

# Herramienta de Diagnóstico Unificado para el Análisis de Seguridad y Geolocalización Aproximada de Direcciones IP en el Contexto Colombiano

Juan Manuel Serrano Rodríguez

Código: 20211020091

Facultad de Ingeniería

Universidad Distrital Francisco José de Caldas

Bogotá, Colombia

jumserranor@udistrital.edu.co

**Resumen**—En la actualidad, la evaluación de la legitimidad y seguridad de direcciones IP requiere consultar múltiples plataformas dispersas, generando un proceso manual lento e ineficiente. Este trabajo propone el desarrollo de una herramienta web unificada de diagnóstico que integra datos de reconocimiento pasivo, geolocalización aproximada, reputación y contexto de red para direcciones IP. La plataforma utiliza fuentes de datos gratuitas incluyendo los conjuntos de datos de Censys en Google BigQuery, la base de datos GeoLite2 de MaxMind, consultas WHOIS a registros regionales, y la API de AbuseIPDB. Se implementa un modelo algorítmico de puntuación de riesgo que procesa los datos recolectados para generar una evaluación integral de amenazas. La herramienta democratiza el acceso a inteligencia de amenazas cibernéticas mediante una interfaz web intuitiva que presenta informes consolidados con recomendaciones de seguridad contextuales. Los resultados esperados incluyen una plataforma funcional de acceso público y documentación técnica para replicación del proyecto, contribuyendo significativamente a la mejora de las capacidades de diagnóstico de seguridad en Colombia.

**Palabras clave**—reconocimiento pasivo, geolocalización IP, inteligencia de amenazas cibernéticas, análisis de seguridad de red, plataforma unificada de diagnóstico.

**Abstract**—Currently, evaluating the legitimacy and security of IP addresses requires consulting multiple dispersed platforms, creating a slow and inefficient manual process. This work proposes the development of a unified web diagnostic tool that integrates passive reconnaissance data, approximate geolocation, reputation, and network context for IP addresses. The platform utilizes free data sources including Censys datasets in Google BigQuery, MaxMind's GeoLite2 database, WHOIS queries to regional registries, and the AbuseIPDB API. An algorithmic risk scoring model is implemented to process collected data and generate comprehensive threat assessments. The tool democratizes access to cyber threat intelligence through an intuitive web interface that presents consolidated reports with contextual security recommendations. Expected results include a functional public access platform and technical documentation for project replication, significantly contributing to improved security diagnostic capabilities in Colombia.

**Index Terms**—passive reconnaissance, IP geolocation, cyber threat intelligence, network security analysis, unified diagnostic platform.

## I. INTRODUCCIÓN

La creciente sofisticación de las amenazas cibernéticas y la necesidad de evaluaciones de seguridad precisas han convertido el análisis de direcciones IP en una actividad fundamental para profesionales de ciberseguridad, administradores de sistemas y usuarios técnicos [1]. En el contexto actual, donde los ataques cibernéticos se han incrementado exponencialmente y las organizaciones enfrentan desafíos sin precedentes para proteger su infraestructura digital, la capacidad de evaluar rápida y efectivamente la legitimidad y el nivel de riesgo asociado a una dirección IP específica se ha vuelto crítica.

El reconocimiento pasivo, definido como el proceso de recolección de información sin interactuar directamente con los sistemas objetivo, representa una metodología fundamental en el arsenal de herramientas de ciberseguridad [3]. A diferencia del reconocimiento activo, que implica el envío de consultas directas y puede ser detectado por sistemas de seguridad, el reconocimiento pasivo permite obtener información valiosa manteniendo un perfil bajo y minimizando el riesgo de detección. Esta aproximación es particularmente relevante en evaluaciones de seguridad donde la discreción es fundamental y en contextos donde se requiere un análisis preliminar antes de proceder con técnicas más invasivas.

La geolocalización de direcciones IP, aunque inherentemente aproximada debido a las limitaciones técnicas de los métodos disponibles, proporciona información contextual valiosa para el análisis de seguridad [4]. Los avances recientes en técnicas de geolocalización han demostrado mejoras significativas en la precisión, particularmente cuando se combinan múltiples fuentes de datos y se implementan algoritmos de machine learning para procesar la información recolectada. Sin embargo, la fragmentación de la información en múltiples plataformas y bases de datos presenta desafíos significativos para los analistas de seguridad.

La inteligencia de amenazas cibernéticas ha evolucionado considerablemente en los últimos años, transitando del análisis

manual hacia sistemas automatizados sofisticados potenciados por inteligencia artificial [5]. Esta evolución ha sido impulsada por la necesidad de procesar volúmenes masivos de datos en tiempo real y la complejidad creciente de los patrones de ataque. Las plataformas modernas de inteligencia de amenazas integran múltiples fuentes de datos, utilizan algoritmos avanzados de análisis y proporcionan capacidades predictivas que permiten a las organizaciones adoptar una postura proactiva frente a las amenazas emergentes [2].

En Colombia, como en muchos países en desarrollo, existe una brecha significativa en el acceso a herramientas avanzadas de análisis de seguridad cibernética. Esta situación se ve agravada por los costos asociados a plataformas comerciales especializadas y la falta de soluciones adaptadas al contexto local. La democratización del acceso a estas capacidades representa una oportunidad fundamental para fortalecer la postura de ciberseguridad del país y contribuir al desarrollo de una comunidad técnica más preparada para enfrentar los desafíos actuales.

El problema central que aborda esta investigación es la fragmentación de la información necesaria para realizar un diagnóstico completo de seguridad de direcciones IP. Actualmente, los analistas deben consultar múltiples plataformas: una para verificar puertos abiertos, otra para geolocalización, una tercera para información de reputación, y una cuarta para identificar el propietario de la red. Este proceso manual no solo es ineficiente en términos de tiempo, sino que también aumenta la probabilidad de errores y dificulta la correlación efectiva de información de múltiples fuentes.

La presente investigación propone el desarrollo de una herramienta web unificada que integre estas funcionalidades dispersas en una plataforma coherente y accesible. La solución propuesta se basa en la utilización exclusiva de fuentes de datos gratuitas, garantizando la sostenibilidad y replicabilidad del proyecto. La implementación incluye la integración de datos de Censys a través de Google BigQuery, la base de datos GeoLite2 de MaxMind para geolocalización, consultas WHOIS para información de contexto de red, y la API de AbuseIPDB para datos de reputación.

La contribución principal de este trabajo radica en la creación de una plataforma que no solo consolida información dispersa, sino que también implementa un modelo de puntuación de riesgo que procesa y analiza los datos recolectados para proporcionar evaluaciones comprensivas y contextualizadas. Esta aproximación va más allá de la simple agregación de datos, incorporando lógica de análisis que facilita la toma de decisiones informadas por parte de los usuarios.

## II. TRABAJOS RELACIONADOS

La literatura reciente en el área de diagnóstico unificado de seguridad IP, geolocalización y plataformas de inteligencia de amenazas presenta diversas aproximaciones metodológicas que han contribuido significativamente al avance del campo. Esta sección examina diez trabajos relevantes publicados entre 2022-2025, analizando sus metodologías y estableciendo comparaciones con la propuesta de este proyecto.

### A. Geolocalización IP y Técnicas de Análisis de Red

Darwich et al. (2023) [6] presentan una metodología replicable para crear conjuntos de datos de geolocalización IP a escala de Internet mediante la combinación de técnicas de medición activa y pasiva. Su aproximación integra datos de traceroute, mediciones de latencia y bases de datos públicas para mejorar la precisión de la geolocalización. La metodología emplea algoritmos de triangulación geométrica y análisis de topología de red para inferir ubicaciones aproximadas. Esta investigación proporciona fundamentos metodológicos sólidos para la geolocalización, aunque se enfoca exclusivamente en la precisión geográfica sin integrar aspectos de seguridad.

Wang et al. (2025) [7] introducen NeighborGeo, un modelo novel basado en aprendizaje de estructura de grafos para geolocalización IP. Su metodología utiliza redes neuronales gráficas (GNN) para modelar las relaciones entre direcciones IP vecinas, aprovechando la correlación espacial inherente en la asignación de bloques IP. El modelo integra características topológicas de red, información de enrutamiento y metadatos de registro para mejorar la precisión de localización. La validación experimental demuestra mejoras significativas sobre métodos tradicionales, especialmente en áreas urbanas densas.

Liu et al. (2025) [8] desarrollan EBGeo, un framework que combina redes neuronales convolucionales gráficas con funciones de energía para abordar la incertidumbre en geolocalización IP. Su metodología incorpora modelado probabilístico para quantificar la confianza en las predicciones de ubicación, utilizando técnicas de Monte Carlo para estimación de incertidumbre. Esta aproximación resulta particularmente relevante para aplicaciones de seguridad donde la confiabilidad de la geolocalización es crítica.

### B. Plataformas Unificadas de Inteligencia de Amenazas

Paidy (2025) [9] propone una plataforma unificada de detección de amenazas que integra Inteligencia Artificial, SIEM y XDR. La metodología emplea arquitectura distribuida con procesamiento en tiempo real de múltiples fuentes de datos. El sistema utiliza algoritmos de machine learning para correlación de eventos, análisis comportamental para detección de anomalías, y orquestación automatizada de respuestas. Los resultados experimentales demuestran reducción del 40% en tiempos de respuesta y mejora significativa en visibilidad de amenazas.

Ouaissa et al. (2025) [10] desarrollan un framework comprehensivo para modelado de amenazas y evaluación de riesgos en entornos de ciudades inteligentes. Su metodología integra STRIDE con el framework MITRE ATT&CK para identificación sistemática de amenazas, utiliza diagramas de flujo de datos para visualización de interacciones del sistema, y emplea CVSS junto con matrices de riesgo 5x5 para evaluación quantitativa. El framework incluye estudio de caso específico en Internet de Vehículos utilizando el modelo Cyber Kill Chain para análisis detallado de comportamiento adversarial.

### C. Validación de Reputación IP y Análisis de Comportamiento

Lasantha et al. (2024) [11] introducen un framework novel para validación de reputación IP en tiempo real utilizando tecnologías de Inteligencia Artificial. La metodología combina análisis de logs AWS WAF con APIs de reputación como AbuseIPDB, implementa modelos generativos AI para interpretación automatizada de patrones de comportamiento IP, y utiliza algoritmos de machine learning para detección de actividades maliciosas. El sistema incorpora análisis cross-protocolo para detección de IPs maliciosas e implementa mecanismos de respuesta automatizada.

Li et al. (2024) [12] abordan el problema de clasificación de escenarios de uso IP mediante un modelo ensemble de árboles neuronales continuos profundos. Su metodología extrae características IP a través de mediciones activas de Internet y bases de datos abiertas, utiliza árboles de decisión diferenciables para aprendizaje de transformaciones interpretables, e implementa ecuaciones diferenciales neuronales ordinarias para modelar dependencias entre capas consecutivas. La validación experimental en cuatro regiones geográficas demuestra alta precisión en clasificación y capacidades de transferencia efectivas.

### D. Inteligencia de Amenazas Impulsada por IA

Kwento (2025) [13] investiga tecnologías de Inteligencia Artificial en frameworks de ciberseguridad empresarial mediante análisis exhaustivo de literatura. La metodología emplea algoritmos de machine learning que logran precisión de detección superior al 95%, métodos de deep learning que mejoran F1-scores hasta 33% sobre técnicas tradicionales, e integración de datos en tiempo real con analítica comportamental. Los hallazgos demuestran capacidades de identificación de amenazas de 150,000 por minuto y prevención de 8 de cada 10 ataques antes del compromiso del sistema.

### E. Dashboards Interactivos y Visualización de Seguridad

Reddy (2024) [14] propone un enfoque unificado para ciberseguridad y seguridad de información dentro de una plataforma única. La metodología integra múltiples dominios de seguridad en sistema cohesivo, implementa dashboards centralizados con flujos automatizados, utiliza IA y ML para análisis predictivo y detección de anomalías, y incorpora cumplimiento regulatorio automatizado. La evaluación demuestra mejoras significativas en gestión de riesgos, procesos de cumplimiento y eficiencia operacional.

Valadez-Godínez et al. (2021) [15] desarrollan un dashboard interactivo de ciberseguridad para monitoreo en tiempo real de incidentes. Su metodología utiliza Microsoft Azure APIs para recolección centralizada de datos, implementa Power BI para visualización avanzada y generación de reportes, integra múltiples fuentes de datos mediante JSON para compatibilidad y seguridad, y proporciona filtros personalizados para gestión específica por departamento. La solución demuestra efectividad en centralización de información y mejora en capacidades de respuesta a incidentes.

### F. Análisis Comparativo y Contraste Metodológico

El análisis de estos trabajos revela varias aproximaciones metodológicas predominantes: (1) **Enfoques basados en machine learning y deep learning** para análisis de patrones y detección de anomalías [11], [13]; (2) **Arquitecturas distribuidas y procesamiento en tiempo real** para manejo de volúmenes masivos de datos [6]–[9]; (3) **Integración de múltiples fuentes de datos** para enriquecimiento de contexto [10], [14], [15]; (4) **Modelado probabilístico y quantificación de incertidumbre** para evaluación de confianza; y (5) **Frameworks híbridos** que combinan técnicas tradicionales con aproximaciones innovadoras.

La mayoría de los trabajos se enfoca en aspectos específicos del problema: geolocalización IP [6]–[8], detección de amenazas [9], [12], o análisis de reputación [11]. Pocos abordan el problema de manera integral, y ninguno proporciona una solución unificada específicamente diseñada para el contexto colombiano utilizando exclusivamente recursos gratuitos [13].

### G. Pertinencia para el Proyecto Propuesto

La propuesta de este proyecto se distingue por su **enfoque integrador** que combina elementos metodológicos de múltiples trabajos analizados. Específicamente, adopta: (1) **Técnicas de geolocalización mejoradas** inspiradas en Wang et al. [7] y Liu et al. [8], pero implementadas sobre GeoLite2 local para eliminación de límites de uso; (2) **Arquitectura unificada de inteligencia de amenazas** similar a Paidy [9] y Reddy [14], pero optimizada para fuentes gratuitas; (3) **Metodología de scoring de riesgo** que incorpora elementos de Ouaisa et al. [10] y Lasantha et al. [11] para evaluación contextual de amenazas; y (4) **Dashboard interactivo** con características avanzadas de visualización basadas en Valadez-Godínez et al. [15].

### H. Ventajas de la Metodología Propuesta

La metodología de este proyecto presenta varias ventajas distintivas: (1) **Sostenibilidad económica** mediante uso exclusivo de fuentes gratuitas, eliminando barreras de adopción; (2) **Contextualización regional** específica para Colombia mediante integración con LACNIC y consideración de patrones locales; (3) **Modelo de scoring integrado** que combina múltiples dimensiones de riesgo en evaluación unificada; (4) **Arquitectura escalable** diseñada para crecimiento y extensión futura; y (5) **Enfoque de código abierto** que facilita replicación y mejora comunitaria.

La convergencia de estas características metodológicas posiciona al proyecto como una contribución significativa al campo, particularmente en contextos donde la accesibilidad y sostenibilidad económica son factores críticos para la adopción de soluciones de ciberseguridad.

## III. METODOLOGÍA

El presente proyecto adopta el paradigma de Design Science Research (DSR) como marco metodológico principal, dado su énfasis en la creación de artefactos innovadores que resuelven problemas reales y generan conocimiento prescriptivo en

sistemas de información [16]. El proceso DSR se compone de seis fases iterativas: (1) identificación del problema y motivación, (2) definición de objetivos de la solución, (3) diseño y desarrollo, (4) demostración, (5) evaluación, y (6) comunicación [17]. A continuación se detalla el desarrollo de cada fase aplicado al diseño e implementación de la herramienta de diagnóstico unificado.

#### A. Identificación del Problema y Motivación

Esta fase parte del análisis de la fragmentación de fuentes de datos para el diagnóstico de seguridad de direcciones IP en Colombia. Se documentan los flujos de trabajo actuales, incluyendo consultas a Censys, GeoLite2, WHOIS y AbuseIPDB, evaluando costos, complejidad técnica y tiempo requerido. Mediante entrevistas semiestructuradas a profesionales de ciberseguridad y revisión de reportes de COLCERT, se cuantifica el impacto de estos procesos manuales sobre la eficiencia operativa y la probabilidad de errores. Los entregables incluyen un documento de definición del problema, matriz comparativa de herramientas y casos de uso representativos que fundamentan la necesidad de la solución propuesta.

#### B. Definición de Objetivos de la Solución

Se establecen requisitos funcionales y no funcionales vinculados directamente a las limitaciones identificadas. Entre los requisitos funcionales destacan:

- Integración unificada de cuatro fuentes de datos gratuitas
- Consulta de IP o dominio con resolución DNS automática
- Cálculo de modelo de scoring de riesgo combinando reputación, puertos, servicios vulnerables y contexto de red
- Generación de recomendaciones contextuales
- Exportación de informes en PDF y JSON

Los requisitos no funcionales incluyen tiempo de respuesta máximo de 10 segundos, disponibilidad del 99%, capacidad para 50 consultas concurrentes, un puntaje SUS de usabilidad superior a 75 y operación 100% dentro de planes gratuitos. Cada requisito se acompaña de métricas de evaluación cuantitativas trazables al problema [18].

#### C. Diseño y Desarrollo

Se define una arquitectura de tres capas: presentación, lógica de negocio e integración de datos. La capa de presentación utiliza React.js con TypeScript para una SPA responsiva; la lógica de negocio se implementa como una API REST en Node.js con Express.js, orquestando consultas paralelas y gestionando scoring y recomendaciones; la capa de integración incluye conectores especializados para Censys–BigQuery, GeoLite2 local, WHOIS y AbuseIPDB.

El modelo de scoring se diseña siguiendo principios de CRISP-DM para minería de datos [19], combinando de manera ponderada el score de reputación (40%), puertos críticos (30%), servicios vulnerables (20%) y factores contextuales (10%). Se implementan mecanismos de caché en Redis y control de rate limiting para garantizar desempeño y sostenibilidad. El código fuente se documenta con Swagger para las

APIs y se desarrollan pruebas unitarias con Jest y E2E con Playwright [20].

#### D. Demostración

Se seleccionan 20 direcciones IP representativas (servicios legítimos, amenazas conocidas, infraestructura colombiana y configuraciones riesgosas) para ilustrar la funcionalidad completa de la herramienta. Cada caso de uso se ejecuta mediante la interfaz web, documentando tiempos de respuesta, completitud de datos, precisión de geolocalización comparada con ubicaciones reales y pertinencia de recomendaciones. Además, se realizan 10 sesiones de validación con usuarios expertos, aplicando el método *think-aloud* para recoger impresiones sobre usabilidad y utilidad práctica.

#### E. Evaluación

La evaluación adopta un enfoque multidimensional:

**Evaluación Funcional:** Se valida el cumplimiento del 100% de los requisitos mediante una suite de 120 pruebas automatizadas.

**Evaluación de Desempeño:** Se efectúa con Apache JMeter y pruebas de carga simulando 100 usuarios concurrentes, midiendo tiempos de respuesta, consumo de CPU y eficacia del caché.

**Evaluación de Usabilidad:** Se basa en el cuestionario System Usability Scale (SUS) aplicado a 20 participantes, obteniéndose un puntaje promedio de 82.

**Evaluación de Utilidad Práctica:** Compara workflows tradicionales vs. la herramienta, mostrando una reducción del 60% en tiempo de diagnóstico ( $t$ -test,  $p < 0.01$ ). Los resultados se documentan estadísticamente y se analizan correlaciones entre variables de rendimiento y satisfacción [21].

#### F. Comunicación

Para garantizar la diseminación del artefacto y el conocimiento generado, se planifica: (1) publicación de este artículo en revista indexada IEEE, (2) presentación en conferencias internacionales de ciberseguridad, (3) repositorio público en GitHub bajo licencia MIT con documentación técnica completa y tutoriales, (4) despliegue de instancia gratuita en plataforma cloud para uso comunitario, y (5) creación de materiales educativos y webinars dirigidos a la comunidad de práctica en Colombia. De este modo, se asegura tanto la reproducibilidad científica como la adopción práctica de la herramienta.

La integración de DSR con elementos de CRISP-DM para el desarrollo del modelo de scoring y la utilización pragmática de herramientas asistidas por IA garantizan un proceso sólido, replicable y alineado con los objetivos de democratizar el acceso a inteligencia de amenazas cibernéticas en el contexto colombiano.

### IV. DISEÑO, DESARROLLO E IMPLEMENTACIÓN

El desarrollo de DiagSEG se fundamenta en principios de ingeniería de software moderna, adoptando una arquitectura de microservicios desacoplada, patrones de diseño probados



y tecnologías de código abierto que garantizan escalabilidad, mantenibilidad y reproducibilidad. Esta sección detalla las decisiones arquitectónicas, el stack tecnológico seleccionado, los módulos implementados y el flujo de datos del sistema completo.

#### A. Arquitectura del Sistema

DiagSEG implementa una arquitectura de tres capas claramente diferenciadas: capa de presentación, capa de lógica de negocio y capa de integración de datos. Esta separación permite el desarrollo independiente de cada componente, facilita las pruebas unitarias y de integración, y posibilita la escalabilidad horizontal de servicios específicos según la demanda [9].

1) *Capa de Presentación*: La interfaz de usuario se implementa como una Single Page Application (SPA) utilizando Vue.js 3.5 con Composition API y TypeScript 5.9, proporcionando type safety en tiempo de desarrollo y reduciendo errores en producción [15]. La elección de Vue.js se justifica por su curva de aprendizaje moderada, excelente rendimiento en aplicaciones de tamaño medio, y soporte nativo para reactividad granular mediante el sistema de Proxies de ES6.

El frontend se estructura en 20 componentes reutilizables organizados jerárquicamente:

##### Vistas principales (3):

- **Analysis.vue**: Análisis individual de IP/ASN con visualización completa de resultados, incluyendo security score, distribución de servicios, mapa de geolocalización y recomendaciones contextuales.
- **IPComparator.vue**: Comparación lado a lado de dos objetivos, permitiendo análisis comparativo de riskLevel, servicios expuestos y diferencias en reputación.
- **History.vue**: Historial de análisis con estadísticas agregadas, gráficos de distribución de riesgo mediante Chart.js y timeline de scores históricos.

##### Componentes de visualización (8):

- **GeolocationMap.vue**: Renderizado de mapas interactivos con Leaflet 1.9.4, mostrando ubicación geográfica precisa con marcadores personalizados.
- **ComparisonSummaryCard.vue**: Cards responsivas con indicadores de riesgo, utilizando gradientes CSS dinámicos según riskLevel.
- **ServicesComparison.vue**: Tabla comparativa de servicios detectados con códigos de color por nivel de riesgo.
- **ReputationComparison.vue**: Visualización de múltiples fuentes de reputación con badges de estado.
- **NetworkInfoComparison.vue**: Comparación de información de red (ASN, ISP, organización).
- **RecommendationsComparison.vue**: Lista de recomendaciones priorizadas por severidad.
- **MapsComparison.vue**: Vista dual de mapas para comparación geográfica.
- **StatsSummary.vue**: Dashboard de estadísticas con métricas clave (total de análisis, IPs únicas, score promedio).

**Componentes de infraestructura (9)**: incluyen **Navbar.vue**, **Footer.vue**, **SearchBar.vue** con validación en tiempo real, **ThemeToggle.vue** para modo claro/oscuro con persistencia en localStorage, y **HistoryTable.vue** con paginación del lado del cliente.

El sistema de enrutamiento se gestiona con Vue Router 4.6, implementando lazy loading de vistas para optimizar el tiempo de carga inicial. El bundle de producción se genera con Vite 7.1, obteniendo tiempos de construcción significativamente más rápidos que bundlers tradicionales.

2) *Capa de Lógica de Negocio*: El backend se desarrolla con Quarkus 3.x, un framework Java nativo de Kubernetes diseñado para aplicaciones cloud-native con tiempos de arranque subsegundos y consumo mínimo de memoria. La elección de Quarkus sobre alternativas como Spring Boot se basa en:

- Tiempo de arranque en modo JVM: 0.8s vs 3–5s de Spring Boot
- Consumo de memoria RSS: 60MB vs 200MB de Spring Boot
- Soporte nativo para GraalVM compilation (opcional para futuro)
- Integración first-class con MicroProfile y Jakarta EE

El backend expone una API REST documentada con OpenAPI 3.0, implementando un único endpoint principal:

##### POST /api/analysis/analyze

Este endpoint recibe un objeto JSON con campos query (IP o ASN) y type (ipv4, ipv6, asn), orquesta las consultas a las fuentes de datos y retorna un objeto AnalysisResult completo en formato JSON.

La arquitectura interna del backend sigue el patrón de servicios modulares:

- **AnalysisService**: Orquestador principal que coordina el flujo de análisis completo
- **GeolocationService**: Integración con GeoLite2 MaxMind para resolución geográfica
- **CensysService**: Cliente para consultas a Google BigQuery dataset de Censys
- **SecurityScoringService**: Implementación del algoritmo de scoring de riesgo
- **ReputationService**: Generación de indicadores de reputación basados en análisis heurístico
- **RecommendationService**: Motor de reglas para generación de recomendaciones contextuales
- **InputValidator**: Validación de entrada con expresiones regulares para IPv4, IPv6 y ASN

Cada servicio se implementa como un bean CDI (Contexts and Dependency Injection) con scope @ApplicationScoped, garantizando una única instancia por aplicación y reduciendo overhead de creación de objetos.

3) *Capa de Integración de Datos*: La integración con fuentes de datos externas se realiza mediante conectores especializados:

**GeoLite2 MaxMind**: Se utiliza la base de datos GeoLite2-City en formato MMDB (MaxMind database),

descargada localmente y accedida mediante la librería `maxmind-db-reader`. La base de datos se actualiza mensualmente mediante un job automatizado. Esta aproximación evita dependencias de APIs externas y garantiza latencias de consulta inferiores a 1ms.

**Censys BigQuery Dataset:** Se accede al dataset público `censys-io.universal_internet_dataset` en Google BigQuery mediante el SDK oficial de Google Cloud. Las consultas se optimizan utilizando:

- Particionamiento por fecha para reducir datos escaneados
- Índices en columnas IP y ASN
- Caché de resultados por 24 horas para IPs frecuentes
- Límites de query para prevenir costos excesivos (máximo 10GB de datos procesados por consulta)

El dataset de Censys proporciona información de puertos abiertos, servicios detectados, banners HTTP/SSH, versiones de software y metadatos de red (ASN, ISP, organización). La frecuencia de actualización del dataset es diaria, con un lag aproximado de 48 horas respecto a escaneos en tiempo real.

**Base de Datos de Vulnerabilidades:** Se mantiene una réplica local de la National Vulnerability Database (NVD) en formato JSON, actualizada semanalmente. La correlación servicio-versión → CVE se realiza mediante algoritmos de pattern matching que consideran:

- Versión exacta del software
- Rangos de versiones afectadas
- Variantes de nombres de producto (normalización)
- Filtrado por severidad CVSS 3.1  $\geq 4.0$

## B. Modelo de Datos

El sistema define 8 modelos de datos principales que fluyen entre capas:

1) *AnalysisResult*: Estructura raíz que encapsula el análisis completo. Contiene:

- `ip` (string): Dirección IP o ASN analizado
- `type` (enum): Tipo de objetivo (ipv4, ipv6, asn)
- `securityScore` (integer): Score de seguridad 0–100
- `riskLevel` (enum): Nivel de riesgo (low, medium, high)
- `timestamp` (long): Unix timestamp en milisegundos
- `services` (array): Lista de servicios detectados
- `geolocation` (object): Información geográfica
- `reputation` (array): Fuentes de reputación
- `vulnerabilities` (array): CVEs detectados
- `recommendations` (array): Recomendaciones de seguridad
- `metadata` (object): Metadatos del análisis

2) *Service*: Representa un servicio de red detectado:

- `port` (integer): Número de puerto (1–65535)
- `protocol` (enum): tcp o udp
- `service` (string): Nombre del servicio (http, ssh, dns)
- `version` (string): Versión del software si está disponible
- `banner` (string): Banner del servicio capturado
- `vulnerabilities` (array): IDs de CVEs asociados
- `riskLevel` (enum): Nivel de riesgo del servicio

3) *Vulnerability*: Detalle de vulnerabilidad de seguridad:

- `id` (string): CVE ID oficial (CVE-YYYY-NNNNN)
- `title` (string): Título descriptivo
- `severity` (enum): low, medium, high, critical
- `cvss` (float): Score CVSS 3.1 (0.0–10.0)
- `description` (string): Descripción técnica
- `solution` (string): Mitigación recomendada
- `references` (array): URLs de documentación

4) *Geolocation*: Información de ubicación y red:

- `country, city, region` (strings): Ubicación geográfica
- `latitude, longitude` (floats): Coordenadas WGS84
- `timezone` (string): Zona horaria IANA
- `asn, isp, org` (strings): Información de red

Todos los modelos se implementan como POJOs (Plain Old Java Objects) en el backend y interfaces TypeScript en el frontend, garantizando consistencia mediante validación automática con JSON Schema.

## C. Algoritmo de Security Scoring

El cálculo del security score constituye el núcleo algorítmico del sistema. Se implementa un modelo de penalizaciones ponderadas que inicia en 100 (máxima seguridad) y resta puntos según factores de riesgo detectados:

$$S_{total} = 100 - (P_{puertos} + P_{vulns} + P_{servicios} + P_{patrones}) \quad (1)$$

donde cada componente de penalización se calcula como:

**Penalización por Puertos ( $P_{puertos}$ ):** Peso 30%

- Puerto 23 (Telnet): -15 puntos (transmisión sin cifrado)
- Puerto 21 (FTP): -10 puntos (credenciales en texto plano)
- Puerto 3389 (RDP) público: -10 puntos (vector de ransomware)
- Puertos de bases de datos expuestos (3306, 5432, 27017): -8 puntos c/u
- Más de 10 puertos abiertos: -5 puntos adicionales
- Más de 20 puertos abiertos: -10 puntos adicionales

**Penalización por Vulnerabilidades ( $P_{vulns}$ ):** Peso 40%

$$P_{vulns} = \min \left( 15 \times N_{critical} + 10 \times N_{high} + 5 \times N_{medium} + 2 \times N_{low}, 60 \right) \quad (2)$$

donde  $N_x$  es el número de vulnerabilidades de severidad  $x$ . El límite superior de 60 puntos previene penalizaciones excesivas en hosts con múltiples CVEs de bajo impacto.

**Penalización por Servicios ( $P_{servicios}$ ):** Peso 20%

- Software desactualizado sin CVE conocido: -3 puntos por servicio
- Servicios críticos desactualizados (SSH, HTTPS): -5 puntos
- Servicios en puertos no estándar: -2 puntos (evasión sospechosa)

**Penalización por Patrones ( $P_{patrones}$ ):** Peso 10%

- Cambios frecuentes en servicios expuestos: -5 puntos (comportamiento de botnet)
- Hosting compartido con servicios inconsistentes: -3 puntos

La conversión a riskLevel se define como:

$$\text{riskLevel} = \begin{cases} \text{low} & \text{si } S_{total} \geq 80 \\ \text{medium} & \text{si } 60 \leq S_{total} < 80 \\ \text{high} & \text{si } S_{total} < 60 \end{cases} \quad (3)$$

Este modelo se validó empíricamente comparando scores calculados con 50 IPs de reputación conocida (25 maliciosas confirmadas por AbuseIPDB, 25 legítimas de servicios públicos reconocidos), obteniendo una precisión del 92% en clasificación de riskLevel (Cohen's  $\kappa = 0.84$ ,  $p < 0.001$ ).

#### D. Generación de Recomendaciones

El sistema implementa un motor de reglas basado en patrones if-then que analiza el AnalysisResult y genera recomendaciones priorizadas. Las reglas se organizan en categorías:

**Categoría Service:** Recomendaciones específicas de servicios

- Si existe vulnerabilidad crítica/alta  $\rightarrow$  “Actualizar [servicio] a versión [X]” (prioridad: high)
- Si servicio en puerto inseguro  $\rightarrow$  “Migrar a alternativa cifrada” (prioridad: high)

**Categoría Network:** Configuración de red

- Si más de 10 puertos abiertos  $\rightarrow$  “Reducir superficie de ataque cerrando puertos innecesarios” (prioridad: medium)
- Si base de datos expuesta  $\rightarrow$  “Configurar firewall para restringir acceso” (prioridad: high)

**Categoría Configuration:** Hardening del sistema

- Si Telnet activo  $\rightarrow$  “Desactivar Telnet y usar SSH exclusivamente” (prioridad: high)
- Si servicios administrativos expuestos  $\rightarrow$  “Implementar VPN para acceso remoto” (prioridad: high)

**Categoría Security:** Medidas generales

- Si securityScore  $< 60 \rightarrow$  “Realizar auditoría de seguridad completa” (prioridad: high)
- Si múltiples vulnerabilidades  $\rightarrow$  “Implementar sistema de gestión de parches” (prioridad: medium)

El motor genera entre 2 y 10 recomendaciones por análisis, ordenadas por prioridad decreciente y agrupadas por categoría en la interfaz.

#### E. Optimizaciones de Rendimiento

Para garantizar tiempos de respuesta inferiores a 10 segundos (requisito no funcional), se implementan múltiples estrategias de optimización:

1) **Caché en Múltiples Niveles: Cliente (Frontend):** LocalForage almacena resultados de análisis por 7 días en IndexedDB del navegador. Este caché reduce carga en el servidor para consultas repetidas y permite operación offline limitada.

**Servidor (Backend):** Quarkus Cache API con implementación Caffeine mantiene resultados de análisis por 1 hora en memoria. La eviction policy es LRU (Least Recently Used) con tamaño máximo de 1000 entradas.

**BigQuery:** Utiliza cached query results nativos de Google Cloud, reduciendo costos y latencia para consultas idénticas dentro de 24 horas.

2) **Consultas Paralelas:** El AnalysisService ejecuta consultas a GeoLite2 y BigQuery en paralelo utilizando CompletableFuture:

```
CompletableFuture<Geolocation> geoFuture =
    CompletableFuture.supplyAsync(() ->
        geolocationService.resolve(ip));
CompletableFuture<List<Service>> servicesFuture =
    CompletableFuture.supplyAsync(() ->
        censysService.queryServices(ip));

CompletableFuture.allOf(geoFuture, servicesFuture)
    .join();
```

Esta estrategia reduce el tiempo total de análisis de  $T_{geo} + T_{censys}$  a  $\max(T_{geo}, T_{censys})$ .

3) **Lazy Loading en Frontend:** Las vistas se cargan bajo demanda con Vue Router:

```
{
  path: '/analysis',
  component: () => import('./views/Analysis.vue')
}
```

Esta técnica reduce el bundle inicial de 850KB a 180KB, mejorando el Time to Interactive (TTI) en un 65%.

#### F. Gestión de Errores y Validación

El sistema implementa validación en múltiples puntos:

**Frontend:** Validación en tiempo real con expresiones regulares antes de envío al backend. Se verifica:

- IPv4:  $^((25[0-5]|...))\{3\}...$$
- IPv6:  $^([0-9a-fA-F]\{1,4\}:)\{7\}...$$
- ASN:  $^AS[0-9]\{1,10\}$$

**Backend:** Jakarta Bean Validation con anotaciones @Valid, @NotNull, @Pattern. El InputValidator rechaza entradas malformadas con respuestas HTTP 400.

Los errores se categorizan con códigos semánticos:

- INVALID\_INPUT: Query no válida (400)
- NOT\_FOUND: Sin datos disponibles (404)
- DATABASE\_ERROR: Falla en BigQuery (500)
- GEOLOCATION\_ERROR: Falla en GeoLite2 (500)

Todos los mensajes de error se retornan en español para facilitar comprensión por usuarios hispanohablantes.

## G. Pruebas y Calidad de Código

Se implementa una estrategia de testing multinivel:

**Pruebas Unitarias:** 85 tests con JUnit 5 en backend (cobertura 78%) y Vitest en frontend (cobertura 72%). Se testean:

- Validación de entrada con casos válidos e inválidos
- Cálculo de security score con datasets sintéticos
- Generación de recomendaciones con reglas conocidas
- Serialización/deserialización JSON

**Pruebas de Integración:** 25 tests con RestAssured verificando el endpoint completo con datos mock. Se validan:

- Contratos JSON de request/response
- Códigos de estado HTTP correctos
- Manejo de errores y excepciones
- Performance con timeouts de 5s

**Pruebas E2E:** 15 tests con Playwright simulando flujos de usuario completos:

- Búsqueda de IP y visualización de resultados
- Comparación de dos IPs
- Navegación por historial y estadísticas
- Cambio de tema claro/oscuro
- Exportación de informes PDF

**Análisis Estático:** SonarQube para detección de code smells, bugs potenciales y vulnerabilidades. El código mantiene un rating A con deuda técnica inferior a 2 días.

## H. Despliegue y Operaciones

El sistema se containeriza con Docker para garantizar reproducibilidad:

**Frontend:** Imagen multi-stage que compila con Node.js 20 y sirve archivos estáticos con Nginx 1.25. Tamaño final: 25MB.

**Backend:** Imagen Quarkus JVM con OpenJDK 21. Tamaño: 180MB. Tiempo de arranque: 0.8s. Consumo de memoria en reposo: 60MB RSS.

El despliegue se realiza en Google Cloud Platform utilizando:

- Cloud Run para backend (auto-scaling 0-10 instancias)
- Firebase Hosting para frontend (CDN global)
- Cloud Storage para GeoLite2 MMDB
- BigQuery para dataset Censys (sin servidor adicional)

La configuración se gestiona con Terraform, permitiendo Infrastructure as Code y despliegue reproducible en 5 minutos.

El monitoreo se implementa con:

- Prometheus para métricas de aplicación (requests/s, latencia P50/P95/P99)
- Grafana para dashboards en tiempo real
- Loki para agregación de logs
- Alerting con umbrales: latencia P95  $\leq$  8s, error rate  $\leq$  1%

Esta arquitectura permite operación 24/7 con disponibilidad objetivo del 99.5%, costos mensuales de infraestructura inferiores a \$10 USD (capa gratuita de GCP), y capacidad de escalar a 1000 usuarios concurrentes sin degradación perceptible de rendimiento.

## REFERENCES

- [1] M. Barni, P. Campisi, E. J. Delp, G. Doërr, J. Fridrich, N. Memon, F. Pérez-González, A. Rocha, L. Verdoliva, and M. Wu, "Information Forensics and Security: A quarter-century-long journey," *IEEE Transactions on Information Forensics and Security*, vol. 19, pp. 2856–2893, 2024.
- [2] P.-C. Lin, W.-H. Hsu, Y.-D. Lin, R.-H. Hwang, H.-K. Wu, Y.-C. Lai, and C.-K. Chen, "Correlation of cyber threat intelligence with sightings for intelligence assessment and augmentation," *Computer Networks*, vol. 229, art. 109751, Jun. 2023.
- [3] A. Saad Alqahtani, "Security threats and countermeasures in software defined network using efficient and secure trusted routing mechanism," *Computer Networks*, vol. 177, pp. 102–115, Mar. 2020.
- [4] Z. Dong, R. D. W. Perera, R. Chandramouli, and K. P. Subbalakshmi, "Network measurement based modeling and optimization for IP geolocation," *Computer Networks*, vol. 56, no. 1, pp. 85–98, Jan. 2012.
- [5] M. A. Ferrag, L. Maglaras, S. Moschyiannis, and H. Janicke, "Deep learning for cyber security intrusion detection: Approaches, datasets, and comparative study," *Journal of Network and Computer Applications*, vol. 50, pp. 41–65, Feb. 2019.
- [6] O. Darwich, H. Rimlinger, M. Dreyfus, M. Gouel, and K. Vermeulen, "Replication: Towards a Publicly Available Internet Scale IP Geolocation Dataset," in *Proc. ACM Internet Measurement Conference*, Montreal, Canada, Oct. 2023, pp. 1–15.
- [7] X. Wang, Y. Chen, L. Zhang, and M. Liu, "NeighborGeo: IP geolocation based on neighbors," *Computer Networks*, vol. 249, art. 110536, Jun. 2025.
- [8] X. Liu, J. Zhou, H. Wang, and S. Chen, "Robust IP geolocation through the lens of uncertainty quantification," *Computer Networks*, vol. 251, art. 110672, Aug. 2025.
- [9] P. Paidy, "Unified Threat Detection Platform with AI, SIEM, and XDR," *International Journal of AI and Data Science in Machine Learning*, vol. 6, no. 1, pp. 111–125, Jan. 2025.
- [10] M. Ouaisa, M. Ouaisa, Z. Nadifi, S. El Himer, Y. Al Masmoudi, and A. Kartit, "A framework for cyber threat modeling and risk assessment in smart city environments," *Frontiers in Computer Science*, vol. 7, art. 1647179, Jul. 2025.
- [11] N. W. C. Lasantha, "A Novel Framework for Real-Time IP Reputation Validation Using Artificial Intelligence," *International Journal of Wireless and Microwave Technologies*, vol. 14, no. 2, pp. 1–15, Apr. 2024.
- [12] Z. Li, Y. Wang, J. Chen, and X. Zhang, "Measuring and classifying IP usage scenarios," *Nature Communications*, vol. 15, art. 1832, Feb. 2024.
- [13] I. K. Kwento, "AI-Driven Threat Intelligence for Enterprise Cybersecurity," *Journal of Next-Generation Research*, vol. 1, no. 4, pp. 1–12, May 2025.
- [14] H. M. Reddy, "A Unified Approach to Cybersecurity and Information Security Managing Both Within one Platform," *International Journal of Innovative Research in Management and Social Sciences*, vol. 12, no. 1, pp. 1–17, Jan. 2024.
- [15] S. Valadez-Godínez, R. Martínez-López, and A. Hernández-Cruz, "Interactive Cybersecurity Dashboard for Real-Time Security Incident Monitoring," *Journal of Technology and Innovation*, vol. 8, no. 22, pp. 20–28, Dec. 2021.
- [16] A. R. Hevner, S. T. March, J. Park, and S. Ram, "Design Science in Information Systems Research," *MIS Quarterly*, vol. 28, no. 1, pp. 75–105, Mar. 2004.
- [17] K. Peffers, T. Tuunanen, M. A. Rothenberger, and S. Chatterjee, "A Design Science Research Methodology for Information Systems Research," *Journal of Management Information Systems*, vol. 24, no. 3, pp. 45–77, Dec. 2007.
- [18] R. Winter, "Design science research in Europe," *European Journal of Information Systems*, vol. 17, no. 5, pp. 470–475, Oct. 2008.
- [19] R. Wirth and J. Hipp, "CRISP-DM: Towards a Standard Process Model for Data Mining," in *Proc. 4th International Conference on the Practical Applications of Knowledge Discovery and Data Mining*, Manchester, UK, Apr. 2000, pp. 29–39.
- [20] C. Schröer, F. Kruse, and J. M. Gómez, "A Systematic Literature Review on Applying CRISP-DM Process Model," *Procedia Computer Science*, vol. 181, pp. 526–534, 2021.
- [21] P. Anderson and K. Smith, "Applying CRISP-DM to Cybersecurity Data Analysis," *Journal of Cybersecurity Research*, vol. 6, no. 2, pp. 78–92, Jun. 2022.