que solo el personal autorizado pueda acceder a la información del paciente. La autenticación multifactor debe ser un requisito para todos los usuarios de la plataforma.
- **Auditoría y Monitoreo:** La aplicación debe registrar todas las actividades de acceso y modificación de datos, permitiendo una trazabilidad completa en caso de una investigación de seguridad o cumplimiento.
- **Acuerdos de Nivel de Servicio (SLAs) y Contratos:** Es crucial definir claramente las responsabilidades de seguridad en los contratos con los establecimientos de salud. Aunque la responsabilidad de la protección de datos puede depender del contrato, es esencial que la aplicación garantice la protección adecuada de la información conforme a los estándares internacionales.
- **Evaluaciones de Riesgo y Cumplimiento:** Realizar evaluaciones de riesgo periódicas y revisiones de cumplimiento para identificar y mitigar vulnerabilidades potenciales en la seguridad de la información.

1.3.3. IMPACTO DE UNA EXFILTRACIÓN DE DATOS

Para una startup de telemedicina, una exfiltración de datos puede ser catastrófica, no solo por las consecuencias inmediatas de las sanciones y la pérdida de confianza, sino también por

el impacto a largo plazo en su reputación y viabilidad financiera. Las startups, al ser entidades en crecimiento y con recursos limitados, enfrentan un desafío mayor al intentar recuperarse de tales incidentes. Además, el daño a la reputación puede dificultar la adquisición de nuevos clientes y la retención de los existentes, lo cual es crucial para el crecimiento y la expansión de la empresa. En este contexto, es vital implementar medidas de seguridad robustas desde el principio y asegurar un cumplimiento riguroso con las regulaciones para proteger no solo la información sensible del paciente, sino también la sostenibilidad y el futuro de la startup en el competitivo mercado de la salud digital.

1.4. SEGURIDAD DE DATOS EN CLOUD SERVERLESS

La elección de AWS Cloud como proveedor de la infraestructura para la aplicación de telemedicina introduce consideraciones específicas en el modelo de responsabilidades de seguridad. AWS opera bajo el modelo de responsabilidad compartida, donde AWS se encarga de la seguridad “de” la nube, incluyendo la protección de la infraestructura que ejecuta los servicios de nube. Por otro lado, los clientes son responsables de la seguridad “en” la nube, que abarca aspectos como la gestión de identidades y accesos, la encriptación de datos, y la seguridad de las aplicaciones.

Además, la arquitectura serverless, orientada a eventos, trae consigo ventajas significativas en términos de escalabilidad y eficiencia operativa. Sin embargo, también plantea desafíos y consideraciones únicas de seguridad:

1.4.1. VENTAJAS DE LA ARQUITECTURA SERVERLESS

- **Escalabilidad Automática:** La capacidad de escalar automáticamente según la demanda sin necesidad de preaprovisionar recursos reduce el riesgo de sobrecarga de sistemas, lo cual es crucial para la disponibilidad y la integridad de la aplicación de telemedicina.
- **Mantenimiento Reducido:** Al eliminar la necesidad de gestionar servidores, los equipos pueden enfocarse en el desarrollo de la aplicación en sí, dejando en manos de AWS la tarea de asegurar la infraestructura, lo que incluye actualizaciones de seguridad y mantenimiento del sistema operativo.
- **Costo-Eficiencia:** La facturación basada en el uso real de recursos, en lugar de recursos preaprovisionados, puede resultar en ahorros significativos, especialmente para aplicaciones con patrones de tráfico variables.

1.4.2. CONSIDERACIONES DE SEGURIDAD EN ARQUITECTURA SERVERLESS

- **Gestión de Identidades y Accesos:** La gestión de permisos finamente granulados es crítica. AWS Lambda, un componente central en arquitecturas serverless, permite asignar roles específicos a cada función, restringiendo el acceso a los recursos mínimos necesarios para su ejecución.
- **Seguridad a Nivel de Aplicación:** A pesar de que AWS se encarga de la seguridad del sistema operativo y de la infraestructura subyacente, la seguridad a nivel de aplicación, incluyendo la gestión de vulnerabilidades en el código y la protección contra ataques a nivel de aplicación, recae en el desarrollador.
- **Automatización de la Seguridad:** La naturaleza efímera y escalable de las funciones serverless requiere un enfoque automatizado para la seguridad, incluyendo el escaneo automático de vulnerabilidades y la respuesta automática a incidentes de seguridad.

1.4.3. MINIMIZACIÓN DE RESPONSABILIDADES SOBRE CAPAS DE SEGURIDAD

La adopción de la arquitectura serverless en AWS representa un cambio estratégico para las startups en el sector de la telemedicina. Este modelo facilita a las empresas emergentes centrarse en la innovación y el desarrollo ágil de aplicaciones sin la necesidad de administrar directamente la infraestructura subyacente. En el modelo serverless, AWS se responsabiliza de la seguridad “de” la nube, asegurando la infraestructura global, los servicios de red y la configuración de la plataforma.

Para startups con recursos limitados y equipos pequeños, esto significa que pueden delegar una parte sustancial de las tareas de seguridad a AWS. Sin embargo, el modelo serverless no exime a la empresa de todas las responsabilidades de seguridad. La protección de la información del paciente y la gestión de la configuración de la aplicación son aspectos críticos que la startup debe manejar.

La Figura 3 demuestra que la seguridad “en” la nube es una tarea colaborativa. La startup debe implementar una gestión de identidades y accesos adecuada, cifrar los datos y asegurar la integridad de la información, además de monitorear y proteger las aplicaciones. Esto incluye controles de acceso rigurosos, cifrado de datos sensibles y un monitoreo constante para detectar y mitigar amenazas de seguridad. Con Serverless, responsabilidades como el sistema operativo, la configuración de redes, la gestión de la plataforma, la encriptación del código y la configuración del firewall, son manejadas por AWS, aliviando la carga que representarían en una arquitectura basada en contenedores o máquinas virtuales, donde tales tareas recaerían en el cliente.

La arquitectura serverless, por tanto, permite a las startups enfocarse en el desarrollo y la seguridad a nivel de aplicación, mientras que AWS se ocupa de la seguridad de la infraestructura. Es esencial para las startups comprender y actuar de acuerdo con este modelo de

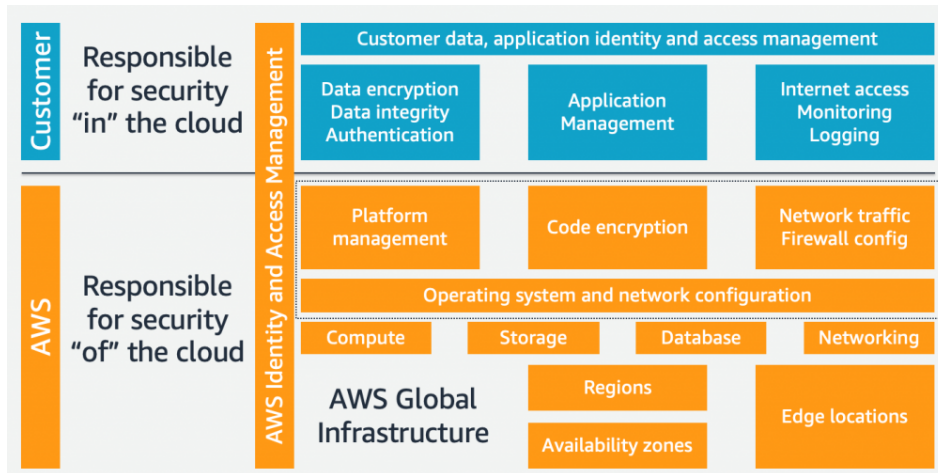


Figura 3: Modelo de responsabilidad compartida en arquitectura serverless.
Fuente: Amazon Web Services.

responsabilidad compartida para proteger eficazmente la información sensible de los pacientes y fortalecer su posición en el mercado de la salud digital.

1.5. ENFRENTANDO LOS DESAFÍOS DE SEGURIDAD EN LA TELEMEDICINA

La aplicación de telemedicina, como se ha detallado anteriormente, enfrenta el reto fundamental de garantizar la seguridad y privacidad de la información de salud del paciente en un entorno digital cada vez más vulnerado por amenazas cibernéticas. Dado el valor sensible y privado de los datos médicos, es crucial adoptar medidas que no solo cumplan con los estándares internacionales, sino que también ofrezcan una protección robusta contra cualquier intento de vulneración de estos datos.

1.5.1. PRIORIZACIÓN DEL CIFRADO DE DATOS

El cifrado de datos se establece como una defensa técnica primordial contra la divulgación no autorizada de datos sensibles. Transformar la información en un formato cifrado, inaccesible sin las claves de descifrado apropiadas, garantiza que, incluso frente a una violación de seguridad, los datos capturados o accedidos ilícitamente permanezcan seguros y sin valor para los atacantes. Esta práctica es fundamental tanto para la protección de datos en reposo como en tránsito, asegurando la seguridad en todos los puntos del ciclo de vida de los datos, desde su creación, pasando por su almacenamiento, hasta su transmisión.

Adoptar el cifrado desde el inicio del diseño y desarrollo de aplicaciones de telemedicina, siguiendo el principio de "seguridad por diseño", asegura que las medidas de protección sean inherentes y no añadidas como una reflexión tardía. Esta consideración es vital en entornos

de nube y arquitecturas sin servidor, donde la gestión de datos se extiende a través de múltiples servicios en una infraestructura compartida y distribuida, ampliando potencialmente la superficie de ataque y subrayando la importancia de un enfoque de seguridad en capas.

Por otro lado, la creciente sofisticación de los ataques cibernéticos, particularmente aquellos dirigidos al sector salud, hace del cifrado una medida de seguridad indispensable. Frente a ataques de ransomware, que buscan cifrar datos críticos para exigir un rescate, el cifrado propio de la organización actúa como un escudo defensivo esencial. Esto permite que, incluso después de un ataque, los datos protegidos por las claves de cifrado de la organización permanezcan seguros y accesibles, minimizando el impacto operacional y asegurando la continuidad del servicio.

En el contexto del almacenamiento de datos en soluciones cloud, el cifrado adquiere una dimensión adicional de relevancia. Al almacenar datos en infraestructuras gestionadas por proveedores de servicios en la nube, se presenta el desafío de proteger la información no solo en el ámbito digital, sino también frente a compromisos físicos potenciales de los servidores. El cifrado asegura que, incluso si un servidor es comprometido físicamente, los datos almacenados permanezcan inaccesibles y seguros, reforzando la protección contra accesos no autorizados y divulgación de información sensible.

1.5.2. DESARROLLO SOSTENIBLE Y SEGURIDAD A LARGO PLAZO

La adopción de tecnologías de cifrado avanzadas, como el cifrado homomórfico, que permite operar en datos cifrados sin necesidad de descifrarlos, se alinea con la visión de ofrecer servicios de telemedicina seguros y privados. Sin embargo, el desarrollo e integración de tales tecnologías deben planificarse cuidadosamente, considerando los recursos financieros y temporales limitados de la startup. La estrategia debe incluir fases de implementación escalonadas, comenzando con soluciones de cifrado más tradicionales y evolucionando hacia enfoques más avanzados a medida que la plataforma madura y se dispone de más recursos.

Además, es vital encontrar un equilibrio adecuado entre la seguridad de los datos y la usabilidad de la aplicación. Un sistema extremadamente seguro pero difícil de usar puede resultar en una baja adopción por parte de los usuarios finales, mientras que un sistema fácil de usar pero inseguro pone en riesgo la información crítica del paciente. Este equilibrio se logra a través de un diseño centrado en el usuario, que integre prácticas de seguridad sin comprometer la experiencia del usuario.

1.6. OBJETIVOS DEL TRABAJO

Dada la importancia de una gestión eficiente y segura del cifrado de datos en entornos serverless, este trabajo se centrará en garantizar la seguridad en la persistencia y manejo de

datos sensibles en la aplicación de telemedicina. El objetivo principal es prevenir ataques de exfiltración de datos, un desafío crítico en el contexto de las arquitecturas serverless donde la gestión de la seguridad de datos es fundamentalmente compleja.

El alcance de este estudio, por lo tanto, estará confinado al marco de los siguientes objetivos:

1.6.1. OBJETIVO GENERAL

Garantizar la seguridad de datos personales de salud en una aplicación de telemedicina frente a ataques de exfiltración de datos.

1.6.2. OBJETIVOS ESPECÍFICOS

- Analizar la arquitectura de la aplicación de telemedicina para identificar el esquema actual de utilización de datos.
- Crear un mecanismo resiliente a las filtraciones de datos de un almacenamiento persistente o volátil basados en mecanismos o algoritmos de cifrado, junto a una gestión confiable de claves de cifrado simétrico.
- Implementar un mecanismo de gestión segura de datos de salud que tenga el menor impacto posible en la arquitectura ya existente en la aplicación.

Si bien la seguridad en la telemedicina incluye aspectos más amplios como la autenticación de usuarios y los protocolos de transferencia de información, este estudio se focalizará específicamente en la gestión de cifrado en entornos serverless. Otros aspectos relevantes de la seguridad en la telemedicina, como la seguridad de los dispositivos utilizados para recopilar datos de salud o las implicaciones de seguridad en la integración con otros sistemas, quedan fuera del alcance de este trabajo, abriendo oportunidades para futuras investigaciones.

CAPÍTULO 2

MARCO CONCEPTUAL

2.1. TELESALUD Y TELEMEDICINA

Como bien refleja el Programa Nacional de Telesalud publicado en 2015 por el Ministerio de Salud de Chile [Ministerio de Salud de Chile, 2015], en el mundo existen cientos de definiciones de lo que es la telemedicina. Si bien en muchos países este término se considera un sinónimo de telesalud, existen diferencias significativas a destacar.

La telesalud corresponde al sistema que utiliza las Tecnologías de la Información y Comunicación (TIC) para proporcionar servicios de salud, atención médica e información, independiente de la distancia entre los involucrados.

En su definición más amplia, la telesalud incluye una gama completa de actividades que ayudan al paciente y a la población en general a estar sano: prevención, promoción, diagnóstico, autoatención y tratamiento.

Por su parte, la telemedicina es una de las ramas más destacadas, conocidas y usadas de la telesalud, pues de acuerdo con la Asociación Estadounidense de Telemedicina (*American Telemedicine Association, ATA*),

La telemedicina es el intercambio de información médica entre dos sitios a través de comunicaciones electrónicas con el objeto de mejorar el estado de salud de un paciente. Incluye una variedad cada vez mayor de aplicaciones y servicios que utilizan videoconferencias, correo electrónico, teléfonos inteligentes, comunicaciones inalámbricas y otras formas de tecnología de las telecomunicaciones.

De esta forma, la telemedicina es el intercambio de información médica a través de algún medio de comunicación electrónico entre dos actores que no están ubicados en el mismo espacio, y que pueden ser médico/paciente o médico/médico. La telesalud, por su parte, es el sistema de salud como tal, haciendo uso de las TIC, con especial énfasis en el carácter preventivo.

2.2. SEGURIDAD DE LA INFORMACIÓN EN SALUD

La seguridad de los sistemas de información clínica descansa en la confidencialidad, integridad y disponibilidad. Tan solo en 2023 se notificaron más de 133 millones de registros de salud expuestos en Estados Unidos, con el *phishing* y el *ransomware* como vectores predominantes [HIPAA Journal, 2024]. Estos riesgos se agravan en la nube, donde la externalización de la infraestructura introduce nuevas superficies de ataque [Rana et al., 2017].

Para reducir la probabilidad y el impacto de incidentes la normativa convergen en tres líneas de defensa principales:

- **Cifrado extremo a extremo** para impedir el acceso a datos sin la clave adecuada.
- **Gestión de identidades y acceso** basada en privilegios mínimos y autenticación multifactor, conforme a las guías de NIST [Marron, 2022].
- **Monitoreo y registro continuo:** auditoría de eventos críticos y respuesta automática a anomalías.

2.3. CLOUD COMPUTING

El concepto de *cloud computing* se remonta a la década de 1960, cuando John McCarthy predijo que “la computación podría algún día organizarse como una utilidad pública”.

Aunque existen discusiones sobre el significado exacto y la escala de la computación en la nube, Ian Foster *et al.* definen la nube como “un paradigma de computación que es un conjunto de plataformas y servicios de almacenamiento, virtualizados, dinámicamente escalables, administrados y entregados a través de Internet bajo demanda” [Foster *et al.*, 2009].

El cómputo en la nube está dando lugar a un cambio revolucionario en los modelos comerciales. Su objetivo es proporcionar a los usuarios servicios más flexibles de manera transparente. En particular, la computación en la nube brinda servicio a pedido, elasticidad, amplio acceso a la red y agrupación de recursos.

Además, la computación en la nube proporciona una solución flexible para hacer frente a las solicitudes de los usuarios en diferentes contextos.

El término “todo como servicio” (XaaS) ha entrado recientemente en nuestro vocabulario [Schaffer, 2009], donde X es una serie de servicios prestados a través de Internet.

El software como servicio (SaaS), la plataforma como servicio (PaaS) y la infraestructura como servicio (IaaS) son los modelos de servicio más conocidos y aceptados (ver Figura 4), pero otros ejemplos incluyen la comunicación como servicio (CaaS), la base de datos como un servicio (DBaaS), seguridad como servicio (SECaaS), gestión de identidad como servicio (IMaaS) y escritorio como servicio (DaaS).

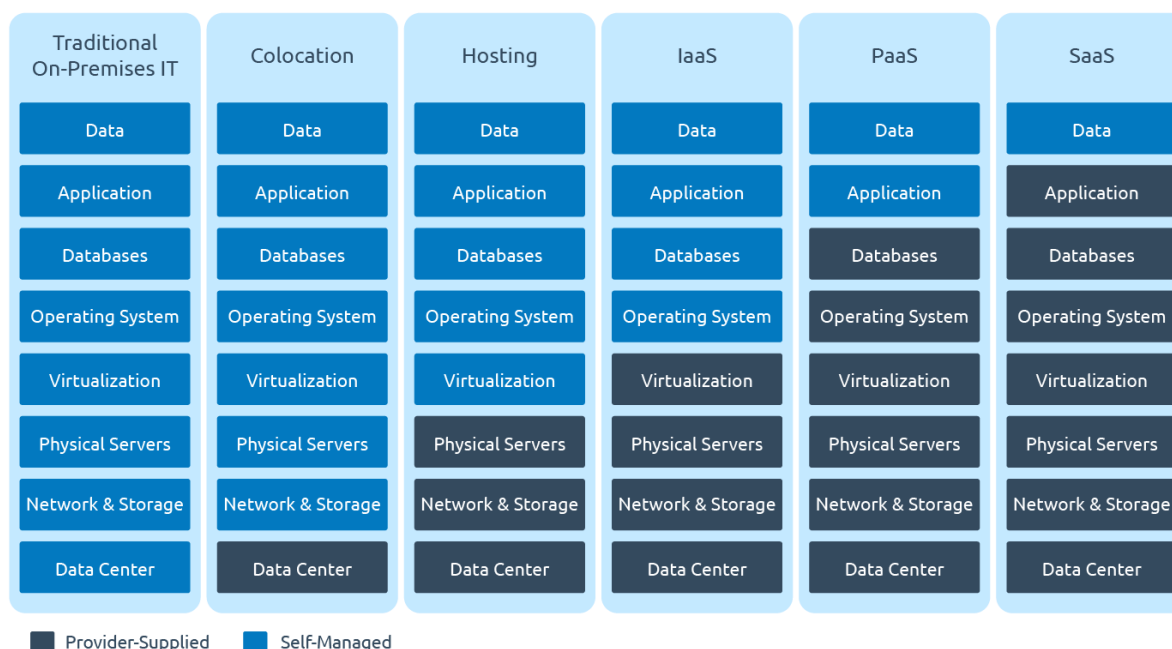


Figura 4: Diferencias entre IaaS, SaaS, PaaS y XaaS.

Fuente: Vansh Varshney. What is IaaS vs SaaS vs PaaS and XaaS: what's the difference. 2020.

2.4. AMENAZAS DE SEGURIDAD EN TELEMEDICINA

En el artículo de Kim *et al.* [Dong-won Kim y hee Han, 2020] basándose en el **ISO/IEC 27005** para la construcción de un diagrama de árbol de ataques, se señalan siete posibles amenazas de seguridad para un sistema de telemedicina (ver Figura 5):

1. Usuario o paciente.
2. Dispositivos de telemedicina.
3. Red del hogar.
4. Dispositivos de puerta de enlace.
5. Internet (red pública).
6. Sistema de telemedicina.
7. Proveedor de servicio de telemedicina.

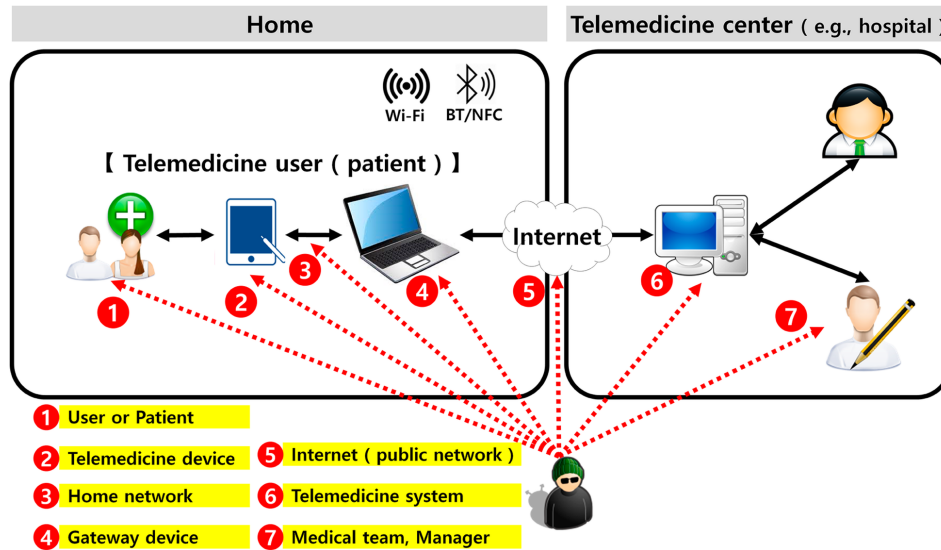


Figura 5: Siete áreas relacionadas con las amenazas a la seguridad de la telemedicina
Fuente: Dong-won Kim, Jin-young Choi and Keun-hee Han. Risk management-based security evaluation model for telemedicine systems. 2020.

Para el caso de la aplicación de la empresa en estudio y el alcance de nuestra propuesta de mejora, nos enfocaremos en tres áreas de posibles amenazas de seguridad:

1. **Dispositivo de puerta de enlace:** En este caso sería el dispositivo desde el que se accedería a la aplicación web de telemedicina.
2. **Internet:** Por donde transitarán los datos de los exámenes realizados, pudiendo ser interceptados por un atacante.
3. **Sistema de telemedicina:** Se encargará de procesar y almacenar los datos del examen realizado.

2.5. MODELO DE RESPONSABILIDAD COMPARTIDA EN ARQUITECTURAS SERVERLESS

En el marco del desarrollo de aplicaciones de telemedicina en la nube, es imperativo comprender el modelo de responsabilidad compartida, el cual establece las obligaciones tanto del proveedor de servicios en la nube como del cliente en materia de seguridad. Este modelo es particularmente pertinente en el contexto serverless, donde la gestión de la infraestructura se abstrae aún más que en los modelos de servicio en la nube tradicionales.

Seguridad “de” la nube: La responsabilidad del proveedor de servicios en la nube, como AWS o Microsoft Azure, engloba la seguridad de la infraestructura cloud que aloja los servi-

cios. Esto implica la protección y el mantenimiento del hardware, el software, y las redes, así como la gestión del sistema operativo y la capa de virtualización. Además, es esencial que el proveedor realice auditorías regulares mediante entidades independientes para validar la robustez de sus medidas de seguridad.

Seguridad “en” la nube: Por otro lado, la responsabilidad del cliente se centra en la seguridad de los componentes que opera dentro de la nube. Esto incluye la gestión de los datos, las aplicaciones, y la configuración de la seguridad, tales como firewalls y políticas de acceso. Los clientes tienen la tarea de configurar los servicios de la nube de acuerdo con sus propias necesidades de seguridad, privacidad y regulaciones aplicables.

La figura 3 ilustra cómo en un entorno serverless, el proveedor de servicios en la nube asume un rol más amplio en la gestión de la infraestructura, lo que permite a los clientes centrarse en el desarrollo de la lógica de la aplicación y en la gobernanza de los datos. Este modelo es vital para garantizar la seguridad y la conformidad en el ámbito de la telemedicina, donde los datos sensibles de salud deben ser manejados con el más alto nivel de integridad y confidencialidad.

En la implementación de servicios serverless, es crucial que los clientes comprendan y actúen de acuerdo con sus responsabilidades, utilizando las herramientas y servicios proporcionados por el proveedor para asegurar sus aplicaciones y datos. De esta manera, se puede aprovechar la flexibilidad y eficiencia de la nube, al tiempo que se mantiene un estándar elevado de seguridad y conformidad.

2.6. RETOS DE LA SEGURIDAD PARA TELEMEDICINA EN CLOUD

En el artículo de Zhanpeng Jin y Yu Chen [Jin y Chen, 2015], se presentan algunos retos de la telemedicina en *Cloud*, desde la interoperabilidad de los datos hasta la seguridad de los sistemas, siendo esta identificada como la principal preocupación cuando se contratan servicios en la nube.

A continuación, se listan algunos desafíos en el ámbito de seguridad de los proveedores de los servicios de *cloud computing*:

- Disponibilidad del servicio.
- Aseguramiento de la información.
- Vulnerabilidad de control de acceso.
- Ataques de canal lateral.
- Ataques de escape al hipervisor.

2.7. PROTECCIÓN DE DATOS Y TECNOLOGÍAS DE CIFRADO

La protección de datos y el cifrado son esenciales para garantizar la confidencialidad y seguridad de la información, especialmente en aplicaciones críticas como la telemedicina.

Las tecnologías de cifrado han evolucionado en años recientes y su implementación adecuada es crucial para prevenir accesos no autorizados y garantizar la integridad de los datos.

2.7.1. ALGORITMOS DE CIFRADO

El cifrar o encriptar datos consiste en intencionalmente volverlos ilegibles, manteniendo la posibilidad de que solo quien conozca la clave de cifrado pueda devolverlos a su estado inicial.

Como objetivo principal de estos mecanismos se encuentra el mantener la confidencialidad de la información, ya sea que esta se encuentre en tránsito o almacenada de forma estática.

La idea es que si un atacante logra interceptar o acceder a datos encriptados, mientras no tenga la clave para descifrarlos no podrá hacer nada con ellos.

Existen dos grandes tipos de algoritmos de cifrado: simétrico y asimétrico, como se puede observar en la Figura 6, la principal diferencia es la clave de cifrado utilizada y en consecuencia el rendimiento de cada uno de estos.

Para esta propuesta de mejora, y debido a las particularidades del problema y la situación actual de la aplicación de la empresa, nos enfocaremos específicamente en el cifrado simétrico.

2.7.2. CIFRADO SIMÉTRICO

El cifrado simétrico es el método más común para garantizar la confidencialidad de un mensaje o información.

En este proceso criptográfico es imposible para cualquier persona que no tenga la clave de descifrado obtener conocimiento del texto plano.

Se denomina simétrico cuando utiliza la misma clave para cifrar y descifrar. La longitud de clave requerida para el cifrado simétrico debe ser de al menos 256 bits para garantizar una protección suficiente contra ataques de fuerza bruta y la posible llegada de computadores cuánticos.

Es fundamental utilizar bibliotecas criptográficas bien establecidas que implementen algo-

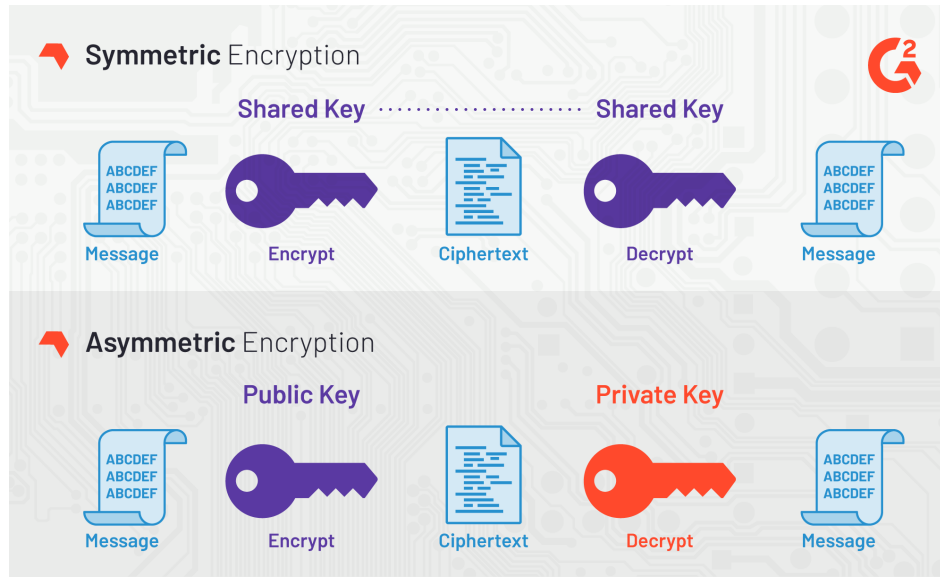


Figura 6: Comparación entre cifrado simétrico y asimétrico.
Fuente: Lane Wagner. Intro to The AES-256 Cipher. 2020.

ritmos estándar en lugar de implementaciones caseras. [Weissbaum y Lugrin, 2023]

2.7.3. GESTIÓN DE CLAVES DE CIFRADO

La gestión de claves describe cómo se crean, almacenan de forma segura, distribuyen a los respectivos titulares de las claves y se utilizan de acuerdo con las especificaciones del protocolo las claves criptográficas.

La gestión de claves de cifrado es una piedra angular de la mayoría de los sistemas criptográficos y debe ser manejada con cuidado. Los avances en módulos de seguridad de *hardware* utilizados en el almacenamiento de claves y generadores de números aleatorios de alto y bajo costo utilizados en la generación de claves muestran un futuro prometedor para una gestión de claves segura y asequible. Sin embargo, desafíos futuros como, por ejemplo, la resistencia cuántica, deben ser superados por nuevos sistemas de gestión de claves. [Krähenbühl y Perrig, 2023a]

2.7.4. HARDWARE SECURITY MODULE (HSM)

Los módulos de seguridad de *hardware* (o HSM por sus siglas en inglés) son dispositivos especializados que realizan operaciones criptográficas. Sus funciones incluyen la generación y gestión de claves, el cifrado, el descifrado y el *hashing*.

Con la llegada de la computación en la nube, la complejidad de asegurar datos críticos ha aumentado. Como resultado, el cifrado de doble clave se ha vuelto cada vez más popular, que cifra los datos utilizando dos claves. Una copia se almacena en un HSM y otra copia se almacena en la nube.

Además, dado que los módulos de seguridad de *hardware* pueden gestionar claves y permitir a los usuarios gestionar claves, ofrecen importantes beneficios de seguridad a las aplicaciones que utilizan criptografía [Sommerhalder, 2023].

2.8. REGULACIÓN INTERNACIONAL EN PROTECCIÓN DE DATOS

El tratamiento de información clínica está fuertemente regulado a nivel mundial. Para los proveedores de telemedicina, cumplir con los marcos internacionales más exigentes es esencial a fin de garantizar la privacidad de los pacientes, habilitar la interoperabilidad global de los servicios y evitar sanciones económicas o daños reputacionales [World Health Organization, 2019]. A continuación se describen los referentes normativos y estándares más importantes y su impacto en sistemas de salud digitales.

2.8.1. HIPAA (HEALTH INSURANCE PORTABILITY AND ACCOUNTABILITY ACT)

Promulgada en 1996 en los Estados Unidos, la HIPAA regula el uso y divulgación de *Protected Health Information* (PHI) y exige salvaguardas administrativas, físicas y técnicas para protegerla [HIPAA Journal, 2023b].

Reglas principales:

- **Regla de Privacidad y Regla de Seguridad:** delimitan quién, cómo y para qué puede acceder o divulgar PHI.
- **Salvaguardas técnicas:** cifrado, control de acceso basado en rol, autenticación multifactor, registros de auditoría y planes de contingencia [Marron, 2022].
- **Notificación de incidentes:** obligación de informar a los pacientes y al *Department of Health & Human Services* cuando ocurre una brecha de seguridad.

Régimen sancionatorio: Las multas administrativas oscilan entre \$100 USD y \$50 000 USD por violación, pudiendo alcanzar hasta \$1,5 millones USD anuales en casos de negligencia grave o reiterada [HIPAA Journal, 2023b]. Hay, además, responsabilidad penal en violaciones dolosas.

2.8.2. GDPR (GENERAL DATA PROTECTION REGULATION)

El GDPR, vigente desde 2018, regula todo tratamiento de datos personales de residentes de la Unión Europea, incluso si el responsable se encuentra fuera de la UE [Commission, 2018].

Principios fundamentales: Lícitud, transparencia, minimización, exactitud, limitación de la conservación, integridad/confidencialidad y responsabilidad proactiva (*accountability*).

Derechos del interesado: Acceso, rectificación, supresión (*derecho al olvido*), portabilidad, oposición y limitación del tratamiento, así como el derecho a no ser objeto de decisiones únicamente automatizadas.

Datos de salud como categoría especial: Su tratamiento requiere consentimiento *explícito* o base legal médica/pública. Debe realizarse una Evaluación de Impacto cuando existan riesgos altos para las libertades del paciente.

Sanciones: Hasta 20 millones EUR o el 4 % de la facturación global anual de la organización, lo que sea mayor [Commission, 2018].

2.8.3. OTROS ESTÁNDARES Y DIRECTRICES INTERNACIONALES

- **ISO/IEC 27799:2016** — guía la aplicación de ISO/IEC 27002 en organizaciones de salud, definiendo controles específicos para garantizar la confidencialidad, integridad y disponibilidad de los datos clínicos [ISO, 2016].
- **ISO/IEC 27701:2019** — amplía ISO/IEC 27001/27002 con un sistema de gestión de *Privacy Information*; permite demostrar *accountability* frente a GDPR y normas equivalentes [ISO, 2019].
- **OMS (2019)** — la Organización Mundial de la Salud recomienda incorporar *privacy by design*, transparencia y consentimiento informado en las intervenciones digitales para el fortalecimiento de los sistemas de salud [World Health Organization, 2019].

Estas normas proporcionan marcos de referencia técnicos y de gobierno que complementan las obligaciones legales. Su adopción facilita auditorías de terceros, acuerdos de interoperabilidad transfronteriza y procesos de certificación.

2.8.4. COMPARACIÓN HIPAA VS GDPR Y BUENAS PRÁCTICAS PARA TELEMEDICINA

- **Alcance territorial:** HIPAA se aplica a entidades cubiertas y *business associates* en EE. UU., mientras que el GDPR tiene efecto extraterritorial, afectando a cualquier proveedor que trate datos de residentes de la UE.

- **Enfoque en privacidad:** HIPAA se centra en la información de salud; el GDPR cubre todo dato personal, con mayores derechos para el titular y principios de minimización.
- **Sanciones máximas:** HIPAA: \$1,5 M USD anual por norma; GDPR: 20 M EUR o 4 % de ingresos globales.
- **Requisitos comunes:** cifrado extremo a extremo, autenticación multifactor, registro de eventos, políticas de retención de datos, contratos con procesadores y notificación de brechas en plazos breves.

Para plataformas de teleconsulta, cumplir con ambos marcos exige:

1. Consentimientos explícitos y registrables para uso de datos sensibles.
2. Diseño de la aplicación siguiendo principios de *privacy by design*.
3. Evaluaciones de Impacto (GDPR) o Análisis de Riesgo (HIPAA) antes de lanzar nuevos servicios.
4. Certificación o alineación con ISO/IEC 27799 y 27701 para demostrar madurez en gestión de seguridad y privacidad.

El alineamiento con estos estándares internacionales no sólo mitiga sanciones: eleva la confianza de pacientes y socios clínicos y facilita la expansión de la solución a múltiples jurisdicciones.

2.9. REGULACIÓN CHILENA DE PROTECCIÓN DE DATOS

El marco jurídico chileno en materia de datos personales ha evolucionado desde un esquema básico, centrado en la Ley 19.628 de 1999, hacia un régimen de estándares altos inspirados en el GDPR europeo con la promulgación de la Ley 21.663 (2024).¹ A continuación se describen los elementos más relevantes de ambas normas y su impacto concreto en la industria de la salud y la telemedicina.

2.9.1. LEY 19.628 SOBRE PROTECCIÓN DE LA VIDA PRIVADA (1999)

- **Principios fundacionales:** Introdujo los principios de *consentimiento*, *finalidad* y *responsabilidad* en el tratamiento de datos personales [Congreso Nacional de Chile, 1999].

¹Proyecto tramitado como Boletín 11144-07 en el Congreso Nacional.

- **Datos sensibles:** Declaró como “sensibles” los datos relativos a salud, vida sexual, filiación genética, orientación religiosa y otros ámbitos íntimos, exigiendo especial protección sobre ellos.
- **Derechos del titular:** Reconoció acceso y rectificación, pero carecía de portabilidad, oposición y cancelación explícita, lo que limitaba el control efectivo del individuo sobre su información.
- **Brechas de la norma:** No creó una autoridad de control especializada; la fiscalización quedó dispersa entre tribunales y, en el sector público, el Consejo para la Transparencia, generando aplicación fragmentaria.

2.9.2. LEY 21.663 QUE FORTALECE LA PROTECCIÓN DE DATOS PERSONALES (2024)

La nueva ley moderniza completamente el régimen chileno, alineándolo con los principios y exigencias del GDPR [Congreso Nacional de Chile, 2024].

Principios rectores: Lícitud, finalidad, proporcionalidad/minimización, exactitud, transparencia, seguridad e *integridad/confidencialidad*, y responsabilidad proactiva (*accountability*).

Derechos del titular: Acceso, rectificación, cancelación/supresión, oposición, *bloqueo/limitación*, portabilidad y derecho a no ser objeto de decisiones únicamente automatizadas.

Obligaciones del responsable:

- *Base legal* para todo tratamiento (consentimiento libre, específico e informado, ejecución contractual, obligación legal, interés legítimo, etc.).
- *Transparencia* y deber de información clara al recolectar datos.
- *Seguridad y privacidad desde el diseño:* cifrado, seudonimización, control de acceso y evaluación de impacto para tratamientos de alto riesgo.
- *Notificación de brechas:* reporte a la Agencia y a los titulares afectados.
- *Delegado de Protección de Datos (DPO):* fuertemente recomendado e indispensable para modelos de cumplimiento certificados.

Agencia de Protección de Datos Personales: Se crea un organismo autónomo con facultades de fiscalización, dictación de instrucciones, resolución de reclamos, certificación de modelos de cumplimiento y aplicación de sanciones.

Régimen sancionatorio: Multas de hasta **20 000 UTM** ($\approx$ US\$1,55 millones) o, alternatively, hasta el 4 % de los ingresos anuales del infractor en infracciones gravísimas; agravantes por reincidencia o falta de cooperación [Congreso Nacional de Chile, 2024].

2.9.3. COMPATIBILIDAD CON EL GDPR Y OTROS ESTÁNDARES

La Ley 21.663 replica casi íntegramente la lógica del GDPR (principios, derechos, brechas, extraterritorialidad y autoridad independiente), lo que facilita una futura decisión de *ade-cuación* europea y el flujo internacional de datos [Commission, 2018].

A su vez, la exigencia de notificar brechas y de medidas de seguridad se alinea con la HIPAA estadounidense en el ámbito sanitario [HIPAA Journal, 2023b].

2.9.4. IMPACTO EN EL SECTOR SALUD Y LA TELEMEDICINA

La información clínica constituye un tipo de dato personal sensible; en consecuencia, su tratamiento está sujeto a ciertas obligaciones que, en síntesis:

1. Requiere consentimiento expreso del paciente o base legal equivalente.
2. Obliga a cifrado fuerte (en tránsito y reposo), registro de accesos y control de privilegios mínimos.
3. Impone la realización de Evaluaciones de Impacto en Privacidad antes de nuevos tratamientos automatizados o a gran escala.
4. Obliga a notificar incidentes de seguridad a la autoridad competente y a los titulares afectados dentro de los plazos legales establecidos.

Para plataformas de telemedicina, esto implica integrar privacidad por diseño, autenticación multifactor, cifrado end-to-end y gobierno de datos robusto, mitigando a la vez los riesgos regulatorios y reputacionales del sector.

CAPÍTULO 3

PROPUESTA DE SOLUCIÓN

3.1. ANÁLISIS DE LA ARQUITECTURA ACTUAL

Para abordar el almacenamiento y tratamiento seguro de datos de salud en la aplicación de telemedicina, es fundamental comprender la arquitectura existente y sus puntos críticos. La aplicación se ha desarrollado siguiendo principios de arquitectura limpia (*Clean Architecture*) y *Domain-Driven Design* (DDD), con el objetivo de lograr un sistema modular, escalable y mantenible. A continuación se detalla la arquitectura actual en sus distintos niveles y se identifican las brechas de seguridad a atender.

3.1.1. ARQUITECTURA A NIVEL DE INFRAESTRUCTURA

El despliegue de la aplicación se realiza sobre servicios serverless de **Amazon Web Services (AWS)**, lo que permite escalar automáticamente y reduce la complejidad operativa. La Figura 7 ilustra la arquitectura de infraestructura, destacando los siguientes componentes principales:

- **AWS Lambda:** Ejecuta la lógica de negocio en funciones bajo demanda, garantizando aislamiento y eficiencia. Cada función Lambda corresponde a casos de uso específicos (por ejemplo, registro de paciente, almacenamiento de examen clínico, etc.), encapsulando la ejecución de procesos sin administrar servidores.
- **Amazon API Gateway:** Expone las API REST utilizadas por los clientes (aplicación frontend web/móvil) u otros servicios. Gestiona la entrada de solicitudes HTTP(S) y su enrutamiento hacia las funciones Lambda correspondientes, además de ofrecer autenticación, limitación de tasas y monitoreo.
- **Amazon DynamoDB:** Servidor de base de datos NoSQL utilizado para almacenar información estructurada de la aplicación (por ejemplo, datos de casos clínicos, perfiles de usuarios, historiales). Proporciona baja latencia y alta disponibilidad de forma automática. Por defecto, DynamoDB cifra los datos en reposo y, en este proyecto, se configuró para usar claves administradas por el cliente (KMS) para un control más granular.
- **Amazon S3:** Servicio de almacenamiento de objetos utilizado para guardar archivos grandes o binarios, como los videos capturados en sesiones de telemedicina (exámenes audiovisuales). S3 ofrece durabilidad elevada y, en nuestro caso, se configuró con cifrado del lado del servidor (*Server-Side Encryption*) utilizando claves KMS, de manera que cualquier objeto almacenado quede cifrado automáticamente a nivel de infraestructura.

- **Amazon Cognito:** Servicio gestionado de identidad utilizado para el registro, autenticación y autorización de usuarios (pacientes, médicos, administradores). Cognito maneja flujos de inicio de sesión seguros y emite *JSON Web Tokens (JWT)* para que el frontend acceda a las APIs de manera autenticada. Este componente reduce la carga de implementar autenticación manualmente y se integra con API Gateway para controlar el acceso según roles.

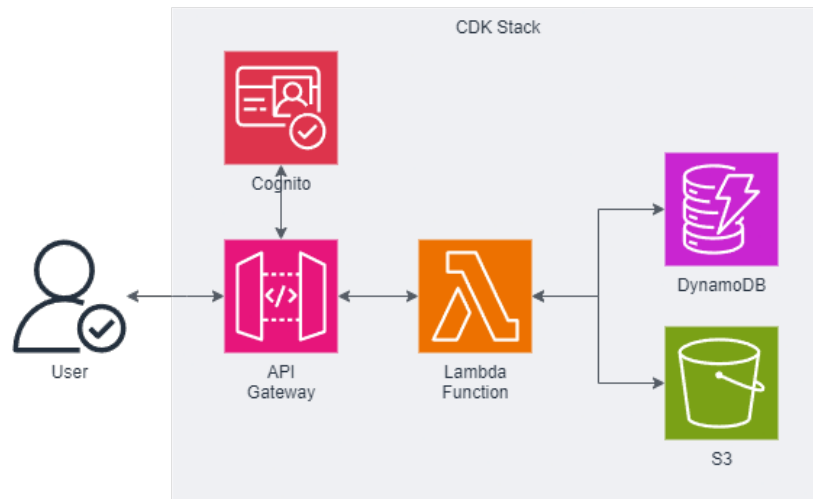


Figura 7: Arquitectura de la Infraestructura de la Aplicación de Telemedicina en Estudio
Fuente: Elaboración propia.

Toda la infraestructura se define como código utilizando **AWS Cloud Development Kit (CDK)** en TypeScript. Esto permite describir los recursos de AWS (Lambdas, API Gateway, tablas DynamoDB, buckets S3, etc.) mediante código, quedando la configuración bajo control de versión. Gracias a **Infrastructure as Code (IaC)** con CDK, se logra trazabilidad de cambios y se integran las provisiones de infraestructura en los procesos de CI/CD de la aplicación, garantizando despliegues consistentes y repetibles. Además, se aprovechan características avanzadas de CDK como los **Aspects para aplicar configuraciones transversales** (por ejemplo, forzar que todas las tablas DynamoDB tengan cifrado habilitado).

3.1.2. ARQUITECTURA A NIVEL DE CÓDIGO

La estructura interna del código de la aplicación sigue principios de arquitectura hexagonal y organización por dominios, alineados con DDD. Esto implica una separación clara de capas:

- **Dominio (Domain):** Contiene las reglas de negocio fundamentales y las entidades del sistema (por ejemplo, entidad Patient, ClinicalCase, etc.). Esta capa es independiente de detalles técnicos; modela el comportamiento y estados válidos de los datos de salud en la telemedicina.

- **Aplicación (Application):** Orquesta casos de uso y procesos de negocio usando el dominio. Aquí residen los use cases o servicios de aplicación, que coordinan las entidades de dominio y definen las transacciones o pasos para cumplir una funcionalidad. Esta capa tampoco conoce detalles de infraestructura; se apoya en abstracciones (interfaces) para acceder a recursos externos.
- **Infraestructura (Infrastructure):** Implementa las interfaces requeridas por la capa de aplicación, conectando con tecnologías externas. Incluye adaptadores para la persistencia (repositorios para DynamoDB, S3), servicios de terceros (como Cognito para autenticación) y cualquier componente externo. Por ejemplo, un `DynamoClinicalCaseRepository` en esta capa sabe cómo guardar y leer un caso clínico desde DynamoDB, o un `S3VideoStorageService` sabe cómo subir un video a S3.

Adicionalmente, se adoptaron prácticas de **Vertical Slicing** y **Screaming Architecture**: el código se organiza en módulos por contexto de negocio (bounded contexts). Cada módulo agrupa todo lo necesario para ese dominio específico de manera vertical (entidades, casos de uso, repositorios, etc.), en lugar de agrupar por tipo de capa globalmente. Por ejemplo, existe un módulo `examination/clinical-cases` que contiene subdirectorios `domain/`, `application/` e `infrastructure/` propios para las funcionalidades de casos clínicos. Este arreglo enfatiza la cohesión dentro de cada contexto (todo el código relevante a casos clínicos está junto) y limita las dependencias entre módulos, facilitando la escalabilidad y mantenibilidad.

Gracias a esta arquitectura limpia, cualquier nueva preocupación transversal (como agregar cifrado, auditoría, validaciones adicionales) idealmente se implementa en la capa de infraestructura o en servicios especializados sin modificar la lógica de dominio. Esto resulta crucial para nuestra estrategia de seguridad: podemos añadir mecanismos de cifrado o controles de acceso en los adaptadores de infraestructura (por ejemplo, en el repositorio de casos clínicos o en el servicio de almacenamiento de videos) sin alterar los casos de uso ni las entidades de negocio. De esta forma se mantiene el **principio de inversión de dependencias**: la lógica de negocio define interfaces criptográficas o de almacenamiento que implementaremos en la periferia, permitiendo evolucionar la seguridad sin afectar el núcleo de la aplicación.

Un ejemplo de la implementación de estos patrones de arquitectura y principios de diseño, se puede observar en la Figura 8, donde se representa un módulo compuesto por el agregado de los **casos clínicos**, específicamente del caso de uso de **inicialización de un caso clínico**.

3.1.3. PUNTOS CRÍTICOS DEL USO DE DATOS

En la aplicación de telemedicina se manejan distintos tipos de datos personales de salud a lo largo de su ciclo de vida. A continuación se enumeran los principales puntos de procesamiento de datos sensibles y por qué son críticos en términos de seguridad:

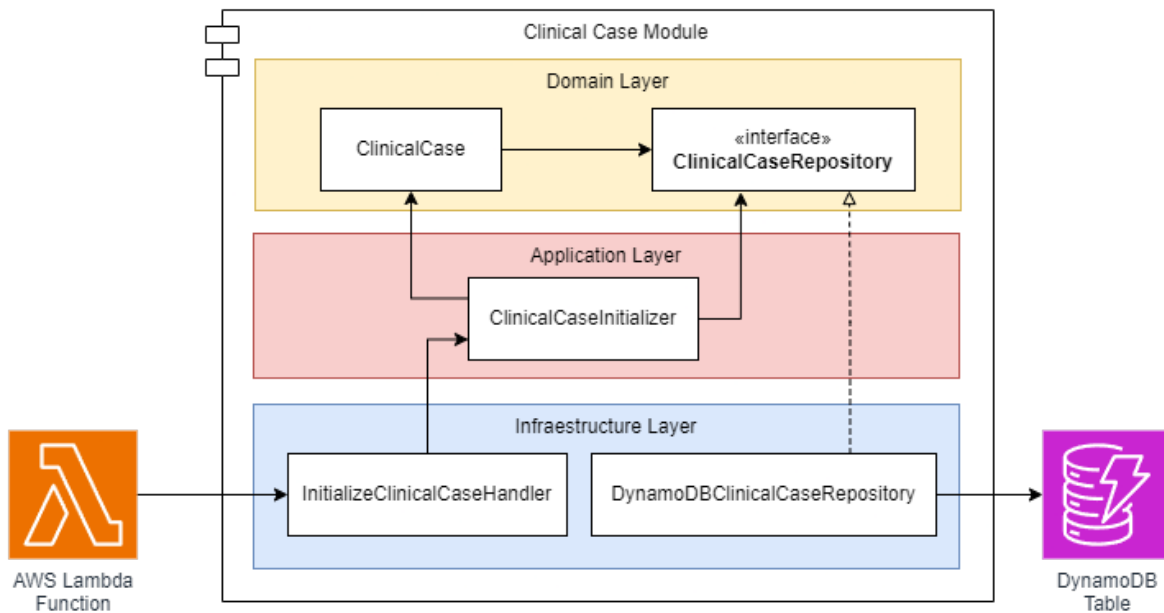


Figura 8: Arquitectura del código de la Aplicación de Telemedicina en Estudio
Fuente: Elaboración propia.

1. **Registro y Autenticación de Usuarios:** Durante el alta de nuevos usuarios (pacientes, médicos, administradores), se recopilan datos personales como nombre, correo electrónico, identificación, entre otros. Estos datos iniciales se almacenan (por ejemplo, en Cognito y DynamoDB) y son sensibles porque, de exponerse, podrían emplearse para suplantar identidades o realizar accesos no autorizados. La protección de esta información (mediante cifrado de campos sensibles y controles de acceso estrictos) es crítica desde la creación de la cuenta. Asimismo, el proceso de autenticación en sí, manejado por Amazon Cognito, emite tokens seguros; sin embargo, es vital asegurar que dichos tokens no se intercepten (cifrado en tránsito con TLS) y que su almacenamiento en el cliente sea seguro.
2. **Creación y Actualización de Casos Clínicos:** Al iniciar un caso clínico nuevo o actualizar uno existente, el personal médico ingresa datos identificatorios del paciente y detalles de su historia médica. Además, pueden adjuntar documentos (ej. informes de laboratorio, recetas, imágenes) que contienen información altamente confidencial. Estos datos viajan desde el front-end al backend a través de la API y luego son procesados por funciones Lambda que los persisten en DynamoDB (datos estructurados) o S3 (archivos adjuntos). Este flujo requiere cifrado en tránsito (TLS) para que la información no sea legible por terceros en la comunicación, y cifrado en reposo una vez almacenada. Cualquier brecha aquí podría exponer datos clínicos, por lo que es un punto central para aplicar nuestra solución de cifrado.
3. **Captura y Almacenamiento de Videos Médicos:** La aplicación permite grabar videos de exámenes (por ejemplo, exploración otorrinolaringológica) que luego se almace-

nan en Amazon S3. Estos videos pueden mostrar al paciente o revelar condiciones de salud, por lo que son datos personales sensibles. Además, su volumen es considerable (archivos de varios MB o GB) incrementando la superficie de riesgo. Es imperativo que cada video se almacene cifrado en S3 y que el acceso esté fuertemente controlado. Este escenario resalta la necesidad de cifrado a nivel de objeto en S3 y políticas estrictas de IAM, ya que una exfiltración de este contenido supondría una grave violación de privacidad.

4. **Procesamiento de Información en la Capa de Negocio:** La lógica de la aplicación en las funciones Lambda realiza validaciones y transformaciones sobre los datos clínicos. Si en esta capa no se siguen principios de seguridad (por ejemplo, sanitización de entradas, remoción o enmascaramiento de datos sensibles en logs), podría ocurrir exposición inadvertida de información. Además, cuando la función Lambda prepara datos para almacenamiento, especialmente en DynamoDB, necesita asegurarse de que campos sensibles (diagnósticos, resultados, notas) no queden en texto claro visible para desarrolladores con acceso a la base de datos o ante una posible filtración. La persistencia en DynamoDB de datos estructurados de salud demanda cifrado (sea a nivel de campo o global de la tabla) y controles de acceso minuciosos para evitar lecturas no autorizadas.
5. **Acceso a Resultados y Revisión por Especialistas:** La plataforma permite que distintos especialistas accedan a los casos clínicos, incluyendo la descarga de documentos adjuntos o reproducción de videos. Esto implica que el sistema genera URLs firmadas (*pre-signed URLs*) de S3 de corta duración o provee los datos a través de la API, solo a quienes tengan permisos. Cada acceso de terceros (aunque sean especialistas dentro de la misma red de salud) es muy sensible, ya que se comparte información médica. Por ello, se requiere trazabilidad (auditoría de quién accede a qué) y garantías de que incluso dentro del sistema nadie accede a datos sin autorización explícita. Desde la perspectiva normativa, este punto exige cumplimiento de regulaciones como **HIPAA** (EE.UU.) y **GDPR** (Europa), que obligan a mínimos privilegios y a informar/limitar el acceso a información de salud.
6. **Logs y Auditoría del Sistema:** El sistema genera registros (logs) de las operaciones realizadas, errores, accesos de usuarios, etc., en servicios como Amazon CloudWatch Logs y AWS CloudTrail. Estos logs, si bien son necesarios para depuración y auditoría, pueden eventualmente contener fragmentos de datos personales (por ejemplo, un ID de paciente, o un mensaje de error con algún dato). Es crucial que los mecanismos de logging anonimicen o enmascaren datos sensibles y que los propios archivos de log tengan protección (cifrado en reposo, controles de acceso y políticas de retención limitadas). De no hacerlo, un atacante con acceso a los logs podría extraer información confidencial. Este es un punto frecuentemente olvidado, pero fundamental para el cumplimiento normativo: normas como HIPAA exigen mantener registros de acceso, pero también protegerlos adecuadamente.

En resumen, el análisis de la arquitectura actual revela que **el cifrado de datos sensibles es**

un requerimiento transversal en múltiples puntos: desde la entrada de datos por el usuario, pasando por su almacenamiento y posterior acceso, hasta en los registros de auditoría. Si bien AWS proporciona una base segura (por ejemplo, Cognito para autenticación, TLS en API Gateway, cifrado predeterminado en algunos servicios), corresponde al equipo de desarrollo implementar capas adicionales de seguridad específicas de la aplicación. Identificadas estas brechas y puntos críticos, pasamos a proponer la solución de seguridad que aborde principalmente la confidencialidad de los datos en reposo y en tránsito, manteniendo la integridad de la arquitectura existente y cumpliendo con las normativas aplicables.

3.2. DISEÑO DE LA SOLUCIÓN

En esta sección se plantea la propuesta de solución para garantizar la seguridad de los datos de salud frente a amenazas de exfiltración y accesos no autorizados. La solución propuesta se alinea con los requerimientos normativos y con la arquitectura existente descrita, evitando alteraciones disruptivas en el sistema actual. En particular, el diseño pone énfasis en los siguientes aspectos fundamentales:

- **Cifrado de Datos en Reposo y en Tránsito:** Se garantiza que los datos sensibles estén cifrados tanto cuando se almacenan (en bases de datos, almacenamiento de archivos) como cuando se transmiten por la red. Para los datos en reposo, se aprovechan técnicas de cifrado a nivel de infraestructura proporcionadas por AWS (por ejemplo, cifrado del lado del servidor con claves KMS en S3 y DynamoDB), siguiendo el concepto de envelope encryption pero aplicado de forma transparente en la nube. Para los datos en tránsito, se apoya en TLS (HTTPS) *end-to-end* entre clientes, API Gateway y funciones Lambda.
- **Gestión Robusta de Claves Criptográficas:** Se utilizan claves administradas de forma centralizada en AWS Key Management Service (KMS) para todas las operaciones de cifrado. Se configura la rotación periódica de las claves maestras y se restringen los accesos a las mismas mediante políticas de IAM y separando roles (administrador de claves vs. servicios que usan las claves). Opcionalmente, se contempla la posibilidad de utilizar HSMs dedicados (como AWS CloudHSM) si a futuro se requieren garantías adicionales.
- **Seguridad y Privacidad desde el Diseño:** Cada nueva funcionalidad y componente se diseñará considerando principios de *Security by Design* y *Privacy by Design*. Esto implica aplicar mínimos privilegios en acceso a datos, aislar los módulos de seguridad en la arquitectura (manteniendo la lógica de negocio desacoplada de detalles de cifrado), y prever controles de auditoría y manejo de datos personales (como anonimización o eliminación segura) desde la concepción del sistema. (*security by design y privacy by design*).

- **Integración Transparente con la Arquitectura Serverless:** La solución aprovecha los servicios serverless de AWS de forma nativa, integrándose sin fricción con la infraestructura existente. Se evitarán soluciones que requieran servidores dedicados u otros componentes que rompan el paradigma serverless. Asimismo, el diseño sigue los lineamientos de la arquitectura limpia y DDD: cualquier mecanismo de seguridad (como el cifrado) se implementará en la periferia (capa de infraestructura), respetando las abstracciones existentes, de modo que la capa de dominio y aplicación permanezcan prácticamente inalteradas.

A continuación, se detallan los componentes del diseño de la solución, incluyendo la estrategia de cifrado de datos propuesta, las decisiones sobre gestión de claves, integración con la arquitectura actual y consideraciones normativas y de desempeño.

3.2.1. ESTRATEGIA DE CIFRADO DE DATOS

El enfoque general para proteger los datos sensibles, tanto estructurados como no estructurados (e.g., registros en DynamoDB y archivos en S3), será el uso de **cifrado en sobre** (*envelope encryption*). Como podemos seguir en la Figura 9, esta estrategia consiste en:

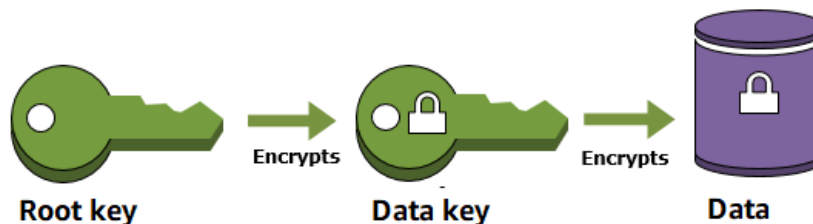


Figura 9: Flujo del cifrado en sobre
Fuente: De KAK Labs (2024).

1. Generar y/o recuperar una *Data Key* (clave de datos simétrica) para cifrar el contenido específico (videos, documentos, datos del paciente, etc.).
2. Cifrar la *Data Key* con una *Customer Master Key* (CMK) administrada en AWS KMS.
3. Almacenar la *Data Key* cifrada junto con los datos (si corresponde) o en un repositorio seguro, permitiendo su descifrado únicamente a través de la CMK.

Dado que el modelo de aplicación se basa en arquitecturas *serverless*, la lógica de cifrado/descifrado puede residir en las funciones Lambda correspondientes. De esta manera, se evita exponer las claves en cada servicio y se minimiza el riesgo de fugas de información.

Selección de Algoritmos de Cifrado

- **AES-256-GCM** (Galois/Counter Mode) para cifrado simétrico de los datos en reposo, por su alto rendimiento y nivel de seguridad.
- **RSA-2048 o ECC** (incluido en AWS KMS) para proteger la *Data Key*.

Estos algoritmos brindan protección robusta ante intentos de exfiltración y se alinean con normas como HIPAA y GDPR.

3.2.2. GESTIÓN Y ROTACIÓN DE CLAVES CRIPTOGRÁFICAS

Un componente crítico de cualquier solución de cifrado es la adecuada gestión de las claves utilizadas. En la propuesta, toda la administración de claves se apoya en AWS Key Management Service (KMS), lo que ofrece un entorno seguro y certificado para almacenar y usar claves criptográficas. Los lineamientos de gestión de claves en la solución son:

- **Claves Maestras en KMS:** Se define al menos una **Customer Master Key (CMK)** en KMS para la aplicación de telemedicina. Esta clave maestra es la que cifra las claves de datos usadas en S3, DynamoDB u otros servicios. Se optó por usar una CMK **administrada por el cliente** (y no solo las claves por defecto administradas por AWS) para tener control total sobre políticas de uso, posibilidad de inhabilitar la clave si fuese necesario, y cumplir ciertos estándares que requieren manejo propio de claves. Por ejemplo, para categorizar datos bajo distintas claves (segmentación) podríamos tener una CMK para datos extremadamente sensibles separada de otra para datos generales.
- **Rotación Periódica de Claves:** Se habilita la rotación automática de la CMK en KMS, con un periodo de rotación de 1 año (365 días) o menos, conforme a las recomendaciones de seguridad y los requisitos normativos. La rotación automática significa que AWS genera nuevas versiones de la clave maestra manteniendo las anteriores para poder descifrar datos antiguos, todo de forma transparente. Esto reduce el riesgo en caso de que una clave maestra se viera comprometida a largo plazo, y además cumple lineamientos de normativas (por ejemplo, algunas guías de HIPAA y estándares NIST recomiendan rotación anual o semestral).
- **Separación de Roles (Políticas IAM):** Se implementan políticas de acceso en KMS de forma que solo servicios específicos puedan usar la clave para cifrar/descifrar, mientras que otros roles administrativos tienen permiso para gestionarla (ej.: rotarla manualmente, cambiar alias) pero no para usarla en operaciones criptográficas de datos. Por ejemplo, las funciones Lambda de la aplicación reciben un rol con permisos `kms:Encrypt/kms:Decrypt` únicamente sobre la CMK designada y para las operaciones estrictamente necesarias. En cambio, el administrador de seguridad podría listar

claves o ver logs pero no cifrar datos con ellas. Esta separación de responsabilidades previene abusos internos y limita el impacto si una credencial de la aplicación se ve comprometida (no podría usarse para tareas fuera de su alcance definido).

- **Custodia en HSM (opcional):** AWS KMS ya provee protección en HSM, pero para ciertos escenarios de alta seguridad o cumplimiento regulatorio específico, se deja la puerta abierta a usar **AWS CloudHSM** o claves externas. Por ejemplo, algunas instituciones podrían requerir manejar sus propias claves en un HSM on-premise o en CloudHSM (modo *custom key store* de KMS). La arquitectura de la solución es compatible con esto, ya que la integración de cifrado está encapsulada via KMS: bastaría con cambiar la CMK por una administrada en CloudHSM o por el cliente, sin afectar al código de aplicación. En esta fase, sin embargo, no se implementó un HSM externo por simplicidad, considerando que KMS cumple con los estándares de seguridad requeridos (certificación FIPS 140-2 Nivel 2, entre otros) y ofrece un balance adecuado entre seguridad y facilidad de uso.
- **Registro y Monitoreo de Uso de Claves:** KMS se integra con AWS CloudTrail para registrar cada uso de la CMK (quién solicitó cifrar/descifrar, hora, contexto). La solución aprovecha esto para auditoría: periódicamente se revisarán los logs de CloudTrail en busca de cualquier uso anómalo de la clave (por ejemplo, si alguna identidad no esperada intentó usarla). Esto contribuye tanto a seguridad operativa como a requerimientos de cumplimiento (tener trazabilidad de acceso a datos sensibles). Además, KMS permite establecer alarmas o alertas ante ciertos eventos, que se podrían habilitar en caso de manejo de datos muy críticos.
- **Plan de Contingencia de Claves:** Aunque KMS es un servicio altamente disponible, parte de la gestión de claves incluye planificar la contingencia. La aplicación debe manejar con gracia errores en operaciones de cifrado/descifrado (p.ej., si KMS no está disponible momentáneamente en una región). La propuesta sugiere implementar re-intentos exponenciales en las funciones Lambda para llamadas a KMS y, de ser posible, *caching* temporal de resultados de cifrado no sensibles para reducir llamadas excesivas. Asimismo, si por alguna razón se necesitara deshabilitar una clave (ej. sospecha de compromiso), se definiría un procedimiento para re-encriptar datos con una nueva clave y actualizar configuraciones, minimizando tiempos de inactividad. Estas consideraciones aseguran la **disponibilidad** y **resiliencia** de la capa de cifrado.

En síntesis, la gestión de claves en la solución recae en AWS KMS con políticas estrictas y automatización de buenas prácticas (rotación, monitoreo). Esto permite mantener un alto nivel de seguridad sin tener que construir un sistema de manejo de claves desde cero. Las decisiones tomadas equilibran la **seguridad** (custodia robusta en HSM, control de accesos) con la **operatividad** (automatización y transparencia en rotación, integración nativa con servicios AWS).

3.2.3. INTEGRACIÓN DE SEGURIDAD Y PRIVACIDAD POR DISEÑO

Adicionalmente al cifrado, la solución se enmarca en un enfoque de **Seguridad y Privacidad por Diseño**. Esto significa que las consideraciones de protección de datos se incorporan en cada elemento de la arquitectura y del proceso de desarrollo, en lugar de ser añadidas después como parches. Algunos aspectos clave de este enfoque en nuestra propuesta son:

- **Principio de Mínimos Privilegios:** Se limitará estrictamente el acceso a datos de salud solo a quienes lo necesitan para sus funciones. A nivel de infraestructura, esto implica configurar políticas de acceso refinadas en DynamoDB y S3. Por ejemplo, las tablas de DynamoDB pueden usar **attribute-based access control** para que solo ciertas funciones Lambda (o usuarios con atributos específicos) puedan leer ciertos ítems. En S3, se utilizan *bucket policies* y URLs firmadas con expiración corta para que solo el especialista designado pueda acceder a un video, y solo durante el periodo necesario. Ningún componente tendrá más permisos de los requeridos: las Lambdas no relacionadas con datos clínicos no podrán tocar dichas tablas o buckets, el front-end no puede acceder directamente a recursos sin pasar por la API segura, etc. Este control fino reduce el riesgo de accesos accidentales o malintencionados.
- **Aislamiento de la Lógica de Seguridad:** Gracias a la arquitectura por capas, cualquier futura evolución en mecanismos de seguridad (por ejemplo, incluir un cifrado a nivel de aplicación más sofisticado, cambiar el proveedor de KMS, implementar algoritmos nuevos) se puede realizar en la capa de infraestructura sin “romper” la lógica de negocio. El diseño actual ya considera esta separación: las funciones de cifrado, autenticación, auditoría, etc., están desacopladas de los use cases de negocio. Esto asegura que implementar seguridad no afecte la experiencia del desarrollador en otras partes del sistema y que los cambios de seguridad no introduzcan bugs en la lógica médica. Por ejemplo, si en el futuro se decide migrar de AWS KMS a un servicio de cifrado on-premise, se podría crear un nuevo adaptador en infraestructura que implemente las mismas interfaces de cifrado, manteniendo intactos los casos de uso que simplemente invocan “guardar caso clínico” sin conocer si el dato se cifra o no.
- **Protección de Datos en Entornos No Productivos:** Un detalle de privacidad por diseño es garantizar que, en entornos de prueba o desarrollo, no haya datos reales sin proteger. La solución propone que si se usan datos reales (ej. para pruebas de integración), estos también se cifren y anonimicen. Idealmente, se usará datasets ficticios en pruebas, pero si existiera un volcado de producción a staging para pruebas, las columnas sensibles deben ir cifradas o enmascaradas. Además, las mismas políticas de cifrado de infraestructura se despliegan en todos los entornos mediante el CDK, evitando descuidos (por ejemplo, que en un ambiente de prueba alguien cree una tabla DynamoDB sin cifrado). Esto muestra una atención a la seguridad de extremo a extremo en el ciclo de vida de desarrollo.
- **Auditoría y Monitoreo Continuos:** Incorporar seguridad por diseño significa que desde ya la solución piensa en cómo detectar y responder a incidentes. Por ello, además de

cifrar, se incluyen registros de auditoría (CloudTrail, CloudWatch) y métricas de acceso que pueden ser analizadas continuamente. Se pueden configurar alarmas (en Amazon CloudWatch Alarms) para eventos sospechosos, como múltiples intentos fallidos de descifrar con KMS, accesos inusuales a los buckets de S3, etc. Esta capacidad de respuesta temprana está planificada en el diseño, cumpliendo también con requerimientos de normativas que exigen monitoreo de eventos de seguridad.

En resumen, la propuesta no solo incorpora controles específicos (cifrado, claves), sino que también **extiende principios de diseño seguro a toda la arquitectura**, garantizando que la privacidad y seguridad no sean una ocurrencia tardía sino parte integral del sistema desde esta fase.

3.2.4. FLUJOS DE CIFRADO Y DESPLIEGUE

El diseño de la solución se integra cuidadosamente con los flujos operativos de la arquitectura serverless, así como con las prácticas de despliegue continuo que utiliza el proyecto. A continuación se describen los pasos técnicos y flujos previstos para incorporar el cifrado y demás controles en el sistema:

1. **Configuración de KMS:** En la CDK Stack se define la creación o importación de la CMK que gestionará la *Data Key*.
2. **Funciones Lambda Actualizadas:** Cada función Lambda que manipule datos sensibles se modifica para:
 - a) Solicitar la *Data Key* a KMS (ya sea generándola o descriptándola).
 - b) Usar la *Data Key* para cifrar/descifrar la información antes de guardarla o de enviarla a otro componente.
3. **Almacenamiento en S3 y DynamoDB:** Tanto en S3 como en DynamoDB, se seguirá usando el cifrado en reposo, pero además se persistirán los datos clínicos críticos ya cifrados de forma *cliente-lado* (client-side encryption), reduciendo el riesgo de que un atacante con acceso a las tablas o al *bucket* obtenga el contenido en texto plano.
4. **Auditoría y Logs:** Toda interacción con KMS queda registrada en *CloudTrail*, mientras que la trazabilidad de uso de los datos se capturará en *CloudWatch Logs*, con la posibilidad de anonimizar o enmascarar ciertos campos en los registros para cumplir las restricciones normativas.
5. **Despliegue y Pruebas:** Se integra la configuración de la CMK, la lógica de cifrado y los cambios en las funciones Lambda dentro de la definición de la infraestructura en AWS CDK, permitiendo un despliegue automatizado y rastreable desde el repositorio del proyecto.

3.2.5. ESCALABILIDAD Y CONTINUIDAD DE LA SOLUCIÓN

Al introducir capas adicionales de seguridad, es necesario analizar el impacto en el desempeño y la escalabilidad para asegurar que la solución sea sostenible a largo plazo. Los puntos a destacar en este diseño son:

- **Escalado Automático de Funciones Lambda:** El cifrado y descifrado aumentarán ligeramente la carga computacional, pero AWS Lambda se escalará automáticamente para manejar picos de uso.
- **Modularidad en la Capa de *Infrastructure*:** La implementación de la lógica de cifrado se concentra en las adaptaciones (*adapters*) e *handlers* de *infrastructure*, manteniendo intacta la capa de negocio (*domain* y *application*). De esta manera, los nuevos módulos podrán fácilmente adoptar la misma estrategia de cifrado sin reescribir la lógica.
- **Rendimiento y Latencia Aceptable:** El uso de cifrado en sobre y KMS está optimizado para entornos *serverless*; se espera un impacto mínimo en la latencia de las llamadas a la API.

En conjunto, la propuesta asegura que la seguridad añadida no compromete la **escalabilidad** ni la **disponibilidad** de la plataforma, manteniendo la experiencia del usuario final fluida y la capacidad de crecimiento del sistema intacta.

3.2.6. ASPECTOS CLAVES DEL CUMPLIMIENTO NORMATIVO

El sector salud está fuertemente regulado en cuanto a protección de datos. La solución propuesta busca alinearse con los principales marcos normativos internacionales y locales, buscando garantizar que la aplicación cumpla con las obligaciones legales de confidencialidad, integridad, disponibilidad y privacidad de la información de pacientes. Algunos puntos de cumplimiento clave son:

- **HIPAA (Health Insurance Portability and Accountability Act):** Esta ley de EE.UU. exige salvaguardas técnicas para la información médica protegida (PHI). El cifrado implementado, tanto en tránsito como en reposo, cumple con una salvaguarda recomendada para que, en caso de brecha, los datos sean ilegibles (lo cual en HIPAA puede eximir de notificación de brecha si los datos estaban cifrados). Además, la separación de privilegios y registro de accesos que llevamos a cabo aborda requisitos del reglamento de seguridad de HIPAA, como controles de acceso únicos y auditoría de actividades en sistemas de información que contienen PHI.

- **GDPR (General Data Protection Regulation):** La normativa europea exige protección de datos personales y privacidad por diseño. Nuestra solución incorpora cifrado de datos personales sensibles, lo que constituye una medida de protección de datos por diseño (Art. 25 GDPR) y por defecto. En caso de violación de seguridad, el hecho de que los datos estén cifrados mitiga el riesgo para los titulares (Art. 34, notificación de brechas). Asimismo, la solución prevé mecanismos para respetar derechos ARCO: aunque no implementado en su totalidad, se deja abierta la posibilidad de portabilidad y supresión segura de datos (por ejemplo, pudiendo eliminar de forma irrecuperable los datos cifrados si un paciente lo solicita, bastando con destruir la clave asociada).
- **Legislación Chilena (Ley 19.628 y Ley 21.663):** La Ley 19.628 sobre Protección de la Vida Privada regula el tratamiento de datos personales en Chile, exigiendo consentimiento y medidas de seguridad para datos sensibles (que incluyen datos de salud). La solución refuerza la confidencialidad mediante cifrado robusto, cumpliendo con la exigencia de resguardar estos datos contra accesos no autorizados. La Ley 21.663 (Marco de Ciberseguridad) impone a los prestadores de servicios de salud la obligación de garantizar integridad, confidencialidad y disponibilidad de los sistemas. Nuestra propuesta atiende estos pilares: confidencialidad mediante cifrado, integridad a través de controles de acceso y auditoría, disponibilidad gracias a la infraestructura robusta de AWS y planes de contingencia. Además, al apoyarnos en AWS, heredamos certificaciones y buenas prácticas que facilitan demostrar el cumplimiento (AWS posee certificaciones internacionales que aportan al cumplimiento de estándares locales).
- **Otras Normativas y Estándares:** Se consideraron también lineamientos de estándares como OWASP (para manejo seguro de claves y datos), recomendaciones de NIST en criptografía (usar algoritmos aprobados como AES-256, rotación de claves), y estándares específicos de salud como los de la ATA (*American Telemedicine Association*) que sugieren medidas de seguridad para plataformas telemédicas. La solución está en línea con estas guías: por ejemplo, los registros de auditoría y la monitorización continua responden a controles típicos de ISO 27001 e ISO 27799 (seguridad de la información en salud).

En conclusión, la propuesta de solución no solo aborda cuestiones técnicas, sino que también **satisface las exigencias normativas** del entorno de datos de salud. Al implementar cifrado fuerte, control de accesos, y mantener registros de auditoría, la aplicación estará mejor preparada para pasar evaluaciones de seguridad, certificaciones y evitar sanciones derivadas de incumplimiento legal. Si bien la implementación inicial se enfoca en cifrado a nivel de infraestructura, esta ya aporta un nivel de protección significativo que sitúa a la plataforma por encima del mínimo requerido por la normativa vigente, dejando espacio para seguir mejorando con medidas adicionales en el futuro.

3.3. INTEGRACIÓN CON LA ARQUITECTURA EXISTENTE

Una consideración fundamental al introducir mejoras de seguridad es hacerlo de manera **compatible con la arquitectura vigente**, evitando interrumpir los servicios o requerir refactorizaciones masivas. La propuesta de solución de cifrado y gestión de datos sensibles se integrará con la arquitectura existente de forma gradual y ordenada, apoyándose en la capa de infraestructura para encapsular cambios. A continuación, se detalla cómo se adaptarán los módulos en la infraestructura, manteniendo los flujos actuales durante la transición y respetando los principios arquitectónicos originales (*Clean Architecture*, *DDD*) durante la incorporación de la solución.

3.3.1. ADAPTACIÓN DE MÓDULOS EN LA CAPA DE INFRAESTRUCTURA

El *envelope encryption* y la lógica de cifrado/descifrado se implementan principalmente en la capa de *infrastructure*, lo que permite mantener la independencia del *domain* y de la *application* respecto de los detalles de seguridad. Concretamente, para cada uno de los **módulos/aggregates** que manipulen datos sensibles (por ejemplo, *clinical-cases*), se agregará o ajustará un `EncryptionService`, encargado de:

1. Solicitar la *Data Key* y/o descriptarla a través de AWS KMS antes de procesar los datos.
2. Cifrar los datos recibidos desde la capa *application* previo a su persistencia (en DynamoDB o S3).
3. Realizar el *decrypt* cuando los datos cifrados sean leídos para casos de uso que requieran visualizar información en texto claro.

Uso de AWS CDK en TypeScript Dado que la infraestructura se define con **AWS CDK**, la integración contempla la creación o importación de recursos como:

- **CMK (Customer Master Key) en KMS:** Configurada con rotación automática, políticas de acceso basadas en IAM y permisos que restringen el uso de la clave a los roles autorizados de cada *bounded context*.
- **Funciones Lambda Actualizadas:** La definición de cada Lambda en el CDK Stack se modifica para que incluya las variables de entorno necesarias y los permisos de cifrado/descifrado en KMS.
- **Políticas de Bucket en S3 y Tablas de DynamoDB:** Ajustes de cifrado en reposo y reglas de acceso, adicionando, si corresponde, la encriptación del lado del cliente (*client-side encryption*).

La gestión de estos cambios se incorpora al mismo repositorio, asegurando que la definición de la solución criptográfica y los recursos de AWS se desplieguen de manera coordinada.

3.3.2. MANTENIMIENTO DE LOS FLUJOS EXISTENTES Y MIGRACIÓN PROGRESIVA

Para introducir el cifrado sin interrumpir el funcionamiento normal de la plataforma, se propone una **migración gradual** y un mantenimiento de compatibilidad durante el despliegue de la solución. Los pasos considerados para lograr esto fueron:

1. **Habilitación del Cifrado:** Inicialmente, las nuevas transacciones o los nuevos datos capturados (ej. videos subidos a S3, creación de casos clínicos) serán cifrados bajo la nueva estrategia.
2. **Proceso de Conversión de Datos Legados:** Gradualmente, se cifra la información ya existente en S3 y DynamoDB, asegurando la compatibilidad retroactiva para no interrumpir la lectura de datos no cifrados.
3. **Refactor de Lógica en Módulos Clave:** Cada `bounded context` y sus aggregates se van ajustando para utilizar los `EncryptionServices`, con pruebas unitarias y de integración que garanticen la continuidad de los procesos.

Este proceso de *roll-out* escalonado permite verificar el rendimiento y la estabilidad en cada etapa, reduciendo riesgos de afectación a los servicios en producción.

3.3.3. ALINEAMIENTO CON PRINCIPIOS DE CLEAN ARCHITECTURE Y DDD

Uno de los objetivos al introducir la solución fue **no violar los principios arquitectónicos base** que rigen el proyecto. Es decir, mantener las ventajas de modularidad, separación de conceptos y independencia de la lógica de negocio frente a detalles de implementación. Examinemos cómo el resultado final sigue alineado con Clean Architecture y DDD:

- **Dominio Inalterado:** La capa de dominio permanece exactamente igual que antes de la solución. Sus entidades (por ejemplo, la entidad `CasoClínico` con atributos como diagnóstico, paciente, etc.) no fueron modificadas para añadir, digamos, campos cifrados o métodos de seguridad. Tampoco se introdujeron dependencias desde el dominio hacia KMS, AWS o cualquier detalle técnico. Esto mantiene el modelo de dominio puro y comprensible, centrado solo en reglas de negocio. Desde la perspectiva de DDD, el *Ubiquitous Language* y los modelos no incorporaron términos criptográficos, lo cual es deseable.

- **Capa de Aplicación Orquestando Servicios Externos:** La capa de aplicación continúa invocando repositorios y servicios de infraestructura a través de interfaces abstractas, sin saber si detrás se cifran datos o no. Por ejemplo, el caso de uso `GuardarDiagnosticoUseCase` simplemente llama a `repositorioCasos.guardar(caso)`; antes, ese método guardaba directamente en DynamoDB, ahora guarda en DynamoDB cifrado, pero el caso de uso no cambia. Si en una futura iteración agregáramos cifrado en aplicación, lo haríamos proporcionando un `EncryptionService` en infraestructura e inyectándolo, pero aún así, la capa de aplicación solo llamaría a métodos como `encriptar(datos)` sin conocer la implementación. En este sentido, la solución respeta el *Dependency Inversion Principle*: la aplicación depende de abstracciones (interfaces), y la infraestructura provee las implementaciones concretas (ahora aseguradas).
- **Infraestructura Especializada en Seguridad:** La capa de infraestructura es la que absorbió todos los detalles de la solución: configuración de AWS, políticas, eventuales clases utilitarias. Esto concuerda con Clean Architecture, donde los detalles externos quedan en el nivel más bajo. Por ejemplo, aunque no codificamos una clase `EncryptionService` custom, sí podemos conceptualmente ver a AWS KMS como un “adapter” más al que nuestra infraestructura recurre. También centralizamos allí la definición de cómo conectar con CloudTrail para logs, cómo manejar los permisos, etc. Así, cualquier desarrollador trabajando en la capa de negocio no necesita entender KMS; y si trabajas en infraestructura, sabes que ahí es donde van esas preocupaciones.
- **Screaming Architecture (Intención clara):** Si alguien revisa el código organizado por módulos, verá que la introducción del cifrado no ensució la organización. Los nombres de los componentes de negocio siguen gritando su intención (examination, clinical-cases, etc.), y las adiciones de seguridad están aisladas (por ejemplo, en un módulo transversal de infraestructura o en la configuración global del CDK). Esto es coherente con el ideal de Screaming Architecture: el sistema sigue “pareciendo un software de telemedicina” más que un software de criptografía, porque los elementos criptográficos no invaden los contextos de negocio.

En resumen, la integración de la solución de seguridad se realizará **respetando los límites de capa y contexto**. Los principios de arquitectura limpia sirven precisamente para facilitar esto: gracias a la separación de responsabilidades, es posible insertar cifrado sin tener que reescribir la lógica de casos de uso ni la de dominio. De esta forma, el alineamiento arquitectónico asegura que el sistema conserve su claridad y que la seguridad añadida no introduzca deuda técnica ni confusión en el diseño.

3.3.4. PASOS TÉCNICOS PARA LA IMPLEMENTACIÓN

Para cerrar la integración de la propuesta, se listan de manera concisa los pasos técnicos principales que se llevarán a cabo para implementar la solución en el entorno existente:

1. **Creación/Actualización de la CMK en AWS KMS:** Se define en la capa de CDK la clave maestra y sus políticas de rotación.
2. **Configuración de Roles e IAM Políticas:** Asegurarse de que las funciones Lambda que manipulan datos sensibles posean sólo los permisos mínimos necesarios para efectuar operaciones de Encrypt y Decrypt en la CMK.
3. **Implementación de EncryptionService (o Adaptador) en Infraestructura:**
 - a) Métodos para generar y cifrar la *Data Key* (`generateDataKey`).
 - b) Métodos para cifrar y descifrar datos (`encryptData`, `decryptData`).
4. **Refactor de Use Cases:** En la capa *application*, se ajustan las operaciones que guardan y leen datos sensibles, inyectando el `EncryptionService` para realizar el cifrado/descifrado antes de la persistencia o la entrega de la información.
5. **Ajustes en S3 y DynamoDB:**
 - Activar cifrado *server-side* (SSE-KMS) en los *buckets* y tablas.
 - Permitir la migración progresiva de datos ya existentes a la nueva estrategia de cifrado en sobre, planificando ventanas de mantenimiento si fuera necesario.
6. **Pruebas de Integración y Validación Normativa:**
 - Validar que los datos en reposo y en tránsito se cifren adecuadamente y que la aplicación pueda descifrarlos sin pérdida de integridad.
 - Asegurar que las políticas de retención de logs y la enmascaramiento de información cumplan con las regulaciones aplicables.

3.3.5. CONSIDERACIONES DE DESEMPEÑO Y CONFIABILIDAD

Integrar la solución de cifrado conlleva también prever cómo se comportará el sistema bajo distintas condiciones y garantizar que no se introduzcan puntos únicos de falla. Algunas consideraciones finales en cuanto a desempeño y confiabilidad de la solución integrada son:

- **Latencia Introducida por KMS:** Al ser una solución *serverless*, se espera un impacto leve en la latencia, que puede mitigarse mediante la reutilización de conexiones a KMS y la implementación de estrategias de caché de claves (conforme a las mejores prácticas de AWS).

- **Escalabilidad Automática:** Dado que Lambda y KMS ofrecen escalado automático, el sistema puede manejar picos de cifrado/descifrado sin necesidad de una intervención manual.
- **Disponibilidad de la Clave y Control de Errores:** Se deben prever planes de contingencia en caso de indisponibilidad temporal de KMS o de problemas de actualización, de modo que la aplicación maneje los errores y retome el proceso de cifrado/descifrado lo antes posible.

CAPÍTULO 4

VALIDACIÓN DE LA SOLUCIÓN

Tras haber detallado el diseño de la solución en el capítulo anterior, este capítulo se dedica a su validación empírica. El propósito es demostrar, a través de un proceso metodológico y reproducible, que los controles de seguridad implementados son efectivos y robustos para proteger la información sensible manejada por la aplicación de telemedicina.

El enfoque de la validación se centra en verificar la resiliencia de la arquitectura frente a los riesgos de exfiltración de datos, un desafío crítico en el sector salud. Se busca corroborar que la estrategia de defensa en profundidad, que combina múltiples capas de cifrado y controles de acceso, protege la confidencialidad e integridad de los datos en todo su ciclo de vida.

Para alcanzar este fin, la validación se estructura en una metodología de tres fases secuenciales:

1. **Implementación del Módulo de Cifrado:** Se materializará la estrategia de cifrado a nivel de aplicación, integrándola en la arquitectura serverless existente mediante infraestructura como código.
2. **Verificación de las Capas de Seguridad:** Se confirmará la correcta configuración de las medidas de seguridad tanto para datos en reposo como para datos en tránsito.
3. **Ejecución de Pruebas de Seguridad:** Se llevará a cabo un conjunto de pruebas funcionales, de control de acceso y de auditoría para simular escenarios de amenaza y verificar la eficacia de los controles implementados.

Los resultados de estas pruebas se presentarán de forma estructurada para facilitar su análisis y discusión, corroborando así la eficacia y resiliencia de la solución diseñada.

4.1. IMPLEMENTACIÓN DEL MÓDULO DE CIFRADO

En esta sección se describe la implementación práctica de la estrategia de cifrado, que se basa en un enfoque de defensa en profundidad. Se combinan dos capas de cifrado en reposo para maximizar la protección de la información sensible.

4.1.1. CIFRADO EN EL LADO DEL SERVIDOR

El cifrado en el lado del servidor constituye la capa de protección base para los datos en reposo. Esta medida, gestionada directamente por AWS, asegura que los datos almacenados

en los servicios de persistencia como Amazon S3 y Amazon DynamoDB estén cifrados a nivel de infraestructura física. Su función principal es mitigar el riesgo asociado a un acceso no autorizado a los discos de almacenamiento subyacentes de AWS. Dentro del modelo de responsabilidad compartida, aunque el cliente es responsable de la seguridad en la nube, AWS es responsable de la seguridad de la nube, y el SSE es un mecanismo clave que el proveedor ofrece para cumplir con esta responsabilidad.

Para asegurar la aplicación consistente y automática de esta política de seguridad, se utilizan Aspects del AWS Cloud Development Kit. Este enfoque de infraestructura como código es una práctica avanzada que aplica reglas de configuración de manera transversal a todos los recursos de un tipo específico (por ejemplo, todas las tablas de DynamoDB) durante la síntesis de la infraestructura. Esto elimina la necesidad de configuración manual por recurso, reduciendo drásticamente el riesgo de error humano y asegurando el cumplimiento normativo a escala.

El siguiente código en TypeScript ilustra la implementación de un Aspect para forzar el cifrado SSE-KMS en todas las tablas de DynamoDB del proyecto, utilizando una clave gestionada por el cliente (CMK) definida en AWS Key Management Service (KMS).

```
// lib\aspects\dynamodb-sse-aspect.ts
import { IAspect } from 'aws-cdk-lib';
import { IConstruct } from 'constructs';
import { CfnTable, Table, TableEncryption } from 'aws-cdk-lib/aws-dynamodb';
import { IKey } from 'aws-cdk-lib/aws-kms';

export interface DynamoDBSSEAspectProps {
  encryptionType?: TableEncryption;
  encryptionKey?: IKey;
}

export class DynamoDBSSEAspect implements IAspect {
  private readonly props: DynamoDBSSEAspectProps;

  constructor(props?: DynamoDBSSEAspectProps) {
    this.props = props || {};
  }

  visit(node: IConstruct): void {
    if (!(node instanceof Table)) return

    const cfnTable = node.node.defaultChild as CfnTable;
    if (cfnTable && !cfnTable.sseSpecification) {
      // Si se solicitó cifrado administrado por el cliente y se proporcionó
      ↪ la clave KMS
      if (this.props.encryptionType === TableEncryption.CUSTOMER_MANAGED &&
        ↪ this.props.encryptionKey) {
```

```
        cfnTable.sseSpecification = {
            sseEnabled: true,
            sseType: 'KMS',
            kmsMasterKeyId: this.props.encryptionKey.keyId,
        };
    } else {
        // Configuración por defecto: habilita SSE (AWS owned key)
        cfnTable.sseSpecification = {
            sseEnabled: true,
        };
    }
}
}
```

Un Aspect similar se implementa para los buckets de Amazon S3, garantizando que todos los objetos, como los videos de los exámenes, se cifren en reposo utilizando la misma clave maestra.

Es importante comprender que el SSE y el cifrado a nivel de aplicación no son redundantes, sino complementarios. Constituyen una estrategia de defensa en profundidad. El SSE protege contra amenazas a nivel de infraestructura física, como el compromiso de un centro de datos de AWS. En cambio, el cifrado a nivel de aplicación, que se detalla a continuación, protege contra un conjunto diferente de vectores de ataque, como un atacante que obtiene acceso a la base de datos a través de una vulnerabilidad en la aplicación o mediante credenciales de IAM comprometidas. Si una de estas capas de seguridad llegara a fallar, la otra permanecería intacta, protegiendo la confidencialidad de los datos. Este argumento es central para demostrar la robustez del diseño de seguridad propuesto.

4.1.2. CIFRADO EN EL LADO DE LA APLICACIÓN

El cifrado en sobre implementado en el lado de la aplicación es el núcleo de la solución propuesta y representa la capa de seguridad lógica. Este método garantiza que los datos sensibles de los pacientes nunca se escriban en texto plano en los servicios de almacenamiento persistente como DynamoDB o S3. Para implementar este patrón de manera robusta y segura, se optó por utilizar el AWS Encryption SDK , una biblioteca especializada que abstrae y simplifica la complejidad del cifrado en sobre, asegurando la adhesión a las mejores prácticas criptográficas de AWS.

La implementación de esta capa se alinea perfectamente con los principios de *Clean Architecture* adoptados en el proyecto. Se crea una implementación `KmsEncryptionService` en la capa de infraestructura (ver Anexo 5.1.3), que encapsula toda la interacción con el **AWS Encryption SDK**. Esta decisión de diseño produce una sinergia fundamental entre la arqui-

tectura de software y la seguridad:

- **Desacoplamiento:** Las capas de dominio y aplicación permanecen completamente agnósticas a los detalles de la implementación criptográfica. No tienen conocimiento de KMS, claves o algoritmos. Simplemente interactúan con la interfaz genérica `EncryptionService` definida en la capa de dominio.
- **Mantenibilidad y Adaptabilidad:** El uso del AWS Encryption SDK, en lugar de una implementación manual de las llamadas a la API de KMS, reduce drásticamente el riesgo de errores criptográficos comunes. El SDK se encarga de la generación segura de claves de datos, la gestión de vectores de inicialización (IVs) y la construcción de mensajes cifrados autenticados. Además, permite que en el futuro se pueda reemplazar el proveedor de criptografía (por ejemplo, migrar a un HSM on-premise) modificando únicamente la implementación en la capa de infraestructura, sin impactar la lógica de negocio.

Flujo de Cifrado

Internamente, el `KmsEncryptionService` utiliza el AWS Encryption SDK para orquestar el siguiente flujo de cifrado en sobre de manera automática y segura :

1. **Generación de Clave de Datos:** Cuando se debe persistir un dato sensible (ej., un nuevo diagnóstico), el SDK invoca internamente la operación `GenerateDataKey` de la API de AWS KMS. Esta llamada se realiza a través de un `KmsKeyring`, que especifica la Clave Maestra de Cliente (CMK) que se debe usar.
2. **Cifrado Local:** La API de KMS devuelve una clave de datos única en texto plano (*Plaintext*) y la misma clave cifrada bajo la CMK (*CiphertextBlob*). El SDK utiliza la clave de datos en texto plano para cifrar el dato del paciente localmente, empleando un algoritmo robusto como AES-256-GCM.
3. **Empaquetado y Limpieza:** Inmediatamente después del cifrado, el SDK empaqueta el dato cifrado, la clave de datos cifrada (*CiphertextBlob*) y otros metadatos necesarios en una única estructura de datos. La clave de datos en texto plano se descarta de la memoria de forma segura.
4. **Persistencia:** Finalmente, lo que se persiste en DynamoDB es el resultado empaquetado por el SDK, en nuestro caso como una cadena de texto codificada en Base64.

Flujo de Descifrado

El proceso inverso para leer los datos es igualmente abstraído por el SDK:

1. **Recuperación de Datos Cifrados:** Para visualizar un caso clínico, la aplicación recupera el dato cifrado desde DynamoDB.
2. **Descifrado con el SDK:** El `KmsEncryptionService` pasa este dato cifrado al método `decrypt` del SDK. El SDK analiza la estructura del dato, extrae la clave de datos cifrada (`CiphertextBlob`) y la envía a la API Decrypt de AWS KMS.
3. **Verificación de Permisos y Descifrado de Clave:** AWS KMS verifica que el rol de ejecución de la función Lambda tenga permisos explícitos para usar la CMK. Si los permisos son correctos, KMS descifra el `CiphertextBlob` y devuelve la clave de datos original en texto plano al SDK.
4. **Descifrado de Datos y Limpieza:** El SDK utiliza la clave de datos en texto plano para descifrar el dato del paciente y verificar su integridad. Inmediatamente después, la clave en texto plano se descarta de la memoria.

Es fundamental destacar que este diseño, orquestado por el AWS Encryption SDK, implementa un modelo de manejo de claves seguro. La CMK actúa como la raíz de confianza, permaneciendo siempre protegida dentro de los Módulos de Seguridad de Hardware (HSM) de AWS KMS, sin ser expuesta jamás a la aplicación. Por otro lado, las claves de datos (DEKs), que son las que cifran directamente la información del paciente, son efímeras: existen en texto plano únicamente durante la operación criptográfica en la memoria segura de la función Lambda y son descartadas de inmediato. Este ciclo de vida transitorio, combinado con el principio de mínimo privilegio aplicado a los permisos de IAM, asegura que incluso en el caso de un compromiso del código de la aplicación, el atacante no pueda acceder ni a la CMK ni a las DEKs de forma persistente, garantizando así la confidencialidad de los datos a largo plazo.

4.1.3. GESTIÓN DE PERMISOS Y CLAVES

La eficacia de la estrategia de cifrado depende críticamente de una gestión de permisos rigurosa. Para que los flujos de cifrado y descifrado descritos anteriormente funcionen de manera segura, las funciones Lambda que los ejecutan deben tener permisos explícitos y mínimos para interactuar con la CMK en AWS KMS.

La definición de estos permisos como código, dentro del mismo stack de AWS CDK que define la infraestructura, es una práctica de seguridad fundamental. Este enfoque de Infraestructura como Código (IaC) garantiza que la configuración de seguridad sea versionada, auditable, reproducible y se despliegue de manera consistente en todos los entornos, eliminando la deriva de configuración y el riesgo de errores manuales.

El siguiente fragmento de código en TypeScript para AWS CDK ilustra cómo se define la CMK y se asignan los permisos necesarios, aplicando de manera programática el principio de mínimo privilegio:

```
// Dentro de la definición del Stack de CDK
// Definir la Clave Maestra de Cliente (CMK) en KMS
this.encryptionKey = new kms.Key(this, 'EncryptionKey', {
  description: 'Clave maestra para el cifrado de datos de pacientes',
  enableKeyRotation: true
});

// Considerando la función que actualiza la información del caso clínico
const updateClinicalCaseLambda = new LambdaFunction(this,
  'UpdateClinicalCaseLambda', {
    // ...
    environment: {
      TABLE_NAME: table.tableName,
      KMS_KEY_ARN: this.encryptionKey.keyArn,
    }
  }
);

// Otorgar permisos explícitos de cifrado al rol de la Lambda.
this.encryptionKey.grantEncrypt(updateClinicalCaseLambda);
// Este método asigna los permisos de:
// "kms:Encrypt",
// "kms:ReEncrypt*",
// "kms:GenerateDataKey*",
// "kms:DescribeKey"

// Considerando la función que recupera la información de un caso clínico
const findClinicalCasesByIdLambda = new LambdaFunction(this,
  'FindClinicalCasesByIdLambda', {
    // ...
    environment: {
      TABLE_NAME: table.tableName,
      KMS_KEY_ARN: this.encryptionKey.keyArn,
    }
  }
);

// Otorgar permisos explícitos de descifrado al rol de la Lambda.
this.encryptionKey.grantDecrypt(findClinicalCasesByIdLambda);
// Este método asigna los permisos de:
// "kms:Decrypt",
// "kms:ReEncrypt*",
// "kms:GenerateDataKey*",
// "kms:DescribeKey"
```

Justificación de la Estrategia de Permisos:

El uso de los métodos de alto nivel `grantEncrypt` y `grantDecrypt` del AWS CDK es una decisión de diseño deliberada que refuerza la seguridad y la mantenibilidad. En lugar de construir manualmente políticas de IAM con una lista de acciones, estos métodos encapsulan las mejores prácticas de AWS para el patrón de cifrado en sobre.

- **Adherencia al Principio de Mínimo Privilegio:** Estos métodos conceden únicamente el conjunto de permisos estrictamente necesarios para cada operación. Por ejemplo, `grantEncrypt` no solo otorga `kms:Encrypt`, sino también `kms:GenerateDataKey`, reconociendo que para cifrar un dato nuevo bajo el modelo de sobre, primero es indispensable generar una clave de datos. De manera similar, `grantDecrypt` se enfoca en la acción `kms:Decrypt`, que es la única necesaria para revertir el proceso. Este enfoque programático previene la asignación de permisos excesivos, que representarían un riesgo de seguridad significativo.
- **Abstracción y Robustez:** Al utilizar estos métodos, la implementación se abstrae de la lista exacta de permisos, que podría evolucionar con futuras versiones del AWS Encryption SDK o de los servicios de AWS. El CDK se encarga de mantener esta correspondencia, haciendo el código más limpio, legible y menos propenso a errores.
- **Auditabilidad y Cumplimiento:** Esta gestión de permisos como código es inherentemente auditable. Cualquier cambio en los privilegios de acceso a la CMK queda registrado en el control de versiones del repositorio de código. Esta trazabilidad es fundamental para demostrar el cumplimiento de normativas como HIPAA y GDPR, que exigen controles de acceso técnicos estrictos y documentados para proteger los datos personales de salud.

4.2. VALIDACIÓN DEL CIFRADO DE DATOS EN TRÁNSITO

Además de la seguridad en reposo, es imperativo garantizar que los datos estén protegidos durante su transmisión desde el cliente (el navegador del examinador) hasta el backend (Amazon API Gateway). El estándar de la industria para esta protección es el protocolo **Transport Layer Security (TLS)**. El objetivo de esta sección es validar que la comunicación con la API se realiza exclusivamente a través de una versión segura y actualizada de TLS, previniendo ataques de interceptación como *Man-in-the-Middle*.

Para esta validación, se distinguen dos escenarios operativos: el entorno de desarrollo, que utiliza el endpoint por defecto de API Gateway, y el entorno productivo, que emplearía un dominio personalizado. Por temas de alcance y recursos, las pruebas empíricas de este trabajo se ejecutaron sobre el contexto de desarrollo. No obstante, se detalla también el plan de validación propuesto y necesario para la futura implementación en producción, asegurando así una visión completa de la seguridad en tránsito del sistema.

4.2.1. ESCENARIO 1: ENDPOINT POR DEFECTO DE API GATEWAY

Para el alcance de este trabajo, las interacciones con la aplicación se realizan a través del endpoint que AWS API Gateway genera automáticamente.

- **Implementación y Verificación:** La seguridad de este endpoint es completamente gestionada por AWS. Por defecto, los endpoints regionales de API Gateway imponen una política de seguridad que exige una versión mínima de TLS 1.2 en todas las conexiones, deshabilitando protocolos más antiguos y vulnerables [Amazon Web Services, 2024a]. Aunque la documentación del proveedor es una fuente fiable, se requiere una demostración empírica para este trabajo.
- **Metodología de Prueba Empírica:** Se realizó una prueba activa utilizando la herramienta estándar openssl para intentar establecer conexiones con el endpoint de la API, forzando el uso de un protocolo obsoleto (TLS 1.0) y verificando el protocolo negociado en una conexión exitosa.

Comando 1: Intento de conexión con el protocolo obsoleto TLS 1.0.

```
openssl s_client -connect
```

```
→ <su-api-id>.execute-api.<su-region>.amazonaws.com:443 -tls1
```

Comando 2: Conexión estándar.

```
openssl s_client -connect
```

```
→ <su-api-id>.execute-api.<su-region>.amazonaws.com:443
```

- **Resultados y Veredicto:**
 - Como podemos observar en la Figura 10, la ejecución del **Comando 1** resultó en un error de handshake failure, demostrando que el servidor rechazó correctamente el protocolo obsoleto.
 - Como se observa en la Figura 11, la ejecución del **Comando 2** estableció una conexión exitosa, y la salida del comando confirmó en la línea "Protocol" que la versión negociada fue TLSv1.2.
 - **Veredicto:** Pasa. Se valida empíricamente que toda la comunicación con el endpoint por defecto está protegida por una implementación de TLS 1.2 gestionada por AWS, garantizando la confidencialidad e integridad de los datos en tránsito.

4.2.2. ESCENARIO 2: DOMINIO PERSONALIZADO

Para un despliegue en producción, se utilizará un dominio personalizado (ej. `api.app-telemedicina.cl`) para proporcionar un punto de acceso profesional y de marca a la API. En este escenario, la configuración de la seguridad es una responsabilidad compartida que debe ser configurada y validada explícitamente.

```
PS C:\Program Files\OpenSSL-Win64\bin> .\openssl s_client -connect p265vdr7hb.execute-api.sa-east-1.amazonaws.com:443 -tls1
Connecting to 18.230.87.94
CONNECTED(000001FC)
A86E0000:error:0A0000BF:SSL routines:tls_setup_handshake:no protocols available:ssl\statem\statem_lib.c:155:
-----
no peer certificate available
-----
No client certificate CA names sent
Negotiated TLS1.3 group: <NULL>
-----
SSL handshake has read 0 bytes and written 7 bytes
Verification: OK
-----
New, (NONE), Cipher is (NONE)
Protocol: TLSv1.3
This TLS version forbids renegotiation.
Compression: NONE
Expansion: NONE
No ALPN negotiated
Early data was not sent
Verify return code: 0 (ok)
-----
```

Figura 10: Resultados de ejecución del Comando 1. Error de Handshake
Fuente: Elaboración Propia.

- **Implementación Planificada:** La configuración, gestionada como Infraestructura como Código a través de AWS CDK, contempla dos elementos clave:
 1. **AWS Certificate Manager (ACM):** Se utiliza para aprovisionar o importar un certificado TLS público y válido para el dominio personalizado. ACM gestiona su renovación automática, eliminando el riesgo de interrupciones por certificados expirados [Amazon Web Services, 2024b].
 2. **Política de Seguridad de API Gateway:** Se configura explícitamente una política de seguridad para el dominio personalizado, estableciendo TLS_1_2 como la versión mínima del protocolo aceptada. Esto es fundamental para prevenir ataques de degradación (downgrade attacks) [Trend Micro, 2024].
- **Método de Validación Propuesto:** La validación de esta configuración se realizaría en un entorno de pre-producción mediante un análisis exhaustivo con herramientas externas especializadas como **Qualys SSL Labs**. Esta herramienta realiza una auditoría completa de la configuración TLS, evaluando la validez del certificado, las versiones de protocolo soportadas, la robustez de los conjuntos de cifrado (*cipher suites*) y la vulnerabilidad a ataques conocidos.
- **Criterio de Éxito Esperado:** Se espera obtener una calificación de A+ en el reporte de SSL Labs. Esto confirmaría que el dominio personalizado no solo utiliza TLS 1.2, sino que su configuración general sigue las mejores prácticas de la industria, ofreciendo el más alto nivel de seguridad para los datos en tránsito.

Al validar ambos escenarios, se demuestra que la solución es segura tanto en su estado actual de desarrollo como en su arquitectura final planificada para producción, garantizando una protección continua y robusta de la información sensible de los pacientes durante su transmisión.

```
PS C:\Program Files\OpenSSL-Win64\bin> .\openssl s_client -connect p265vdr7hb.execute-api.sa-east-1.amazonaws.com:443
Connecting to 18.230.173.205
CONNECTED(000001FC)
depth=2 C=US, O=Amazon, CN=Amazon Root CA 1
verify error:num=20:unable to get local issuer certificate
verify return:1
depth=1 C=US, O=Amazon, CN=Amazon RSA 2048 M03
verify return:1
depth=0 CN=*.execute-api.sa-east-1.amazonaws.com
verify return:1
---
Certificate chain
 0 s:CN=*.execute-api.sa-east-1.amazonaws.com
  i:C=US, O=Amazon, CN=Amazon RSA 2048 M03
  a:PKEY: RSA, 2048 (bit); sigalg: sha256WithRSAEncryption
  v:NotBefore: Aug 30 00:00:00 2024 GMT; NotAfter: Sep 28 23:59:59 2025 GMT
 1 s:C=US, O=Amazon, CN=Amazon RSA 2048 M03
  i:C=US, O=Amazon, CN=Amazon Root CA 1
  a:PKEY: RSA, 2048 (bit); sigalg: sha256WithRSAEncryption
  v:NotBefore: Aug 23 22:26:04 2022 GMT; NotAfter: Aug 23 22:26:04 2030 GMT
 2 s:C=US, O=Amazon, CN=Amazon Root CA 1
  i:C=US, ST=Arizona, L=Scottsdale, O=Starfield Technologies, Inc., CN=Starfield Services Root Certificate Authority - G2
  a:PKEY: RSA, 2048 (bit); sigalg: sha256WithRSAEncryption
  v:NotBefore: May 25 12:00:00 2015 GMT; NotAfter: Dec 31 01:00:00 2037 GMT
---
Server certificate
-----BEGIN CERTIFICATE-----
MIIEF9iCCBN6nAwIBAgIQBith/7uabthB/JD5YF5hgDANBgkqhkiG9w0BAQsFADAB
```

Figura 11: Resultados de ejecución del Comando 2. Conexión Exitosa

Fuente: Elaboración Propia.

4.3. PRUEBAS DE SEGURIDAD Y FUNCIONALIDAD

Esta sección es la culminación del proceso de validación. A través de una serie de pruebas controladas, se demuestra que la solución de seguridad no solo es funcional, sino que es efectiva contra escenarios de ataque simulados. Los resultados se presentan en una matriz de pruebas para mayor claridad.

4.3.1. PRUEBAS DE INTEGRACIÓN FUNCIONAL (HAPPY-PATH)

El primer paso de la validación empírica es verificar que la implementación del cifrado no ha comprometido la funcionalidad básica de la aplicación, asegurando que el sistema sigue siendo útil y operativo para los usuarios finales. Esta prueba se conoce como “happy-path”, ya que sigue el flujo de uso esperado sin introducir errores.

- **ID de la Prueba:** INT-01
- **Objetivo:** Validar que el ciclo completo de creación, cifrado, almacenamiento, recuperación y descifrado de un caso clínico funciona correctamente, manteniendo la integridad de los datos de principio a fin.
- **Escenario de Prueba:**
 1. Se invoca el endpoint de la API correspondiente para crear un nuevo caso clínico, enviando un objeto JSON con datos sensibles.

2. Se verifica en la base de datos DynamoDB que el ítem ha sido creado y que los datos sensibles están cifrados.
 3. Se invoca el endpoint de la API para recuperar el caso clínico recién creado.
 4. Se verifica que la respuesta de la API contiene los datos sensibles en su formato original en texto plano.
- **Evidencia y Resultados Obtenidos:** Para la ejecución de la prueba, se utilizó como dato sensible la información personal del paciente.

1. **Invocación de Creación:** Se rellenó la información del paciente de ejempli, como podemos observar en la Figura 12, lo que envió la siguiente carga útil (payload) JSON al endpoint de actualización de casos clínicos:

```
{
  "patientInfo": {
    "rut": "11.111.111-1",
    "birthDate": "1997-07-02T00:00:00.000Z",
    "gender": "male"
  }
}
```

Información del Paciente

RUT

Fecha de Nacimiento

Género

Masculino

Registrar

Figura 12: Formulario de Información del Paciente.
Fuente: Elaboración Propia.

2. **Verificación en Base de Datos:** Tras la invocación, se inspeccionó el ítem directamente en la tabla de DynamoDB. Como se evidencia en la Figura 13 y en el objeto JSON que se observa a continuación, el atributo de información del paciente no contiene el texto original, sino una cadena de texto opaca en formato Base64, correspondiente al dato cifrado por el KmsEncryptionService:

```
{
  "id": { "S": "3cd440be-f0bc-4faa-8c9b-b6c4cc4650d2" },
  "patientInfo": {
    "S": "AgV...<larga cadena de texto cifrado en Base64>...="
  },
}
```

```
"createdAt": { "S": "30/6/2025, 7:56:53" },
"status": { "S": "patient_info_completed" }
}
```

(Nota: La cadena Base64 se ha truncado por brevedad).

☐	id (Cadena)	createdAt	patientInfo	status
☐	e7c48641-7578-4a58...	30/6/2025,...		initialized
☐	3cd440be-f0bc-4faa-...	30/6/2025,...	AgV4kAAW6qs3HyDQOA6HldG...	patient_info_completed

Figura 13: Casos Clínicos Registrados en DynamoDB

Fuente: Elaboración Propia.

3. **Invocación de Recuperación y Verificación:** Finalmente, al invocar el endpoint de recuperación del caso, la API retornó el objeto JSON que podemos observar en la Figura 14, donde el campo con la información del paciente ha sido correctamente descifrado y restaurado a su valor original:

```

Headers  Preview  Response  Initiator  Timing  Adblock
[
  {
    "id": "e7c48641-7578-4a58-adf9-d4be70798ae1",
    "createdAt": "30/6/2025, 8:11:08",
    "status": "initialized"
  },
  {
    "id": "3cd440be-f0bc-4faa-8c9b-b6c4cc4650d2",
    "createdAt": "30/6/2025, 7:56:53",
    "status": "patient_info_completed",
    "patientInfo": {
      "rut": "11.111.111-1",
      "birthDate": "1997-07-02T00:00:00.000Z",
      "gender": "male"
    }
  }
]
```

Figura 14: Respuesta de API con Casos Clínicos

Fuente: Elaboración Propia.

- **Veredicto: Pasa.** El dato recuperado es bit a bit idéntico al dato original enviado. Esto demuestra que el ciclo criptográfico completo es funcional, reversible y no corrompe la información, cumpliendo con el objetivo de la prueba.

4.3.2. PRUEBAS DE CONTROL DE ACCESO Y SIMULACIÓN DE AMENAZAS (NEGATIVE-PATHS)

Estas pruebas son el núcleo de la validación de seguridad, ya que simulan activamente intentos de acceso no autorizados y demuestran la eficacia de los controles implementados contra escenarios de ataque realistas. A diferencia de las pruebas de “*happy-path*”, los “*negative-paths*” buscan confirmar que el sistema falla de manera segura cuando se enfrenta a condiciones adversas.

Prueba A: Intento de Descifrado por un Principal no Autorizado (SEC-01)

- **Objetivo:** Verificar que la política de la clave KMS y las políticas de IAM impiden de manera efectiva que una entidad no autorizada pueda realizar operaciones de descifrado, protegiendo así la raíz de confianza del sistema.
- **Metodología:** Se utilizó el **IAM Policy Simulator**, una herramienta de AWS que permite probar el efecto de las políticas de IAM de forma segura, sin realizar llamadas reales a los servicios. Se creó un rol de IAM de prueba, denominado `UnauthorizedRole`, sin ninguna política que le otorgue permisos sobre la clave KMS del proyecto.
- **Evidencia y Resultados Obtenidos:** En el simulador, se configuró el siguiente escenario:
 - **Principal:** `UnauthorizedRole`
 - **Acción:**
 - `kms:GenerateDataKey`
 - `kms:Encrypt`
 - `kms:Decrypt`
 - **Recurso:** El ARN de la Key utilizada para Cifrar

Como podemos observar en la Figura 15, al ejecutar la simulación, el resultado fue un **denied (denegado)** explícito para cada acción. El simulador indicó que la denegación se produce porque no existe ninguna política (*identity-based* o *resource-based*) que permita las acciones para el rol no autorizado sobre el recurso especificado.

- **Veredicto: Pasa.** La prueba confirma que la política de la clave KMS funciona como una barrera de control de acceso efectiva. Solo las identidades explícitamente autorizadas (como el rol de la función Lambda de recuperación de datos) pueden realizar operaciones de descifrado, validando la correcta implementación del principio de mínimo privilegio.

Prueba B: Simulación de Exfiltración de Datos desde el Almacenamiento (SEC-02)

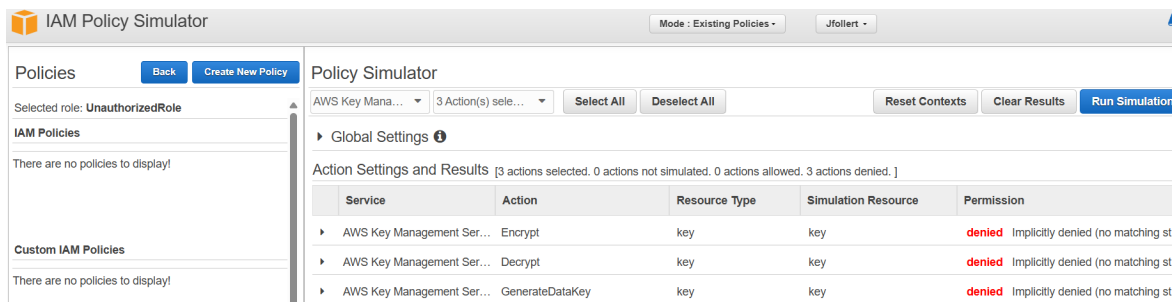


Figura 15: Resultados Simulador de Permisos - UnauthorizedRole.
Fuente: Elaboración Propia.

- **Objetivo:** Demostrar la eficacia de la estrategia de defensa en profundidad, específicamente validando que el cifrado a nivel de aplicación protege la confidencialidad de los datos incluso si un atacante logra comprometer la capa de almacenamiento.
- **Metodología:** Se simuló un escenario de ataque común en el que un actor malicioso ha obtenido credenciales con acceso de solo lectura a la base de datos (dynamodb:GetItem), pero no tiene permisos para usar la clave KMS. Se realizó una lectura directa del ítem creado en la prueba funcional (INT-01) desde la AWS CLI.
- **Evidencia y Resultados Obtenidos:** La ejecución del comando aws dynamodb get-item devo-EncryptionConstructStack-TableCD117FA1-11A5F3BI6VD0I devolvió el siguiente objeto JSON que podemos ver en la Figura 16. Se observa que, si bien los datos no sensibles como patientId son visibles, el atributo diagnóstico contiene una cadena de texto opaca e inutilizable para el atacante. Este resultado es una

```
C:\Users\josef>aws dynamodb get-item --table-name dev-EncryptionConstructStack-TableCD117FA1-11A5F3BI6VD0I
--key '{"id": {"S": "\3cd440be-f0bc-4faa-8c9b-b6c4cc4650d2\"}}'
{
  "Item": {
    "id": {
      "S": "\3cd440be-f0bc-4faa-8c9b-b6c4cc4650d2"
    },
    "createdAt": {
      "S": "30/6/2025, 7:56:53"
    },
    "status": {
      "S": "patient_info_completed"
    },
    "patientInfo": {
      "S": "AgV4kAAW6qs3HyDQ0A6HldGJ8DxnvXgLiPkJiR2IYUreoAXwABABVhd3MtY3J5cHRvLXB1YmxpYy1rZXkAREEv
U3A1dE04WTV6Ym81QytCQjMrb0xXVSszUp4T0dvcDJCeEVLQlc1REtkeHdoVFNrk2pKRncra1J0cTZkU2pIUT09AAEAB2F3cy1rbXMAS2
FybJphd3M6a21z0nNhLWVhc3QtMTowMTc3MDMyNjU3OTM6a2V5LzQ2ZTFhNDlLLWl4NWItNGM2NS1izWQ0LWY4MjMwZjRlY2EYzQc4AQIB
AHi/DoFny5IuD73auhyHn9BQcYMOyC89HgCKuBUDCOzrswH3vjFq9qmr0JdXTV7Jd6W1AAAAfjB8BgkqhkiG9w0BBwagbzBtAgEAMGgGCS
qGSIB3DQEHATAeBgglghkgBZQMEAS4wEQQMIAyUL4gaZw94Aos0AgEQgDStZ4LC4LxjMh89RDzCsOpaxovdChTrtYRbrCjWD5Spzg6oiQP1
Xp/RkaDaibxl8nYGGb1CJJ5FRk0E4wIAABAAiFGHuQqapG/iBryCpAsExDV8KGAQinXgitEwMbjHD5ptASVCDfkP9vyCnemogLnB/////w
AAAAEAAAAAAAAAAAAAAAAEAAABNtGEMAGWm9c00B3hRvQDRQRy+HpADi12KkEbsRj00L3Y72/TY2grEP/u+idUpQweWBUalsWMUvwhbyFLy
+yeqjryB0+pFEZVQP5rJxTjmvjTY8/U4Mgf2M0V98HAGcwZQIXAJbg2cSdFGr4VNd8968TnmJhsijTIz3u8C2nvDZpaTuvdBy0fw2ofS
zx6WQ9bBKCBAlwEUJpPqTsdQWEXfpm2ryXi8wXrB00o18yJzDFVWkMkx3KnBRHv4FkynM+P1p5G+R"
    }
  }
}
```

Figura 16: Consulta de Caso Clínico Directa a DynamoDB
Fuente: Elaboración Propia.

demonstración poderosa de que, incluso con un compromiso total de la capa de alma-

cenamiento, la confidencialidad de los datos de salud se mantiene intacta. El cifrado del lado del servidor (SSE) protege contra el acceso físico a los discos de AWS, mientras que el cifrado del lado de la aplicación (el que se valida aquí) protege contra el acceso lógico no autorizado a la base de datos.

- **Veredicto: Pasa.** La prueba valida exitosamente el pilar central de la propuesta de seguridad. La confidencialidad de la información no depende únicamente de los controles de acceso a la base de datos, sino de la fortaleza del cifrado a nivel de aplicación, que vuelve los datos inútiles sin la clave correspondiente.

Matriz de Pruebas de Validación

La siguiente tabla consolida los escenarios de prueba, las metodologías y los resultados, proporcionando un resumen claro y conciso del proceso de validación.

ID de Prueba	Objetivo de la Prueba	Metodología	Resultado Esperado	Resultado Obtenido	Veredicto
INT-01	Validar la integridad de los datos en el ciclo cifrado/descifrado.	Llamada a la API para crear y luego leer un caso clínico.	Los datos recuperados son idénticos a los originales.	Los datos recuperados coinciden exactamente con los datos enviados.	Pasa
SEC-01	Verificar que un principal no autorizado no puede descifrar datos.	Simulación con IAM Policy Simulator para la acción kms:Decrypt.	El simulador muestra un resultado explícito de denied.	El simulador reportó denied para el rol no autorizado.	Pasa
SEC-02	Validar la confidencialidad de los datos en caso de acceso directo al almacenamiento.	Getitem en DynamoDB a través de la consola/CLI.	Los campos sensibles se muestran como texto cifrado ilegible.	El atributo de diagnóstico contenía una cadena Base64 cifrada.	Pasa
SEC-03	Verificar que la comunicación con la API usa un protocolo seguro (TLS 1.2+).	openssl para conectar al endpoint de la API.	La conexión con protocolos obsoletos falla; la conexión normal negocia TLS 1.2+.	La conexión con TLS 1.0 falló; la conexión exitosa negoció TLSv1.2.	Pasa

Tabla 1: Matriz de Pruebas de Validación de la Solución

4.4. ANÁLISIS Y DISCUSIÓN DE RESULTADOS

Los resultados obtenidos y documentados en la matriz de pruebas confirman de manera empírica la validez y eficacia de la solución de seguridad implementada. Cada prueba superada se correlaciona directamente con la mitigación de un riesgo específico y con el cumplimiento de los objetivos planteados en este trabajo.

El éxito de la prueba de integración funcional (INT-01) demuestra que la introducción de una capa de cifrado compleja no ha comprometido la funcionalidad principal de la aplicación. Los datos mantienen su integridad a lo largo de todo su ciclo de vida, un requisito fundamental para cualquier sistema de información, y más aún en el ámbito de la salud.

Las pruebas de seguridad (SEC-01, SEC-02 y SEC-03) son las más significativas. La prueba SEC-01 valida la correcta implementación del control de acceso basado en políticas de IAM y KMS, asegurando que solo las entidades autorizadas pueden acceder a las claves de descifrado. La prueba SEC-02, por su parte, demuestra la resiliencia de la solución frente a un ataque de exfiltración de datos desde la capa de persistencia. Este escenario valida el pilar central de la propuesta: incluso si un atacante logra penetrar las defensas perimetrales y acceder directamente a la base de datos, la información sensible del paciente permanece confidencial e inutilizable. Finalmente, la prueba SEC-03 confirma que la comunicación entre el cliente y el servidor está protegida por protocolos criptográficos modernos, previniendo ataques de interceptación (*Man-in-the-Middle*).

En conclusión, la combinación de una arquitectura de software limpia, el uso de servicios gestionados de AWS, y la implementación de una estrategia de defensa en profundidad que incluye cifrado en el lado del servidor, cifrado a nivel de aplicación con gestión robusta de claves en KMS, y la validación de la seguridad en tránsito, constituye una solución integral y robusta para el problema del almacenamiento y tratamiento seguro de datos de salud en una aplicación de telemedicina. La validación empírica ha demostrado que la solución no solo es teóricamente sólida, sino prácticamente efectiva en la protección contra amenazas realistas.

CAPÍTULO 5

CONCLUSIONES

La telemedicina se ha consolidado como una alternativa poderosa para agilizar la atención de pacientes y abordar problemas como las largas listas de espera en especialidades como otorrinolaringología. Sin embargo, el manejo de datos de salud conlleva una responsabilidad ineludible en materia de seguridad y privacidad de la información, dada la naturaleza sensible de estos datos y las regulaciones vigentes, tanto a nivel local (Ley19.628, Ley21.663) como internacional (HIPAA, GDPR). En este trabajo de titulación se diseñó e implementó una solución técnica integral para el almacenamiento y tratamiento seguro de datos de salud en una aplicación de telemedicina, incorporando medidas criptográficas avanzadas sin afectar la funcionalidad ni la experiencia de usuario. A lo largo de esta sección se resumen los resultados obtenidos, se analiza el grado de cumplimiento de los objetivos planteados, se discuten las contribuciones técnicas, metodológicas y organizacionales del proyecto, se recopilan las lecciones aprendidas durante su desarrollo y se proponen líneas de trabajo futuro y recomendaciones.

RESUMEN DE RESULTADOS OBTENIDOS

La solución desarrollada consiste en una arquitectura de software que integra de forma nativa los principios de *arquitectura limpia* y *Domain-Driven Design* (DDD), complementados con patrones de diseño como *vertical slicing* y *screaming architecture*, sobre una infraestructura *serverless* en la nube de AWS. Esta base arquitectónica modular y escalable facilitó la incorporación de un mecanismo de seguridad de defensa en profundidad centrado en el cifrado robusto de los datos clínicos. Específicamente, se implementó un esquema de cifrado en sobre (*envelope encryption*) de extremo a extremo utilizando claves de cifrado simétricas administradas a través de AWS Key Management Service (KMS). Adicionalmente, se integraron servicios gestionados de AWS (como AWS Cognito para la autenticación, Amazon DynamoDB y S3 para el almacenamiento de datos, entre otros) manteniendo una estricta separación de responsabilidades entre la lógica de negocio y la infraestructura. De esta forma, la aplicación asegura la confidencialidad e integridad de la información médica almacenada, minimizando el riesgo de brechas de seguridad. Los resultados experimentales respaldan la efectividad de la solución: las pruebas de integración funcional demostraron que la introducción de esta capa de cifrado no compromete la operatividad del sistema ni la integridad de los datos en las transacciones habituales (prueba INT-01). En paralelo, una batería de pruebas de seguridad fue ejecutada para simular ataques y condiciones adversas (pruebas SEC-01, SEC-02, SEC-03), comprobándose que los controles implementados detienen eficazmente intentos de acceso no autorizados y protegen los datos sensibles aun ante escenarios de compromiso de la base de datos o el almacenamiento. En síntesis, la combinación de una arquitectura de software limpia con servicios gestionados en la nube y estrategias de cifrado robusto ha dado como resultado una solución integral y prácticamente efectiva para el problema abordado. La validación empírica confirmó que la solución propuesta no solo es teóricamente sólida,

sino también confiable en entornos reales, fortaleciendo la protección de la información de salud frente a amenazas realistas.

CUMPLIMIENTO DE OBJETIVOS

El desarrollo de este proyecto ha logrado satisfacer a cabalidad los objetivos planteados en la memoria. El objetivo principal consistía en diseñar e implementar un mecanismo de protección de datos clínicos contra filtraciones con el menor impacto posible sobre la arquitectura existente y sin sacrificar la experiencia de usuario. Dicho objetivo se ha cumplido mediante la incorporación transparente de una capa de cifrado robusto en la aplicación de telemedicina, sin alterar la lógica de negocio ni introducir penalizaciones perceptibles en el rendimiento para médicos y pacientes.

En cuanto a los objetivos específicos, todos fueron abordados exitosamente:

1. **Cifrado end-to-end con KMS:** Se adoptó un esquema de cifrado en sobre de extremo a extremo utilizando claves simétricas gestionadas por AWS KMS, tal como se propuso inicialmente. Cada dato sensible, tanto en la base de datos DynamoDB como en los archivos almacenados en S3, es cifrado utilizando una Data Key efímera protegida por una Master Key en KMS, garantizando que solo los servicios autorizados puedan descifrar la información [Weissbaum y Lugrin, 2023]. Esto asegura que, incluso si un atacante obtuviera acceso directo a la base de datos o al almacenamiento, los registros clínicos permanezcan cifrados y por tanto ininteligibles para el intruso.
2. **Rotación automática y aislamiento de claves:** Se implementó la gestión automatizada de las claves de cifrado mediante las funcionalidades nativas de KMS, cumpliendo el objetivo de minimizar la exposición de las claves y reducir la intervención operativa. Las Customer Master Keys (CMK) empleadas pueden rotarse regularmente de forma programada, y se utilizó un esquema de separación de privilegios donde ninguna entidad sin autorización explícita (políticas IAM/KMS estrictas) puede utilizar las claves para descifrar datos, tal como lo demostraron las pruebas de simulación de acceso no autorizado (SEC-01) [Krähenbühl y Perrig, 2023b]. Este aislamiento de las claves asegura un control granular sobre quién o qué componente del sistema tiene acceso a la raíz de confianza del cifrado.
3. **Cifrado en tránsito y experiencia transparente:** Se logró integrar el cifrado tanto en el almacenamiento persistente como en la transferencia de datos, de modo transparente para los usuarios finales (personal médico y pacientes). Para los datos en reposo, además del cifrado a nivel de aplicación mencionado, se aprovecharon mecanismos de cifrado del lado del servidor provistos por AWS (SSE en S3 y en la capa de almacenamiento de DynamoDB) agregando capas adicionales de protección. Para los datos en tránsito, se aseguraron las comunicaciones cliente-servidor mediante protocolos TLS de última generación, previniendo ataques de interceptación tipo *Man-in-the-Middle*. Estas medidas no requirieron intervención ni configuraciones especiales por parte de

los usuarios, cumpliendo así con el objetivo de que la seguridad añadida no entorpeciera la usabilidad ni la eficiencia del sistema. Vale la pena destacar que las pruebas de seguridad (SEC-03) confirmaron que la comunicación entre los componentes del sistema está cifrada y configurada según las mejores prácticas, sin impactar la experiencia de uso.

En resumen, todos los objetivos originales del trabajo fueron alcanzados. La solución propuesta protege efectivamente los datos de salud contra accesos indebidos y filtraciones, y lo hace manteniendo intacta la arquitectura previa de la aplicación y la calidad de la experiencia para el usuario final. Esto reafirma que es posible elevar significativamente el nivel de seguridad de sistemas de telemedicina ya existentes sin incurrir en refactorizaciones masivas ni degradar su desempeño. **Contribuciones técnicas, metodológicas y organizacionales:** Este

trabajo de titulación ha generado contribuciones relevantes en múltiples dimensiones:

- En el **ámbito técnico**, la principal contribución es la implementación de una arquitectura de cifrado de datos de salud altamente segura, integrada de manera armónica con una aplicación de telemedicina real. Se demuestra en la práctica cómo aprovechar servicios en la nube (KMS, Cognito, DynamoDB, S3, API Gateway, Lambda, entre otros) para alcanzar un nivel de seguridad y cumplimiento normativo propio de sistemas sanitarios de clase mundial, pero sin desarrollar soluciones criptográficas desde cero. El uso correcto de criptografía estándar y gestionada [Weissbaum y Lugin, 2023] elimina vulnerabilidades asociadas a implementaciones caseras y sienta una base sólida para la confidencialidad y la integridad de la información. Técnicamente, el proyecto entrega un “blueprint” o patrón de diseño que puede ser reutilizado por otras aplicaciones en la industria de la salud que requieran proteger datos sensibles en entornos de nube, combinando escalabilidad y seguridad por diseño.
- En el **ámbito metodológico**, este trabajo contribuye con una estrategia estructurada para incorporar mejoras de seguridad en un sistema existente sin interrumpir sus operaciones. Se siguió un método sistemático que incluyó: análisis de la arquitectura vigente para identificar brechas, diseño de una solución de seguridad compatible con dicha arquitectura, implementación incremental mediante módulos transversales (preservando el principio de inversión de dependencias de la arquitectura limpia) y una rigurosa fase de validación con pruebas tanto funcionales como de resistencia a ataques. Este enfoque metodológico puede servir de guía para futuras iniciativas de refuerzo de seguridad en proyectos de software, especialmente aquellos que operan en dominios regulados. Asimismo, la documentación detallada de las pruebas de penetración simuladas y de los resultados obtenidos aporta al conocimiento práctico sobre cómo evaluar la seguridad efectiva de una aplicación nativa en la nube.
- En el **ámbito organizacional**, los resultados de la investigación tienen un impacto tangible en la startup de telemedicina que sirvió de caso de estudio. La solución propuesta

eleva el estándar de protección de datos dentro de la organización, ayudando a cumplir con exigencias legales (leyes chilenas, HIPAA, GDPR) y generando mayor confianza entre los usuarios (pacientes y profesionales de la salud). Esta contribución organizacional se manifiesta en una mejora de la postura de seguridad de la empresa: ahora cuenta con un sistema más robusto frente a ciberamenazas, con políticas claras de manejo de información sensible y con mecanismos automatizados que reducen la probabilidad de error humano en la gestión de claves y datos confidenciales. Además, la experiencia obtenida a través del proyecto ha fortalecido la cultura de seguridad dentro del equipo de desarrollo, demostrando el valor de integrar consideraciones de seguridad desde las primeras etapas del ciclo de vida de software.

LECCIONES APRENDIDAS

Durante la ejecución del proyecto se obtuvieron valiosos aprendizajes que merecen ser destacados. En primer lugar, se corroboró la importancia de integrar la seguridad desde el diseño inicial de los sistemas (*security by design*). Si bien en este caso se trabajó sobre una plataforma ya existente, la aplicación de principios de arquitectura limpia y módulos independientes permitió añadir controles de seguridad robustos sin incurrir en re-arquitecturas costosas; esto refuerza la noción de que una buena modularidad y una separación clara de responsabilidades facilitan la incorporación posterior de mejoras de seguridad. En segundo lugar, la experiencia subraya el beneficio de apoyarse en servicios gestionados y estándares de la industria en lugar de intentar soluciones criptográficas ad-hoc. Al confiar en herramientas especializadas como AWS KMS para la gestión de claves y en protocolos probados para la comunicación segura, se reducen enormemente los vectores de fallo y la complejidad de la implementación [Weissbaum y Lugin, 2023]. Otra lección aprendida fue la relevancia de alinear estrechamente el desarrollo tecnológico con los requerimientos normativos: comprender a fondo regulaciones como HIPAA o la Ley 21.663 desde el comienzo guió muchas decisiones de diseño, evitando reprocesos y garantizando que la solución final estuviera en conformidad con dichas exigencias. Finalmente, se constató la necesidad de pruebas exhaustivas en materia de seguridad. Las pruebas negativas (o *negative-path*) realizadas dejaron en claro que no basta con suponer que un sistema es seguro por construcción, sino que es indispensable verificarlo simulando escenarios de ataque reales. Este enfoque proactivo de *pentesting* y validación continua es una lección aplicable a cualquier proyecto de desarrollo de software: la seguridad no es un estado absoluto, sino un proceso de mejora continua que requiere evaluación y adaptación constante.

RECOMENDACIONES Y TRABAJO FUTURO

Si bien la solución implementada cumple con los objetivos y logra un alto nivel de seguridad, siempre existen oportunidades de mejora y expansión. A continuación, se proponen algunas líneas de trabajo futuro y recomendaciones para profundizar este proyecto:

- **Integración de HSM (Hardware Security Module):** Como complemento al uso de AWS KMS, se sugiere evaluar la incorporación de módulos de seguridad de hardware para

la custodia de las claves maestras de cifrado. Un HSM proporciona un nivel adicional de aislamiento físico y protección frente a accesos no autorizados a las claves, lo que podría elevar aún más la confianza en la raíz de confianza criptográfica del sistema [Sommerhalder, 2023]. En la práctica, AWS ofrece servicios como CloudHSM que podrían integrarse con la solución existente, o alternatively, se podría considerar la migración hacia un esquema de gestión de claves híbrido que combine KMS con dispositivos HSM on-premise.

- **Análisis de rendimiento y optimización:** Profundizar en la evaluación del impacto de la capa de cifrado en el rendimiento del sistema bajo cargas de trabajo intensivas. Si bien en las pruebas funcionales realizadas no se observó una degradación notable en los tiempos de respuesta, sería valioso llevar a cabo pruebas de estrés y de volumen que midan el comportamiento del cifrado end-to-end en escenarios de uso masivo. Los resultados podrían guiar optimizaciones adicionales, por ejemplo, en la configuración de *cache* de claves de datos o en la paralelización de operaciones de cifrado/descifrado cuando se procesan múltiples registros clínicos simultáneamente.
- **Extensión a nuevos alcances de datos:** Actualmente la solución protege principalmente los datos almacenados en la base de datos y archivos clínicos en reposo. Una línea futura de desarrollo es extender mecanismos de seguridad similares a otros flujos de información dentro de la plataforma. Por ejemplo, asegurar con cifrado adicional los registros de auditoría que contengan datos sensibles, o implementar cifrado de extremo a extremo en las transmisiones de video en tiempo real (teleconsultas), garantizando que solo los participantes autorizados puedan acceder al audio/vídeo de las sesiones clínicas. Esto reforzaría aún más la confidencialidad global del ecosistema de la aplicación.
- **Certificaciones y cumplimiento avanzado:** Dado que la solución se enmarca en un contexto clínico, sería recomendable en el futuro someter el sistema a certificaciones o auditorías externas de seguridad y cumplimiento. Obtener certificaciones reconocidas (por ejemplo, ISO/IEC 27001 en gestión de seguridad de la información, o evaluaciones de conformidad con estándares HIPAA) no solo validaría la robustez de las medidas implementadas, sino que aportaría credibilidad adicional frente a clientes, entidades reguladoras y potenciales inversionistas. Asimismo, estas evaluaciones podrían revelar áreas de mejora no consideradas previamente, fomentando un ciclo de mejora continua.
- **Generalización de la solución y transferencia de conocimientos:** Por último, conviene explorar la posibilidad de abstraer y generalizar la solución de cifrado desarrollada para que pueda ser adoptada por otros sistemas de telemedicina o en general por aplicaciones que manejen datos altamente sensibles. Documentar las mejores prácticas extraídas de este proyecto y quizás desarrollar librerías o módulos reutilizables (por ejemplo, un SDK de cifrado para aplicaciones de salud) contribuiría a que la industria en su conjunto eleve sus estándares de seguridad. Adicionalmente, compartir los hallazgos y resultados en conferencias, publicaciones académicas o comunidades

de software podría tener un impacto positivo más allá de la organización, promoviendo una cultura de protección de datos en el ámbito de la transformación digital de la salud.

En conclusión, el trabajo realizado demuestra que es factible fortalecer significativamente la seguridad de una plataforma de telemedicina existente mediante el empleo inteligente de arquitectura modular y tecnologías de cifrado avanzadas. La solución propuesta logra resguardar la información sensible de pacientes frente a amenazas de seguridad actuales y futuras, al mismo tiempo que mantiene la eficiencia operativa y la facilidad de uso del sistema. Los objetivos planteados fueron alcanzados plenamente, sentando las bases para que la empresa de telemedicina opere con mayores garantías de confidencialidad y cumplimiento normativo. Se espera que las lecciones aprendidas y recomendaciones aquí expuestas sirvan de guía para proyectos venideros, ya sea para perfeccionar esta solución en particular o para inspirar desarrollos similares en otros contextos donde la privacidad de los datos es primordial. En definitiva, este proyecto aporta un camino práctico y comprobado para almacenar y tratar datos de salud de forma segura, contribuyendo a que la telemedicina siga ganando confianza y adopción tanto por parte de los profesionales como de los pacientes, en un entorno cada vez más seguro y confiable.

ANEXOS

5.1. CÓDIGO FUENTE DE LA SOLUCIÓN DE CIFRADO

Este anexo presenta los fragmentos de código fuente más relevantes que componen la solución de seguridad implementada, descrita en el Capítulo 3 y validada en el Capítulo 4 de este trabajo. El objetivo es proporcionar evidencia técnica concreta de la implementación, ilustrar la integración de los mecanismos de cifrado en la arquitectura de la aplicación y facilitar la reproducibilidad de la solución.

El código se organiza siguiendo la estructura del proyecto, reflejando los principios de Clean Architecture y Domain-Driven Design (DDD) adoptados. Se muestran los componentes clave para:

1. **Cifrado en el Lado del Servidor:** Mediante el uso de Aspects de AWS CDK para garantizar que los recursos en AWS (DynamoDB y S3) apliquen cifrado en reposo de manera automática y estandarizada.
2. **Cifrado en el Lado de la Aplicación:** A través de un servicio de cifrado (KmsEncryptionService) que utiliza el AWS Encryption SDK para realizar el cifrado en sobre (envelope encryption) de los datos sensibles antes de su persistencia.
3. **Pruebas Unitarias:** Demostrando la correctitud y robustez del servicio de cifrado a través de pruebas unitarias aisladas que simulan la interacción con los servicios de AWS.

5.1.1. Repositorio de Código Fuente

El código fuente completo de la aplicación, incluyendo la infraestructura como código, la lógica de negocio y las pruebas, se encuentra disponible para su consulta y verificación en el siguiente repositorio público:

Repositorio: Encryption Repository

5.1.2. Cifrado en el Lado del Servidor con Aspects de AWS CDK

Para cumplir con la política de cifrado en reposo (Sección 4.1.1), se implementaron Aspects en AWS CDK. Este enfoque de infraestructura como código (IaC) aplica de manera transversal y automática la configuración de cifrado a todos los recursos de un tipo específico durante la fase de despliegue, eliminando el riesgo de errores de configuración manual.

Aspecto de CDK para Cifrado Automático en DynamoDB

Este Aspect asegura que todas las tablas de Amazon DynamoDB del stack utilicen cifrado en el lado del servidor con una clave gestionada por el cliente en AWS KMS.

```
// lib\aspects\dynamodb-sse-aspect.ts
import { IAspect } from 'aws-cdk-lib';
import { IConstruct } from 'constructs';
import { CfnTable, Table, TableEncryption } from 'aws-cdk-lib/aws-dynamodb';
import { IKey } from 'aws-cdk-lib/aws-kms';

export interface DynamoDBSSEAspectProps {
  encryptionType?: TableEncryption;
  encryptionKey?: IKey;
}

export class DynamoDBSSEAspect implements IAspect {
  private readonly props: DynamoDBSSEAspectProps;

  constructor(props?: DynamoDBSSEAspectProps) {
    this.props = props || {};
  }

  visit(node: IConstruct): void {
    if (!(node instanceof Table)) return

    const cfnTable = node.node.defaultChild as CfnTable;
    if (cfnTable && !cfnTable.sseSpecification) {
      // Si se solicitó cifrado administrado por el cliente y se proporcionó
      ↪ la clave KMS
      if (this.props.encryptionType === TableEncryption.CUSTOMER_MANAGED &&
        ↪ this.props.encryptionKey) {
        cfnTable.sseSpecification = {
          sseEnabled: true,
          sseType: 'KMS',
          kmsMasterKeyId: this.props.encryptionKey.keyId,
        };
      } else {
        // Configuración por defecto: habilita SSE (AWS owned key)
        cfnTable.sseSpecification = {
          sseEnabled: true,
        };
      }
    }
  }
}
```

Aspecto de CDK para Cifrado Automático en S3

De manera análoga, este Aspect garantiza que todos los buckets de Amazon S3 cifren los objetos almacenados (como los videos de los exámenes) utilizando la misma CMK de AWS KMS.

```
// lib\aspects\buckets3-sse-aspect.ts
import { IAspect } from 'aws-cdk-lib';
import { IConstruct } from 'constructs';
import { Bucket, BucketEncryption, CfnBucket } from 'aws-cdk-lib/aws-s3';
import { IKey } from 'aws-cdk-lib/aws-kms';

export interface BucketS3SEEAAspectProps {
  encryptionType?: BucketEncryption;
  encryptionKey?: IKey;
}

export class BucketS3SEEAAspect implements IAspect {
  private readonly props: BucketS3SEEAAspectProps;

  constructor(props?: BucketS3SEEAAspectProps) {
    this.props = props || {};
  }

  visit(node: IConstruct): void {
    if (!(node instanceof Bucket)) return

    const cfnBucket = node.node.defaultChild as CfnBucket;
    if (cfnBucket && !cfnBucket.bucketEncryption) {
      if (this.props.encryptionType === BucketEncryption.KMS_MANAGED &&
        ↪ this.props.encryptionKey) {
        cfnBucket.bucketEncryption = {
          serverSideEncryptionConfiguration: [{
            serverSideEncryptionByDefault: {
              sseAlgorithm: 'aws:kms',
              kmsMasterKeyId: this.props.encryptionKey.keyId,
            },
          ]},
        };
      } else {
        // Configuración por defecto: S3-managed encryption (AES256)
        cfnBucket.bucketEncryption = {
          serverSideEncryptionConfiguration: [{
            serverSideEncryptionByDefault: {
              sseAlgorithm: 'AES256',
            },
          ]},
        };
      }
    }
  }
}
```

```
    }],  
  };  
}  
}  
}
```

5.1.3. Cifrado en el Lado de la Aplicación con AWS Encryption SDK

El núcleo de la protección contra la exfiltración de datos reside en el cifrado a nivel de aplicación. Se implementó un servicio agnóstico utilizando el AWS Encryption SDK, que abstrae y simplifica la lógica del cifrado en sobre.

Interfaz del Servicio de Cifrado (Capa de Dominio)

Esta interfaz define el contrato que debe cumplir cualquier servicio de cifrado, permitiendo que las capas de dominio y aplicación permanezcan desacopladas de la implementación concreta (AWS KMS, en este caso).

```
// contexts/src/encryption/domain/EncryptionService.ts  
  
export interface EncryptionService {  
  /**  
   * Cifra un texto plano.  
   * @param plainText El string a cifrar.  
   * @returns Un string en formato Base64 que contiene el texto cifrado y  
→ la metadata necesaria para el descifrado.  
   */  
  encrypt(plainText: string): Promise<string>;  
  
  /**  
   * Descifra un texto cifrado.  
   * @param cipherText El string en formato Base64 generado por el método  
→ encrypt.  
   * @returns El texto plano original.  
   */  
  decrypt(cipherText: string): Promise<string>;  
}
```

Implementación del Servicio de Cifrado con AWS Encryption SDK (Capa de Infraestructura)

Esta es la implementación concreta del servicio, que utiliza el AWS Encryption SDK para interactuar con AWS KMS de forma segura y eficiente.

```
// contexts\src\encryption\infrastructure\KmsEncryptionService.ts
import { KmsKeyringNode, buildClient, CommitmentPolicy } from
  ↳ '@aws-crypto/client-node';
import { EncryptionService } from '../domain/EncryptionService';

// Se construye un cliente de cifrado con una política de compromiso.
// COMMITMENT_POLICY_REQUIRE_ENCRYPT_REQUIRE_DECRYPT es la configuración más
  ↳ segura y recomendada.
const { encrypt, decrypt } = buildClient(
  CommitmentPolicy.REQUIRE_ENCRYPT_REQUIRE_DECRYPT
);

export class KmsEncryptionService implements EncryptionService {
  private readonly keyId: string;
  private readonly keyring: KmsKeyringNode;

  constructor(keyId: string) {
    this.keyId = keyId
    // El Keyring es el componente central que le dice al SDK qué Clave
      ↳ Maestra (CMK) usar en KMS.
    this.keyring = new KmsKeyringNode({
      generatorKeyId: this.keyId,
    });
  }

  async encrypt(plainText: string): Promise<string> {
    // El SDK maneja todo:
    // 1. Llama a KMS para generar una clave de datos.
    // 2. Cifra el texto plano con esa clave.
    // 3. Cifra la clave de datos con la CMK.
    // 4. Empaqueta todo en un solo resultado.
    const { result } = await encrypt(this.keyring, plainText);

    // El resultado (result) es un Buffer que contiene el texto cifrado y la
      ↳ metadata necesaria.
    // Lo convertimos a Base64 para un almacenamiento simple.
    // No necesitamos devolver un objeto complejo, solo el resultado del SDK.
    return result.toString('base64')
  }

  async decrypt(cipherText: string): Promise<string> {
    const encryptedData = Buffer.from(cipherText, 'base64');

    // El SDK se encarga de todo el proceso inverso:
    // 1. Lee la metadata del payload cifrado.
```

```
// 2. Extrae la clave de datos cifrada.  
// 3. Llama a KMS para descifrar la clave de datos.  
// 4. Usa la clave de datos en texto plano para descifrar el texto.  
const { plaintext } = await decrypt(this.keyring, encryptedData);  
  
return plaintext.toString();  
}  
}
```

REFERENCIAS BIBLIOGRÁFICAS

- [ISO, 2016] (2016). ISO/IEC 27799:2016 — Health informatics—Information security management in health using ISO/IEC 27002. Disponible en: <https://www.iso.org/standard/62777.html>.
- [ISO, 2019] (2019). ISO/IEC 27701:2019 — Security techniques—Extension to ISO/IEC 27001 and ISO/IEC 27002 for privacy information management—Requirements and guidelines. Disponible en: <https://www.iso.org/standard/71670.html>.
- [Amazon Web Services, 2024a] Amazon Web Services (2024a). Infrastructure security in amazon api gateway. Consultado el 28 de julio de 2025.
- [Amazon Web Services, 2024b] Amazon Web Services (2024b). Managed certificate renewal in aws certificate manager. Consultado el 28 de julio de 2025.
- [Commission, 2018] Commission, E. (2018). *General Data Protection Regulation (GDPR)*. Official Journal of the European Union. Disponible en: <https://eur-lex.europa.eu/eli/reg/2016/679/oj>. Consultado el 28 de julio de 2025.
- [Congreso Nacional de Chile, 1999] Congreso Nacional de Chile (1999). Ley 19.628 sobre Protección de la Vida Privada. <https://www.bcn.cl/leychile/navegar?i=141599>. Publicado en el Diario Oficial de la República de Chile. Consultado el 28 de julio de 2025.
- [Congreso Nacional de Chile, 2024] Congreso Nacional de Chile (2024). Ley 21.663 que Fortalece la Protección de Datos Personales. <https://www.bcn.cl/leychile/navegar?i=1202434>. Publicado en el Diario Oficial de la República de Chile. Consultado el 28 de julio de 2025.
- [Dong-won Kim y hee Han, 2020] Dong-won Kim, J.-y. C. y hee Han, K. (2020). Risk management-based security evaluation model for telemedicine systems.
- [Foster et al., 2009] Foster, I., Zhao, Y., Raicu, I., y Lu, S. (2009). Cloud computing and grid computing 360-degree compared. *Cloud Computing and Grid Computing 360-Degree Compared*, 5.
- [HIPAA Journal, 2023a] HIPAA Journal (2023a). Healthcare data breach statistics. Consultado el 28 de julio de 2025.
- [HIPAA Journal, 2023b] HIPAA Journal (2023b). *HIPAA Breach Notification Rule and Enforcement*. HIPAA Journal. Disponible en: <https://www.hipaajournal.com/hipaa-breach-notification-rule/>.
- [HIPAA Journal, 2024] HIPAA Journal (2024). December 2023 healthcare data breach report. Consultado el 28 de julio de 2025.

- [Iqbal et al., 2022] Iqbal, Y., Tahir, S., Tahir, H., Khan, F., Saeed, S., Almuhaideb, A., y Syed, A. (2022). A novel homomorphic approach for preserving privacy of patient data in telemedicine. *Sensors*, 22(12):4432.
- [Jin y Chen, 2015] Jin, Z. y Chen, Y. (2015). Telemedicine in the cloud era: Prospects and challenges. *IEEE Pervasive Computing*, 14(01):54–61.
- [Krähenbühl y Perrig, 2023a] Krähenbühl, C. y Perrig, A. (2023a). *Key Management*, pp. 15–20. Springer, Cham.
- [Krähenbühl y Perrig, 2023b] Krähenbühl, C. y Perrig, A. (2023b). *Key Management*, pp. 15–20. Springer, Cham.
- [Marron, 2022] Marron, J. A. (2022). Implementing the health insurance portability and accountability act (hipaa) security rule: A cybersecurity resource guide. Technical Report SP 800-66 Rev. 2, National Institute of Standards and Technology (U.S.). Consultado el 28 de julio de 2025.
- [Ministerio de Salud de Chile, 2015] Ministerio de Salud de Chile, S. d. R. A. (2015). Programa nacional de telesalud en el contexto de redes integradas de servicios de salud. https://www.minsal.cl/wp-content/uploads/2015/08/TELESALUD_webV2.pdf. Consultado el 28 de julio de 2025.
- [Rana et al., 2017] Rana, M. E., Jayabalan, M., y Kubbo, M. (2017). Privacy and security challenges towards cloud based access control in electronic health records. *Asian Journal of Information Technology*, 16(1):274–281.
- [Schaffer, 2009] Schaffer, H. (2009). X as a service, cloud computing, and the need for good judgment. *IT Professional*, 11:4–5.
- [Sommerhalder, 2023] Sommerhalder, M. (2023). *Hardware Security Module*, pp. 83–87. Springer, Cham.
- [Trend Micro, 2024] Trend Micro (2024). Ensure aws api gateway domain uses a modern security policy. Consultado el 28 de julio de 2025.
- [Weissbaum y Lugrin, 2023] Weissbaum, F. y Lugrin, T. (2023). *Symmetric Cryptography*, pp. 7–10. Springer, Cham.
- [World Health Organization, 2019] World Health Organization (2019). Who guideline: Recommendations on digital interventions for health system strengthening. <https://www.who.int/publications/i/item/9789241550505>. Consultado el 28 de julio de 2025.