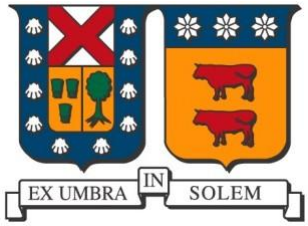
UNIVERSIDAD TÉCNICA FEDERICO SANTA MARÍA
DEPARTAMENTO DE INFORMÁTICA
VALPARAÍSO - CHILE



"IMPLEMENTACIÓN DE UN SISTEMA DE DETECCIÓN DE AMENAZAS EN TIEMPO REAL BASADO EN MACHINE LEARNING PARA ANÁLISIS DE TRÁFICO NETFLOW"

FERNANDO CAMPOS OSES

TESINA PARA OPTAR AL GRADO
DE MAGÍSTER EN TECNOLOGÍAS DE
LA INFORMACIÓN

Profesor Guía: JAVIER CAÑAS ROBLES

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 X☐ Tesis de Postgrado

Título del trabajo: Implementación de un Sistema de Detección de Amenazas en Tiempo Real basado en Machine Learning para Análisis de Tráfico NetFlow_____

Nombre del candidato(a): Fernando Campos Oses_____

Carrera / Grado: Magíster en Tecnologías de la Información_____

Campus: _____ **Departamento:** _____

2.- VALIDACIÓN DEL PROFESOR GUÍA/DIRECTOR DE TESIS

Yo, Javier Cañas Robles, en mi calidad de profesor(a) guía/director(a) del trabajo académico mencionado anteriormente **DEJO CONSTANCIA** que:

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

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

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

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

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

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

_____

_____

_____

4.- FIRMAS

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

Fecha: 2/11/2026 **Firma:** _____

Estudiante o Candidato(a):

Fecha: 03/11/2026 **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.

Implementación de un Sistema de Detección de Amenazas en Tiempo Real basado en Machine Learning para Análisis de Tráfico NetFlow

Fernando Campos Oses

Magíster en Tecnologías de la Información
Universidad Técnica Federico Santa María
frcamposo@gmail.com

Resumen: Esta tesina presenta el diseño, la implementación y la evaluación de un sistema que detecta amenazas en tiempo real analizando tráfico de red NetFlow/IPFIX, con aprendizaje automático. Una arquitectura de microservicios en contenedores captura las exportaciones de equipos de distintos fabricantes, las normaliza, las clasifica con un modelo XGBoost, las almacena en PostgreSQL y las presenta en Grafana. El modelo se entrenó con el corpus CICIDS2017, una de las referencias más citadas del área, pese a sus defectos de construcción documentados, y se evaluó en dos escenarios: fuera de línea, sobre el corpus etiquetado, y en un prototipo de laboratorio, sobre tráfico real. Sobre el corpus, tras descartar las variables que acertaban por particularidades de su construcción y no por el comportamiento del tráfico, el modelo alcanza un F1-Score de 0,968 ante familias de ataque conocidas y de 0,429 ante familias que nunca vio, aunque en este caso conserva una precisión de 0,887: detecta menos, pero se equivoca poco cuando alerta. La hipótesis se contrasta con la carga de triaje, el número de alertas que un analista debe revisar por cada amenaza real hallada, que en este escenario el modelo reduce en torno a un 70 % frente a reglas estáticas. En el prototipo, desplegado con exportadores reales, el sistema opera de extremo a extremo con una latencia de minutos; sobre ese tráfico, el 84,5 % de las alertas del clasificador fueron amenazas reales, pero alertó solo al 3,1 % de los orígenes remotos que contactaron direcciones propias sin equipos, porque el sondeo sin respuesta no está en el corpus y su señal reside en el agregado de flujos. Una segunda capa, que analiza cada origen en conjunto, eleva la cobertura al 79,5 % de los orígenes con más de veinte flujos. La hipótesis queda, por tanto, verificada sobre el corpus y pendiente sobre tráfico real, donde la comparación con las reglas no pudo medirse; tampoco se contrastó que el desempeño se mantenga entre exportadores de distintos fabricantes. El trabajo concluye que el aprendizaje automático es útil para la detección de amenazas como mecanismo de priorización, y que su límite está en la correspondencia entre los datos de entrenamiento y la telemetría observada, de donde se deriva una hoja de ruta para mejorar el modelo.

Palabras Clave: NetFlow, Machine Learning, XGBoost, Detección de Amenazas, Ciberseguridad, Reglas Estáticas, Microservicios, Docker, Carga de Triaje, Fuga de Información.

1 Introducción

1.1 Contexto y Motivación

El crecimiento del tráfico de red en entornos empresariales e institucionales transformó el panorama de la ciberseguridad durante la última década. El volumen de datos que circula sobre Internet más que se duplicó entre 2020 y 2025, con predominio del tráfico de banda ancha fija sobre el móvil en una proporción de 5:1 [1]. En paralelo se incrementó la sofisticación de las amenazas. Frente a ese escenario, los equipos de seguridad enfrentan el desafío de procesar volúmenes masivos de metadatos de tráfico para identificar comportamientos maliciosos en tiempo real.

El protocolo NetFlow, desarrollado originalmente por Cisco Systems e institucionalizado más tarde como estándar internacional a través de IPFIX (RFC 7011), provee capacidades de recolección de metadatos sobre flujos de red que constituyen la base del análisis de tráfico moderno [2]. Las aproximaciones tradicionales basadas en reglas estáticas —también llamadas detección por firmas, reglas fijas o heurística determinista— son instrucciones lógicas predefinidas del tipo «si ocurre A, entonces ejecutar B», y han sido la primera línea

de defensa. Frente a las dinámicas actuales revelan, sin embargo, limitaciones estructurales: escasa capacidad de adaptación a variantes emergentes, tasas elevadas de falsos positivos y una latencia de detección alta ante incursiones sofisticadas.

Los algoritmos de ensamble basados en árboles de decisión —en particular, para este trabajo, XGBoost (Extreme Gradient Boosting), creado por Chen y Guestrin [3]— figuran entre los de mejor desempeño sobre datos tabulares [4], y han mostrado resultados elevados en la clasificación de flujos de red sobre conjuntos de datos etiquetados de referencia [5]. A diferencia de la detección basada en firmas, que por construcción solo reconoce patrones catalogados de antemano, un clasificador aprendido puede generalizar, al menos en parte, a variantes que no vio. La disponibilidad de conjuntos de datos etiquetados, como CICIDS2017, y el aumento del poder de cómputo hicieron factible llevar estos modelos a sistemas de detección en tiempo real.

1.2 Definición del Problema

Los sistemas de monitoreo basados en NetFlow/IPFIX enfrentan dificultades significativas al implementar detección de amenazas en tiempo real. Estas dificultades, documentadas en la literatura y observables en cualquier operación de red de mediana escala, donde coexisten exportadores de distintos fabricantes, se agrupan en dos categorías: capacidad de análisis y operación.

En el plano del análisis, los sistemas procesan millones de flujos diarios sin capacidad de discriminación, y las reglas fijas no alcanzan a las amenazas no catalogadas. A ello se suma la carga que impone el volumen de falsos positivos. IDC (International Data Corporation), empresa global de inteligencia de mercado dedicada a los análisis de tecnología y telecomunicaciones, en una encuesta a 350 profesionales de centros de operaciones de seguridad, sitúa entre el 45 % y el 53 % la proporción de alertas que resultan falsas [6], coincidiendo con la medición de Yang et al. sobre 14,4 millones de alertas producidas en un Centro de Operaciones de Seguridad (SOC; Security Operations Center) universitario, donde el 48,9 % correspondió a tráfico benigno y solo el 0,01 % a ataques efectivamente exitosos [7]. El resto se reparte entre un 27 % de intentos de ataque sin consecuencias y un 24 % de ruido o falsas alarmas. En otras palabras, una alerta genuina por cada diez mil hace inviable la revisión manual exhaustiva y explica la fatiga de alertas documentada en la literatura [8].

En el plano operacional, el problema de fondo es la latencia. El informe M-Trends 2026 —publicación anual de Mandiant, división de respuesta a incidentes de Google Cloud, que sintetiza las investigaciones forenses conducidas por la firma durante el año— sitúa la mediana global de permanencia del atacante en la red en catorce días, cifra que interrumpe la tendencia decreciente sostenida desde 2011, cuando era de 416 días [9]. IBM, que emplea la media y abarca el ciclo completo del incidente, reporta 183 días para identificar una brecha y 64 para contenerla [10]. Las dos magnitudes describen fenómenos distintos y no son comparables entre sí, pero coinciden en lo esencial, que el intervalo durante el cual un atacante opera sin ser advertido se mide en semanas o meses, no en horas. Acortar esa ventana de exposición es lo que persigue un sistema de detección automatizada sobre telemetría de flujo.

1.3 Propuesta de Solución

La solución propuesta es un sistema de detección de amenazas basado en ML que integra múltiples fuentes NetFlow en una arquitectura de microservicios sobre contenedores Docker, con cinco componentes funcionales: (1) Un módulo nfcapd que recibe exportaciones NetFlow vía UDP en el puerto 9995 con ventanas de 5 minutos. (2) Un procesador Python que normaliza y extrae características. (3) Una API construida sobre FastAPI —marco de desarrollo web en Python orientado a la construcción de interfaces de programación, con validación de tipos incorporada [11]— que incluye el modelo XGBoost para la clasificación binaria de amenazas. (4) Una base de datos PostgreSQL [12] con esquema optimizado para consultas analíticas. (5) Tableros Grafana [13] conectados directamente a ella. La descomposición en servicios independientes, comunicados por interfaces declaradas y empaquetados en contenedores, sigue el patrón descrito por Newman [14] y se detalla en el Capítulo 3.

El modelo ML se entrenó con CICIDS2017 [15], del que se emplearon 2,66 millones de registros etiquetados de tráfico benigno y de múltiples familias de ataque. Como algunas familias de ataque tienen muy pocos registros, el entrenamiento incorpora, junto al preprocesamiento y la ingeniería de características, un balanceo

de clases mediante SMOTE [16], técnica que corrige el desbalance generando ejemplos sintéticos de la clase minoritaria por interpolación entre vecinos reales en lugar de duplicar observaciones.

1.4 Objetivos

El objetivo general de este trabajo es diseñar, implementar y evaluar un sistema de detección de amenazas de red que aplique aprendizaje automático supervisado, sobre telemetría de flujo NetFlow/IPFIX proveniente de múltiples exportadores heterogéneos, y demostrar cuantitativamente que dicho sistema reduce el esfuerzo de triaje que un analista de seguridad debe invertir por cada amenaza efectivamente identificada, respecto de una línea base de detección por reglas estáticas.

De ese objetivo general se desprenden seis objetivos específicos, cada uno asociado a un resultado verificable dentro del informe:

- 1.4.1. Caracterizar el protocolo NetFlow/IPFIX y sus alternativas de telemetría, y fundamentar la elección de la fuente de datos sobre la que opera el sistema (Capítulo 2).
- 1.4.2. Diseñar una arquitectura de microservicios que cubra la cadena completa de captura, normalización, inferencia, almacenamiento y visualización, incluyendo el tratamiento de la heterogeneidad entre fabricantes de exportadores (Capítulo 3).
- 1.4.3. Construir un flujo de entrenamiento reproducible para un clasificador binario sobre el set de datos CICIDS2017, incorporando ingeniería de características de dominio y tratamiento del desbalance de clases (Capítulo 4).
- 1.4.4. Verificar que el desempeño obtenido no provenga de fugas de información (data leakage, en este contexto ocurre cuando el modelo tiene acceso durante su entrenamiento a datos o información que no debería conocer o que no estará disponible en el momento de hacer predicciones reales en producción) propias del conjunto de datos, mediante controles automáticos de entrada al clasificador y un estudio de exclusión progresiva que cuantifique la contribución de cada bloque de características (Capítulos 4 y 6).
- 1.4.5. Cuantificar la hipótesis mediante una métrica operacional de obtención simple —la carga de triaje—, midiéndola sobre el mismo conjunto de prueba que las líneas base de comparación y bajo un régimen de evaluación que reproduzca la aparición de amenazas no observadas previamente (Capítulo 6).
- 1.4.6. Materializar el diseño en un prototipo de laboratorio con exportadores físicos de distintos fabricantes, y acreditar la operación de extremo a extremo de la cadena de detección (Capítulo 5).

El objetivo 1.4.4 introduce una distinción que atraviesa el informe y de la que depende la lectura del Capítulo 6, porque las características que alimentan al clasificador pueden describir dos cosas muy distintas: el comportamiento del tráfico o las particularidades del conjunto de datos.

El comportamiento del tráfico, esto es, cómo transcurre la conversación en sí, debe poder responder preguntas como la cantidad de paquetes que van y vienen, de qué tamaño, con qué regularidad en el tiempo, cuánto dura el flujo y qué proporción existe entre lo enviado y lo recibido. Son propiedades del fenómeno que se quiere detectar. Un ataque de denegación de servicio produce flujos cortos, de alta tasa y sin respuesta, y eso es cierto tanto en CICIDS2017 como en la red de cualquier organización.

Las particularidades del conjunto de datos corresponden a propiedades de cómo se fabricó esa captura en particular. Entrega información del tipo de quién generó el tráfico, desde qué máquina, en qué día y sobre qué protocolo. Correlacionan con la etiqueta dentro del archivo CICIDS2017, pero esa correlación es un accidente propio del montaje experimental y no se sostiene en otra red.

La prueba que las separa cabe en una pregunta: ¿seguiría significando lo mismo esta característica al desplegar el modelo en una red distinta? Si la respuesta es afirmativa, describe comportamiento, y, si es negativa, describe el conjunto de datos. El Capítulo 6 aplica esa prueba de manera sistemática y cuantifica cuánto del desempeño observado se apoya en cada tipo.

1.5 Hipótesis y Metodología

La hipótesis principal establece que un sistema de detección basado en aprendizaje automático, que procesa tráfico NetFlow/IPFIX proveniente de múltiples dispositivos, mejora la eficiencia operacional del análisis de

seguridad respecto de los métodos tradicionales basados en reglas estáticas. Verificarla requiere una magnitud observable, medida sobre el mismo conjunto de datos para las dos opciones que se contrastan: la detección basada en ML y el método tradicional de reglas estáticas. La métrica adoptada es la carga de triaje, definida como el cociente entre el número de eventos que el sistema presenta al analista y el número de esos eventos que resultan ser amenazas reales:

$$\text{Carga de triaje} = \text{eventos presentados} / \text{amenazas reales halladas} \quad (1)$$

La carga de triaje admite una lectura inmediata al expresar cuántos eventos debe abrir el analista de seguridad para encontrar una amenaza genuina. Una carga de 3,6 significa revisar treinta y seis casos de alertas para hallar diez amenazas reales. Un resultado de 1,1, implica revisar once para hallar diez. Su mínimo absoluto es 1, la situación ideal en que todo evento presentado resulta ser una amenaza.

La carga de triaje se eligió por tres razones. Se obtiene con la información que el propio sistema de entrenamiento del modelo ya registra —el recuento de eventos presentados y las etiquetas del conjunto de prueba—, sin requerir mediciones sobre personas. Es directamente comparable entre el sistema propuesto y cualquier alternativa evaluada sobre el mismo conjunto de flujos, porque al ser un cociente no depende del volumen de tráfico procesado, es decir, si se duplica el tráfico, se duplican tanto los eventos presentados como las amenazas contenidas en ellos, y la razón entre ambos se mantiene. Y traduce el desempeño estadístico a la unidad que gobierna la operación de un SOC, es decir, el esfuerzo humano invertido por hallazgo. Cabe señalar que la carga de triaje es el recíproco de la precisión del detector, de modo que no aporta información nueva respecto de las métricas clásicas; su valor es de interpretabilidad operacional, y es esa interpretabilidad la que permite formular un criterio difícil de refutar.

La medición principal no se realiza sobre personas ni sobre el sistema en operación, sino sobre el conjunto de prueba de CICIDS2017, fuera de línea, en el mismo cuaderno que entrena y evalúa el modelo. Ese conjunto trae la respuesta correcta de cada flujo, de modo que se puede contar con exactitud cuántos eventos presentados son amenazas reales. Conviene distinguir desde ya los dos escenarios en que el informe usa la métrica, porque se miden de manera distinta: sobre el corpus, las amenazas reales las indican las etiquetas (Capítulo 6); sobre el prototipo, donde no hay etiquetas, cada alerta se adjudica después con criterios fijados de antemano (Capítulo 5). En ambos casos el numerador son las alertas presentadas y el denominador, cuántas de ellas resultaron ser amenazas. Para la comparación se necesita, además, una línea base que reproduzca el método tradicional y se mida con la misma vara.

Esa línea base representa lo que hace hoy un equipo de seguridad que trabaja con reglas estáticas, es decir, fija de antemano condiciones sobre un IDS (Intrusion Detection System), un cortafuegos o un SIEM (Security Information and Event Management), y cada flujo que cumple alguna regla configurada se convierte en una alerta que alguien debe revisar. Cabe precisar que la solución aquí propuesta se clasifica como un NIDS (Network Intrusion Detection System), un sistema diseñado para detectar vectores de ataque mediante el monitoreo del tráfico de red. Esta aproximación difiere de la naturaleza de un SIEM, el cual opera como un repositorio centralizado enfocado en la ingesta, normalización y correlación de eventos y logs de múltiples activos de la infraestructura. La línea base se construye sobre cuatro reglas fijas —servicio sensible contactado, volumen anómalo, duración mínima y ausencia de respuesta— que alertan ante todo flujo que cumpla al menos una, según la Tabla 1. Cada regla mira un solo campo del registro NetFlow y codifica una intuición de seguridad que cualquier analista reconocería.

Tabla 1. Reglas de la línea base de detección por umbrales estáticos

Regla	Qué campo mira	Condición	Qué intuición de seguridad codifica
R1	Puerto de destino (de la 5-tupla)	Está en {22, 23, 3389, 445, 1433, 8080, 8000}	*«Alguien toca un servicio sensible»* — SSH, Telnet, escritorio remoto, compartición de archivos, base de datos
R2	Paquetes en sentido directo	Por sobre el percentil 99	*«Este flujo mueve un volumen anormal»* — inundación o transferencia masiva
R3	Duración del flujo	Por debajo del percentil 1	*«Esta conexión es demasiado breve»* — típico del sondeo que toca y se va
R4	Paquetes en sentido inverso	Igual a cero	*«Nadie responde»* — conversación unidireccional, sin respuesta del otro extremo

Las cuatro reglas se formularon para este trabajo y no reproducen un producto comercial. Cubren cuatro dimensiones del registro de flujo —la identidad del servicio, el volumen, la temporalidad y la direccionalidad—

y se apoyan en el tipo de atributos con que la detección basada en flujos aborda el sondeo y la denegación de servicio, según la revisión de Sperotto et al. [17]. La regla de ausencia de respuesta (R4) recoge, además, la observación de Jung et al. [18], quienes señalan que los intentos de conexión de un equipo que sondea la red fracasan con mucha más frecuencia que los de un equipo legítimo, porque apuntan a direcciones y servicios que no existen. Cabría objetar que la calibración de los umbrales por percentil —el percentil 99, por ejemplo, es el valor que solo supera el 1 % de los flujos— favorezca al sistema propuesto. Por eso, la evaluación compara también contra una línea base sin parámetro alguno, la revisión de la totalidad de los flujos. Las reglas R3 y R4 suponen registros bidireccionales como los del corpus; sobre la exportación unidireccional de un cortafuegos pierden su sentido, como se constatará al operar el prototipo.

Ambos métodos —reglas estáticas y modelo ML— operan sobre los mismos flujos y campos NetFlow, de modo que lo único que los distingue es cómo deciden, es decir, umbrales fijos escritos por una persona frente a una combinación aprendida de los datos. Cualquier diferencia en la carga de triaje es, por tanto, atribuible al mecanismo de decisión y no a una ventaja de partida. La hipótesis se declara verificada si, sobre un mismo conjunto de flujos de prueba, el sistema propuesto exhibe una carga de triaje estrictamente menor que la de la línea base de reglas fijas, sin reducir la cobertura de detección. En caso contrario, se declara refutada. El enunciado reúne tres afirmaciones de distinto alcance: la mejora sobre datos etiquetados, su persistencia sobre tráfico real y su independencia del exportador. El criterio solo puede aplicarse completo donde hay etiquetas, es decir, sobre el corpus; el prototipo aporta evidencia de viabilidad y de precisión, y el Capítulo 7 informa por separado el estado de cada afirmación.

Para contrastar la hipótesis se adoptó un enfoque cuantitativo experimental estructurado en cinco fases: diseño e implementación de la plataforma, entrenamiento del clasificador sobre un corpus etiquetado, verificación de la validez interna mediante controles de fuga y un estudio de exclusión progresiva, medición de la carga de triaje frente a la línea base bajo dos regímenes de exigencia creciente, y despliegue del prototipo en laboratorio con exportadores reales. Las métricas del clasificador son la precisión, la cobertura, el F1-Score y el AUC-ROC, y la métrica primaria de contrastación de la hipótesis es la carga de triaje. Para el sistema en operación se definieron además el tiempo medio de detección (MTTD, el lapso entre el inicio de un evento y su alerta) y el caudal en flujos por segundo.

1.6 Organización del Informe

El informe se organiza en siete capítulos. El Capítulo 1 fija el problema, los objetivos y la operacionalización de la hipótesis. El Capítulo 2 desarrolla el marco teórico y el estado del arte. El Capítulo 3 presenta el diseño de la solución, y el Capítulo 4, el entrenamiento del clasificador y los controles de fuga de información. El Capítulo 5 documenta el prototipo de laboratorio: lo que el clasificador logra y no logra sobre tráfico real, y la capa de detección por agregados que esas limitaciones hicieron necesaria. El Capítulo 6 evalúa el modelo sobre el corpus etiquetado y cuantifica la hipótesis, y el Capítulo 7 reúne las conclusiones —entre ellas, si el aprendizaje automático es útil para esta tarea—, las limitaciones y una hoja de ruta de trabajo futuro. Seis anexos recogen el material técnico de apoyo: campos NetFlow por fabricante (A), esquema de base de datos (B), pipeline de entrenamiento (C), captura multi-dispositivo (D), implementación del prototipo (E) y acrónimos (F). Cada capítulo indica al comienzo cuál de ellos amplía su contenido, de modo que el cuerpo pueda leerse de corrido.

2 Marco Teórico y Estado del Arte

Este capítulo reúne los conceptos sobre los que descansa el trabajo: qué es NetFlow y qué permite observar, por qué se eligió frente a otras fuentes de telemetría, qué familia de algoritmos se aplica y por qué XGBoost, qué exige capturar desde varios dispositivos y dónde se sitúa la propuesta frente al mercado y la investigación. Quien desee profundizar encontrará en el Anexo A la evolución del protocolo, los campos por fabricante, el detalle de las alternativas y la revisión de productos; en el Anexo C, la taxonomía de algoritmos y la notación de XGBoost, y en el Anexo D, los desafíos de la captura multi-dispositivo.

2.1 Protocolo NetFlow y Análisis de Tráfico de Red

NetFlow es un protocolo desarrollado por Cisco Systems en 1996 que recolecta información estadística sobre los flujos IP que atraviesan las interfaces de routers, switches y firewalls. Cada uno de estos equipos cumple el rol de exportador de NetFlow hacia un servidor que actúa como colector, donde se almacena y analiza la data. Un flujo se define como una secuencia unidireccional de paquetes que comparte una séptupla de atributos: direcciones IP de origen y destino, puertos de origen y destino, protocolo, interfaz de entrada y tipo de servicio [2], [19]. Por cada flujo activo el exportador mantiene contadores de bytes, paquetes y marcas temporales, que envía al colector al expirar el flujo o al alcanzar el temporizador configurado.

El tráfico NetFlow es un «registro de flujo», y su diferencia con una captura íntegra de los paquetes admite una comparación con el detalle de una factura telefónica. Dice quién llamó a quién, a qué hora, cuánto duró y cuánto se transfirió, pero no qué se dijo. Esa renuncia deliberada al contenido es la que hace viable observar una red completa con un costo de almacenamiento acotado, y la que fija a la vez el techo de lo que el sistema puede detectar.

La evolución del protocolo admite un resumen en cuatro etapas. La versión 1, liberada en 1996, fue un prototipo limitado a IPv4. La versión 5 fijó el formato de registro de cuarenta y ocho bytes que la industria adoptó como referencia y que sigue vigente en buena parte del equipamiento heredado. El cambio cualitativo llegó con la versión 9, documentada por la IETF en el RFC 3954 [19], que sustituyó los campos fijos por plantillas dinámicas en que el exportador declara qué campos envía y en qué orden, de modo que el formato deja de estar congelado. IPFIX (RFC 7011) [20] estandarizó ese esquema de plantillas como norma abierta e independiente del fabricante.

2.1.1 Campos Relevantes para Detección de Amenazas

El valor de un registro de flujo no reside en ningún campo aislado, sino en la lectura conjunta de cinco familias de atributos, a saber: la temporalidad, que revela el ritmo de la comunicación y con él la diferencia entre software automatizado y actividad humana; el volumen y direccionalidad, que distinguen una sesión conversacional de una comunicación unidireccional, propia del barrido y de la exfiltración; las banderas (flags) de control TCP, que separan las conexiones legítimas de los sondeos; la textura —tiempo entre llegadas y dispersión de tamaños—, donde la regularidad delata al proceso automático; y las métricas derivadas, que condensan relaciones que los campos individuales solo expresan por separado y son por ello las de mayor poder discriminativo.

2.2 Análisis de Alternativas y Justificación de la Elección de NetFlow/IPFIX

La exportación de flujos no es la única fuente de telemetría disponible para alimentar un sistema de detección de amenazas. Antes de comprometer el diseño con NetFlow/IPFIX, se evaluaron otras alternativas técnicamente viables, incluidas dos que ofrecen capacidades funcionales superiores en dimensiones específicas. La elección final no se apoya, por tanto, en la superioridad absoluta del protocolo escogido, sino en su adecuación a las restricciones y objetivos concretos de este trabajo.

El análisis se estructura sobre cinco criterios derivados de los requisitos del problema planteado en el Capítulo 1. El primer factor es la cobertura de infraestructura heterogénea, entendida como disponibilidad en equipos de distintos fabricantes y generaciones sin obligar a renovar el parque instalado. El segundo, la suficiencia informativa para el modelo, esto es, entrega de los atributos de volumen, duración, direccionalidad y temporalidad que el clasificador requiere sin acceder al contenido del paquete IP. Luego, el costo de adquisición y operación, relevante para la transferibilidad de la solución a organizaciones con presupuesto acotado. También, se debe considerar el impacto sobre el equipamiento productivo, que la instrumentación no debe degradar, y, finalmente, las implicancias legales y de privacidad, que exigen que la recolección sea proporcional al fin perseguido.

Frente a esos cinco criterios se evaluaron cuatro alternativas viables: sFlow, la telemetría de streaming, la inspección profunda de paquetes (DPI; Deep Packet Inspection) y la captura íntegra en PCAP. Ninguna de ellas supera a NetFlow/IPFIX en el conjunto de los criterios, por las razones que siguen.

2.2.1 Decisión Adoptada

Ninguna de las cuatro alternativas domina a las restantes en todos los criterios y cada una resuelve bien un problema distinto. La telemetría de streaming ofrece latencia muy inferior, pero observa el equipo y no la conversación. DPI entrega información incomparablemente más rica, a costa de un presupuesto, un desempeño y una exposición legal que el problema no justifica. sFlow iguala a NetFlow/IPFIX en costo y cobertura, pero sacrifica precisamente los contadores por flujo de los que depende el clasificador. La captura completa (PCAP) maximiza la fidelidad y resulta inviable como régimen permanente.

NetFlow/IPFIX se impone porque satisface los cinco criterios, aunque no sea el mejor en cada uno, y porque el atributo en que sí resulta superior —su disponibilidad universal sobre infraestructura ya instalada— es justamente el que la hipótesis pone a prueba, es decir, un sistema que exigiera renovar el equipamiento tendría escaso valor para las organizaciones que se enfrentan a la definición del problema del Capítulo 1. A ello se añade que los corpus etiquetados que permiten evaluar de manera reproducible están construidos sobre atributos de flujo, con un esfuerzo consolidado por estandarizarlos precisamente sobre NetFlow [5].

La limitación asumida debe declararse con claridad al operar sobre metadatos, y es que el sistema es estructuralmente incapaz de identificar amenazas cuya firma resida en el contenido, y no distingue aplicaciones que enmascaran su puerto. La apuesta del trabajo consiste en que el comportamiento observable de un flujo —su ritmo, su volumen y su asimetría— contiene señal suficiente para discriminar tráfico malicioso sin abrir el paquete, y los Capítulos 5 y 6 permiten evaluar hasta qué punto se sostiene.

2.3 Machine Learning en Ciberseguridad

El aprendizaje automático agrupa familias que difieren en cómo aprenden de los datos: supervisado, no supervisado, de aprendizaje profundo, y por refuerzo.

Supervisado, cuando se dispone de ejemplos etiquetados con la respuesta correcta; no supervisado, cuando el algoritmo debe descubrir por sí mismo la estructura de los datos; profundo, cuando emplea redes neuronales de múltiples capas; y por refuerzo, cuando aprende mediante recompensas en un entorno dinámico. Aplicadas al análisis de tráfico NetFlow/IPFIX, las tres primeras dan lugar a enfoques complementarios. Por ejemplo, la clasificación supervisada reconoce ataques cuyo comportamiento fue observado antes, en cambio, la detección de anomalías no supervisada —Isolation Forest [21], One-Class SVM— señala desviaciones respecto de una línea base sin necesidad de etiquetas, y los autocodificadores profundos abordan patrones de mayor complejidad a costa de interpretabilidad (el problema de la caja negra) y de volumen de datos.

Los algoritmos de clasificación supervisada han demostrado eficacia en la detección de intrusiones sobre tráfico de red. Por ejemplo, Random Forest resiste bien el sobreajuste, aunque no modela por sí mismo la secuencia temporal de los flujos; las máquinas de vectores de soporte (SVM; Support Vector Machine) son eficientes con pocos datos, pero escalan mal ante conjuntos masivos, y las redes profundas exigen mucho más volumen etiquetado porque de lo contrario se facilita el sobreajuste. Sobre datos tabulares como los registros de flujo la evidencia sitúa a los métodos de ensamble basados en árboles de decisiones por delante en la relación entre exactitud y costo computacional [4], con la cautela que imponen las limitaciones documentadas de los sets de datos de referencia [22], [23].

2.3.1 XGBoost: Gradient Boosting Extremo

El clasificador utiliza como motor principal XGBoost [3], un método basado en la potenciación (boosting). Su funcionamiento se centra en entrenar secuencialmente modelos débiles, que son árboles de decisión individuales tan sencillos que por sí solos apenas superan el acierto aleatorio. El algoritmo corrige los errores de predicción del sistema de manera progresiva mediante un ensamble aditivo; esto significa que, en lugar de cambiar lo que ya se ha construido, se van sumando nuevos árboles especializados exclusivamente en resolver las fallas de los árboles anteriores, logrando una predicción final robusta. Se distingue así del agregado por remuestreo (bagging) de Random Forest, donde los submodelos se entrenan en paralelo y la predicción final es un promedio.

La contribución distintiva de XGBoost respecto de la potenciación clásica de Friedman [24] está en cómo decide cuán complejo puede llegar a ser cada árbol. Un modelo al que se permite crecer sin límite acaba memorizando el conjunto de entrenamiento —incluidas sus peculiaridades y su ruido— y falla ante datos nuevos, fenómeno que se conoce como sobreajuste. XGBoost lo evita cobrando un peaje por cada bifurcación nueva, de modo que el árbol solo incorpora una distinción si el beneficio en exactitud compensa ese costo.

Formalmente, en cada iteración el algoritmo busca minimizar una cantidad (función objetivo) que puede leerse en términos conceptuales como la suma de dos penalizaciones contradictorias entre sí: el error de predicción, qué tan bien se ajusta a los datos, y la complejidad del modelo, qué tan grande o enredado es el árbol de decisión. En otras palabras, si se hace el árbol muy simple, el costo por complejidad es bajo, pero el error de predicción será alto (el modelo no aprende lo suficiente, incurriendo en un subajuste o underfitting). Si se hace el árbol muy complejo, el error de predicción baja a cero, pero el costo por complejidad se dispara (generando un sobreajuste o overfitting). Por tanto, cada iteración minimiza una función objetivo que se lee como la suma de dos fuerzas opuestas:

$$\text{Objetivo} = (\text{error de predicción}) + (\text{costo por complejidad del modelo}) \quad (2)$$

Escrita con la notación habitual de la literatura, esa misma expresión adopta la forma:

$$\text{Obj} = \Sigma_i \, l(y_i, \hat{y}_i) \; + \; \Sigma_k \, \Omega(f_k) \quad (3)$$

El primer bloque corresponde a la función de pérdida “l” (Loss Function), y es el marcador de errores. Mide qué tan lejos está la predicción del algoritmo de la realidad. y_i es el valor real e $\hat{y}_i$ es la predicción que hizo el algoritmo. Por tanto, $l(y_i, \hat{y}_i)$ es la función de pérdida que simplemente calcula la distancia o la "penalización" por cada error cometido, y que idealmente debe tender a cero.

El segundo bloque sería el de la regularización, es decir, el freno que penaliza la complejidad. Mide qué tan grandes y complejos son los árboles de decisión que se suman al modelo. f_k representa a un árbol individual y Ω es una función matemática que castiga al árbol si se vuelve muy "salvaje". Lo penaliza si tiene demasiadas ramas (nodos) o si los valores que asigna en sus hojas son exageradamente altos o extremos. En este caso, el modelo quiere que esta suma también sea baja. Al obligar a que los árboles sean sencillos y moderados, se asegura de que el modelo final no memorice el ruido de los datos (evita el sobreajuste).

El primer término empuja al modelo a ser preciso y el segundo a mantenerse simple, de modo que el entrenamiento consiste en hallar el equilibrio entre ambos.

Dos propiedades inclinaron la elección hacia XGBoost. La primera, su comportamiento con clases desbalanceadas, condición estructural del tráfico de red, donde la clase maliciosa es normalmente minoritaria. La segunda, que expone la importancia de cada característica, lo que permite justificar una alerta ante el analista —requisito de adopción operacional según Capuano et al. [25]— y auditar el propio modelo para determinar de dónde proviene su desempeño, como hace el Capítulo 6.

2.4 Elección de Arquitecturas de Microservicios para Análisis en Tiempo Real

Las arquitecturas de microservicios descomponen una aplicación en servicios independientes, de responsabilidad acotada y desplegables por separado, comunicados mediante interfaces de programación (APIs) bien definidas [14]. Para sistemas de análisis de seguridad en tiempo real, esta arquitectura ofrece escalabilidad independiente por componente, tolerancia a fallos por aislamiento de servicios, facilidad de actualización del modelo ML sin detener el sistema completo, y flexibilidad para integrar nuevas fuentes de datos mediante adaptadores específicos. La containerización mediante Docker y Docker Compose garantizó la reproducibilidad del entorno de ejecución y la portabilidad entre entornos de desarrollo y producción.

2.5 Captura Multi-Dispositivo en Redes Empresariales

En una red real la telemetría no proviene de un único dispositivo. Es como vigilar un edificio con una sola cámara en la puerta de entrada: permite saber quién entra y quién sale, pero no quién va de piso en piso. En

consecuencia, la instrumentación exclusiva del router de borde mantiene un punto ciego en el monitoreo de la red, puesto que solo registra el tráfico que entra o sale del entorno observado. De este modo, queda fuera de análisis el tráfico que transita entre segmentos internos, el cual es especialmente relevante para identificar actividades de reconocimiento interno por parte de un atacante y sus movimientos laterales para desplazarse desde el equipo inicialmente comprometido hacia otros sistemas de mayor valor. La captura multi-dispositivo persigue por ello ampliar la cobertura sobre el tráfico interno y elevar la confianza de las detecciones mediante la corroboración de un mismo evento observado desde varios puntos, siempre que esa multiplicidad se trate como evidencia y no como volumen.

La estrategia introduce cuatro desafíos previos a que los flujos alimenten al modelo de aprendizaje automático. Una necesaria normalización de formatos, porque cada fabricante implementa el estándar con matices propios y el modelo espera un vector de estructura fija. La sincronización temporal, pues una deriva de pocos segundos basta para que dos observaciones del mismo flujo se interpreten como eventos distintos. La deduplicación, porque un flujo que atraviesa tres exportadores se exporta tres veces y sin tratarlo el sistema multiplicaría las alertas de un único evento. Y el muestreo diferencial —equipos de alta capacidad pueden exportar la totalidad de los flujos o solo una muestra estadística— que vuelve incomparables los contadores y degrada selectivamente la señal de baja intensidad, justo la de las fases tempranas de un compromiso de amenaza.

Tabla 2. Desafíos de la captura multi-dispositivo y estrategias de mitigación

Desafío	Manifestación	Estrategia de mitigación propuesta
Heterogeneidad de formatos	Campos ausentes o con distinta semántica según fabricante	Capa de normalización a esquema común con imputación de campos ausentes
Duplicación de flujos	Un flujo exportado por cada salto de su trayectoria	Clave de identidad de flujo + ventana de tolerancia de 2 segundos
Deriva temporal entre relojes	Correlación errónea o pérdida de correspondencia entre observaciones	Sincronización NTP obligatoria en todos los exportadores
Muestreo diferencial	Contadores no comparables entre dispositivos	Escalado por factor de muestreo declarado por exportador
Timestamps relativos	sysUpTime en lugar de tiempo absoluto	Reconstrucción desde la cabecera del paquete de exportación
Volumen agregado	La suma de exportadores satura el colector	Ventanas de 5 min y procesamiento por lotes de 1000 flujos

La Tabla 2 sintetiza esos desafíos junto a la estrategia de mitigación adoptada para cada uno. Su resolución es condición previa para que la fusión de fuentes aporte valor en lugar de introducir ruido, y constituye parte sustantiva del trabajo de ingeniería del Capítulo 3.

2.6 Estado del Arte

2.6.1 Soluciones Comerciales

El mercado de la detección de amenazas sobre telemetría de flujo se organiza en tres segmentos que responden a problemas distintos y se comparan mal entre sí. El primero lo ocupan las plataformas anti-DDoS de escala de operador de telecomunicaciones —NETSCOUT Arbor Sightline [26], Nokia Deepfield Defender [27], GenieATM [28], Radware DefenseFlow [29]—, que comparten tres rasgos delimitantes: la escala, pues están dimensionadas para terabits por segundo y operan sobre telemetría muestreada y agregados por objeto protegido antes que sobre el flujo individual; el foco, centrado en la denegación de servicio volumétrica, cuya firma es una desviación abrupta del volumen y no la clasificación fina de familias heterogéneas; y la opacidad, porque ninguno publica sus algoritmos y el operador no puede inspeccionar ni sustituir el modelo. Su licenciamiento por capacidad protegida, con hardware dedicado, las sitúa fuera del alcance de la organización mediana.

El segundo segmento reúne plataformas de visibilidad que incorporan detección como una función entre otras. SolarWinds NPM con detección mediante umbrales estadísticos. Cisco Secure Network Analytics con

aprendizaje no supervisado restringido a su propio ecosistema. Plixer Scrutinizer con agrupamiento y soporte multi-fabricante. Y, Kentik, que lleva el modelo a la nube con licenciamiento más accesible a costa de enviar la telemetría de la organización a otro proveedor.

El tercer segmento es el del código abierto. FastNetMon [30] y ntopng son gratuitos y auditables, pero su detección es la de umbrales fijos que este trabajo toma precisamente como línea base. SiLK [31], del CERT de la Universidad Carnegie Mellon, ocupa un lugar distinto: es una infraestructura de recolección, almacenamiento y consulta de flujos, orientada al análisis retrospectivo que un analista conduce sobre grandes volúmenes, y su componente de análisis continuo detecta mediante reglas, listas de vigilancia y heurísticas. Ninguno incorpora un clasificador supervisado que discrimine sobre el flujo individual. SiLK, más que una alternativa a la propuesta, lo es a su capa de captura y almacenamiento, para la cual se prefirió nfcapd por su menor costo de configuración. Cabe notar que SiLK exige declarar los bloques de direcciones internas para clasificar el tráfico, la misma necesidad de conocimiento local que este trabajo formula en el Capítulo 5.

La lectura de conjunto sitúa este trabajo con precisión. No compite con las plataformas de escala de operador. Su objetivo de procesamiento está cuatro o cinco órdenes de magnitud por debajo, y un clasificador que evalúa cada flujo no es la herramienta adecuada para el tráfico agregado de un operador. El espacio que ocupa es el de la organización mediana que ya dispone de exportadores NetFlow, no puede costear licencias por capacidad, necesita discriminar familias diversas sobre su propio tráfico y exige poder auditar la detección. Allí las alternativas comerciales resultan sobredimensionadas y opacas, y las de código abierto reproducen justamente la detección por umbrales cuya limitación este trabajo cuantifica.

2.6.2 Investigación Académica Relevante

La literatura sobre detección basada en flujos ofrece cuatro aportes especialmente pertinentes. Hofstede et al. [2] proporcionan el tratamiento integral de referencia sobre todas las etapas de una instalación de monitoreo y documentan las configuraciones con múltiples puntos de observación, marco sobre el que se apoya el diseño del Capítulo 3. En el plano de los datos, Ring et al. [23] relevan sistemáticamente los corpus disponibles y proponen quince propiedades para juzgar su idoneidad, lo que fundamenta la elección de CICIDS2017 y a la vez advierte sobre sus limitaciones. Panigrahi y Borah [22] las examinan en detalle e identifican deficiencias suficientes para sesgar un sistema entrenado sin precauciones, y Sarhan et al. [5], [32] proponen un conjunto estándar de características basado en NetFlow para resolver la no comparabilidad entre estudios.

En el plano metodológico, Sommer y Paxson [33] explican la brecha entre la abundancia de investigación académica sobre detección de anomalías y su escasa adopción operacional. La detección de ataques difiere estructuralmente de otros problemas de clasificación por el costo asimétrico de los errores y por la dificultad de definir la normalidad, argumento que justifica la cautela con que este trabajo interpreta sus propias métricas. Arp et al. [34] catalogan diez trampas metodológicas recurrentes —entre ellas la fuga de información— y sus recomendaciones se incorporaron como criterio de diseño, en particular en la exclusión de las direcciones IP. Y Capuano et al. [25] argumentan que justificar una alerta ante el analista es requisito de adopción y no un refinamiento opcional, lo que respalda la preferencia por un modelo que expone importancias de características.

2.7 Diferenciación de este Trabajo

Respecto de la investigación académica previa, habitualmente limitada a evaluaciones fuera de línea sobre corpus estáticos, este trabajo aporta una plataforma operativa y reproducible, desplegable mediante Docker Compose y agnóstica al fabricante del exportador, verificada de extremo a extremo con exportadores físicos heterogéneos. Respecto de las plataformas comerciales, opera sobre el flujo individual sin muestreo, con un clasificador inspeccionable y sustituible, y sin licencias ni hardware dedicado. A cambio, renuncia deliberadamente a la escala de operador y a la mitigación automática. Y respecto de las herramientas de código abierto, reemplaza la detección por umbrales fijos por una combinación aprendida de los datos, cuya ventaja es precisamente lo que la hipótesis cuantifica.

3 Diseño de la Solución

Este capítulo describe la solución como diseño, con independencia de cómo se implementó: qué componentes la forman, cómo circula un registro de flujo entre ellos, qué datos se conservan y cómo se integra la telemetría de varios exportadores. Lo efectivamente construido, y lo que su operación enseñó, se trata en el Capítulo 5. El Anexo B detalla el esquema de la base de datos; el Anexo D, la especificación de la captura multi-dispositivo, y el Anexo E, los parámetros de configuración de cada componente.

3.1 Visión General de la Arquitectura

El sistema se concibe como una plataforma de microservicios en contenedores, orquestados mediante Docker Compose. La elección responde a tres criterios: reproducibilidad del despliegue, independencia entre el ciclo de vida del modelo y el de la infraestructura, y posibilidad de sustituir un componente sin afectar al resto de la cadena.

La arquitectura se organiza en tres capas funcionales. La de ingesta recibe la telemetría de los exportadores y la almacena de forma intermedia (nfcapd). La de procesamiento e inferencia la normaliza a un esquema común, la clasifica con XGBoost y guarda el resultado (procesador de flujos, servicio FastAPI y PostgreSQL). La de presentación expone los resultados al operador y vigila la salud de la plataforma (Grafana y Prometheus). Un proxy inverso Nginx unifica el acceso externo, de modo que ningún servicio interno queda expuesto directamente. Todos los servicios comparten una red Docker dedicada con direcciones estáticas, que separa el tráfico de gestión del tráfico analizado. La Figura 1 representa esa disposición.

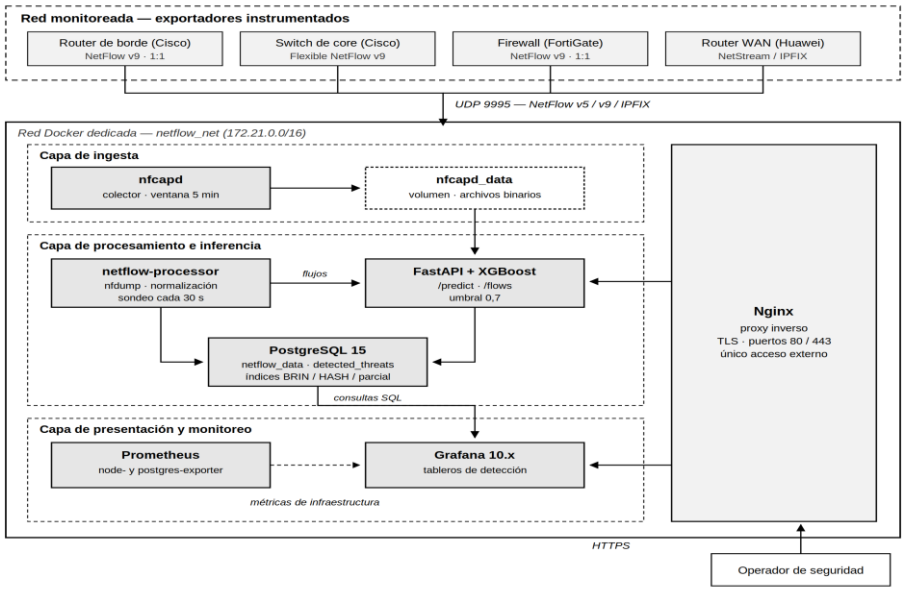


Figura 1. Arquitectura de componentes de la plataforma, organizada en tres capas funcionales sobre una red Docker dedicada.

3.2 Flujo de Datos del Sistema

La Figura 2 representa las cinco etapas que recorre un registro de flujo desde su exportación hasta su presentación.

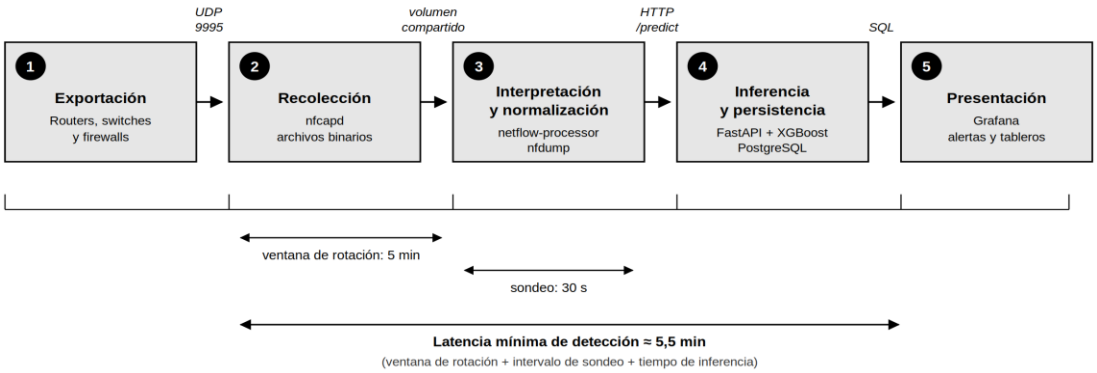


Figura 2. Flujo de datos del sistema en cinco etapas, con las interfaces entre etapas y la composición de la latencia mínima de detección.

La latencia de detección es la suma de tres esperas: el cierre del archivo del colector, que rota cada cinco minutos; el sondeo del procesador, cada treinta segundos, y el tiempo de inferencia, despreciable frente a las anteriores. Sin contar los temporizadores del propio exportador, el peor caso ronda los cinco minutos y medio; en promedio, un flujo espera media ventana de rotación más los temporizadores del exportador. Reducirla exige acortar la ventana de rotación, a costa de procesar con más frecuencia.

3.3 Descripción de los Componentes

La cadena la forman los equipos exportadores y cinco componentes de software. Los dispositivos de la red —routers, conmutadores y cortafuegos— exportan sus flujos por UDP al puerto 9995, en NetFlow v5, v9 o IPFIX según su generación y fabricante. El colector nfcapd los recibe y cierra un archivo cada trescientos segundos. El procesador de flujos, escrito en Python, decodifica con nfdump cada archivo cerrado y, sobre cada registro, proyecta los campos al esquema común, construye el identificador de conexión y calcula las características derivadas. El servicio de inferencia, sobre FastAPI [11], carga una sola vez el pipeline entrenado y puntúa cada flujo. PostgreSQL [12] conserva el registro completo junto a su puntuación, y Grafana [13] y Prometheus [35] mantienen separados el plano de detección y el de salud de la plataforma.

Dos decisiones de diseño condicionan la lectura de los resultados. La primera es que la inferencia devuelve para cada flujo una puntuación de amenaza, el threat_score, entre 0 y 1, y no un veredicto binario: la medida continua permite ordenar los flujos por prioridad de revisión, examinar la franja intermedia donde se concentran los casos ambiguos y reinterpretar una decisión sin reprocesar el tráfico. La segunda es que convertir esa puntuación en alerta es un acto separado, gobernado por un umbral configurable, fijado en 0,7, por sobre el punto neutro de 0,5, para privilegiar la precisión. Como todo flujo se conserva con su puntuación, elevar el umbral saca un evento de la cola de alertas, pero no lo elimina.

Conviene explicitar, por último, una distinción que se presta a confusión: lo que el modelo recibe como entrada y lo que el sistema registra obedecen a criterios opuestos. El clasificador no recibe las direcciones IP, porque podría memorizar la identidad de los equipos en lugar de aprender su comportamiento. La base de datos, en cambio, conserva la 5-tupla completa junto a la puntuación, porque una alerta que no dice quién habla con quién no le sirve al analista. La restricción pertenece al modelo, no al registro.

3.4 Diseño del Esquema de Datos

El esquema atiende dos exigencias que no siempre coinciden: representar fielmente el registro tal como lo entrega el exportador y disponer de las características que el modelo requiere sin recalcularlas en cada consulta. La tabla de flujos incorpora por ello, junto a los campos del registro, columnas que corresponden directamente

a las características del clasificador. Esa redundancia controlada permite reconstruir después el vector exacto sobre el que se emitió cada predicción, condición necesaria para auditar las alertas y para reentrenar con datos propios. Una segunda tabla registra las amenazas con su tipo, severidad y un campo de detalles en formato JSON, elegido porque los metadatos relevantes varían según el tipo de amenaza.

3.5 Arquitectura de Captura Multi-Dispositivo

La captura multi-dispositivo se diseñó con un colector centralizado único: todos los exportadores envían su telemetría al mismo puerto 9995/UDP, sin agregadores por fabricante. Un colector por familia de dispositivos simplificaría cada componente, pero multiplicaría los puntos de configuración y de falla; el colector único concentra, en cambio, el tratamiento de la heterogeneidad en la capa de normalización, que es software propio y puede modificarse sin intervenir los equipos de red.

3.5.1 Topología de Captura y Correlación entre Exportadores

El diseño ubica exportadores en puntos complementarios para observar tanto el tráfico que cruza el perímetro de la red como el que circula solo en su interior, entre segmentos internos. Incluir equipos de distintos fabricantes es parte del diseño, porque obliga a que la normalización sea independiente del proveedor. La topología condiciona además la interpretación de las direcciones: con traducción de direcciones (NAT, el mecanismo por el cual muchos equipos internos salen a Internet con una misma dirección pública), una dirección observada en el perímetro puede representar a decenas de equipos. Por eso se prioriza observar la interfaz interna, y las direcciones perimetrales se tratan como un atributo declarado, no como algo que se infiere del rango observado.

La correlación entre exportadores se estructura en cuatro fases encadenadas. La identificación etiqueta cada registro con el exportador que lo envió. La normalización unifica nombres de campos, escala contadores, lleva las marcas temporales a tiempo universal e imputa los campos ausentes. La deduplicación agrupa en un registro canónico las observaciones de un mismo flujo hechas por varios equipos, conservando cuántos lo vieron. Y la validación cruzada ajusta la severidad de una alerta según cuántos dispositivos coinciden, sin alterar la puntuación original, que se preserva para auditoría y reentrenamiento.

Como el sistema concentra telemetría que expone la topología lógica de la organización, su propia seguridad descansa en tres pilares alineados con el NIST Cybersecurity Framework [36]: segmentación, con una red Docker aislada que solo recibe de los exportadores por el puerto 9995/UDP; minimización, porque almacena solo metadatos y descarta el contenido, y retención acotada, de noventa días para los registros individuales y un año para los agregados y las amenazas confirmadas, aplicada automáticamente por funciones de la base de datos.

4 Modelo de Aprendizaje Automático: Entrenamiento y Validación

Este capítulo describe el componente que emite el juicio sobre la telemetría, en el orden en que se construyó: el corpus de entrenamiento y sus limitaciones, el preprocesamiento, el tratamiento del desbalance, la ingeniería de características, el pipeline de entrenamiento y la prevención de la fuga de información. Los hiperparámetros, la descripción detallada de los transformadores y el procedimiento reproducible se recogen en el Anexo C.

4.1 Dataset de Entrenamiento: CICIDS2017

El modelo se entrenó sobre CICIDS2017 [15], del Canadian Institute for Cybersecurity. El corpus reúne unos 2,8 millones de registros de flujo, capturados durante cinco jornadas laborales en una red que reproduce el tráfico de una organización, y etiquetados como benignos o como alguno de catorce tipos de ataque. El archivo de la mañana del jueves, que contiene los tres ataques web, se excluyó por inconsistencias de codificación en

el campo de etiqueta. Los siete archivos restantes suman 2.660.377 registros, de los cuales 555.466, el 20,88 %, son maliciosos y corresponden a once tipos de ataque. La Tabla 3 muestra cómo se reparten.

Tabla 3. Distribución del tráfico malicioso por jornada y tipo de ataque en los siete archivos empleados de CICIDS2017

Jornada	Tipo de ataque	Flujos	% del tráfico malicioso
Martes	FTP-Patator (fuerza bruta)	7.938	1,43 %
Martes	SSH-Patator (fuerza bruta)	5.897	1,06 %
Miércoles	DoS Hulk	231.073	41,60 %
Miércoles	DoS GoldenEye	10.293	1,85 %
Miércoles	DoS Slowloris	5.796	1,04 %
Miércoles	DoS Slowhttptest	5.499	0,99 %
Miércoles	Heartbleed	11	< 0,01 %
Jueves (tarde)	Infiltration	36	< 0,01 %
Viernes	DDoS	128.027	23,05 %
Viernes	PortScan (sondeo de puertos)	158.930	28,61 %
Viernes	Bot (botnet)	1.966	0,35 %
Total	11 tipos	555.466	100,00 %

La distribución está lejos de ser uniforme. Tres tipos —la denegación de servicio Hulk, el sondeo de puertos y la denegación de servicio distribuida— reúnen el 93,3 % de los flujos maliciosos, mientras la infiltración y Heartbleed suman apenas 47 flujos. El desbalance no es solo entre tráfico benigno y malicioso, sino también entre familias, y por eso las métricas globales deben acompañarse de un desglose por familia. Además, el diseño experimental tiene una propiedad determinante para el Capítulo 6: como cada familia de ataque se concentra en una sola jornada, cualquier partición cronológica del corpus equivale a evaluar con familias de ataque no observadas durante el entrenamiento.

4.1.1 Fundamento de la Elección y Limitaciones Conocidas del Corpus

La elección de CICIDS2017 es deliberada y conviene fundarla, porque el corpus no es reciente y su construcción tiene defectos documentados. Su principal ventaja es la comparabilidad: el artículo que lo presenta supera holgadamente las dos mil citas, de modo que un resultado obtenido sobre él puede contrastarse con una literatura amplia. Se suman cuatro propiedades que este trabajo aprovecha: el tráfico benigno lo generan perfiles que imitan a usuarios reales sobre los protocolos habituales de una organización; las amenazas son diversas; se publican las capturas de paquetes completas, lo que permite reprocesar el corpus con otro extractor de flujos, y el calendario documentado del experimento hace posible la evaluación con familias no observadas.

Sus limitaciones también están documentadas. Engelen et al. [37] mostraron que la herramienta que extrajo los flujos, CICFlowMeter, da por cerrada una conexión TCP al primer paquete FIN (solicitud de término de la conexión), cuando el protocolo solo la termina tras el intercambio de FIN y ACK de ambos extremos; los paquetes restantes forman flujos residuales que heredan la etiqueta del original y que, según esos autores, constituyen cerca de un cuarto del corpus. Señalaron además intentos de ataque sin efecto etiquetados como ataque y la presencia de identificadores —direcciones, puertos y marcas de tiempo— que invitan al modelo a tomar atajos. Lanvin et al. [38] añadieron paquetes duplicados o fuera de orden y ataques que quedaron sin etiquetar, y Panigrahi y Borah [22] documentaron el fuerte desbalance entre clases.

Este trabajo no corrige el corpus, pero se protege de sus defectos de tres maneras: un estudio de exclusión progresiva diseñado para descubrir lo que el modelo aprende de los artefactos de construcción, el reporte de las cifras sobre el corpus como cotas y no como desempeño esperable en otra red, y la medición del modelo sobre tráfico real con instrumentos que no dependen de las etiquetas del corpus. Un defecto conocido y acotado es preferible a uno ignorado.

4.2 Preprocesamiento y Limpieza de Datos

El preprocesamiento comprendió tres etapas. La primera reemplazó por NaN (Not a Number) los valores infinitos presentes en el corpus, originados en divisiones por cero al computar tasas. La segunda imputó los faltantes mediante la mediana de cada columna, preferida sobre la media por su robustez frente a las distribuciones de cola larga del tráfico de red, en las que unos pocos flujos de gran volumen desplazarían la media hacia valores no representativos. La tercera excluyó las columnas de varianza nula o con problemas sistemáticos de formato, que no aportaban capacidad discriminativa.

4.3 Mapeo de Etiquetas y Balance de Clases

Las etiquetas originales se convirtieron a una clasificación binaria, 0 para el tráfico benigno y 1 para cualquier ataque, porque el objetivo operacional es distinguir lo que requiere atención de lo que no. La etiqueta original se conserva junto a cada registro para poder descomponer la cobertura por familia.

El desbalance se abordó con SMOTE [16], que genera ejemplos sintéticos de la clase minoritaria interpolando entre vecinos reales en lugar de duplicar observaciones: la duplicación simple induce al modelo a memorizar ejemplos repetidos, mientras la interpolación introduce variabilidad. SMOTE se aplica como una etapa del pipeline y solo durante el entrenamiento, por la razón metodológica que se explica al describir el pipeline.

4.4 Ingeniería de Características

La ingeniería de características transforma los datos crudos en variables que incorporan conocimiento del dominio y resaltan lo que el modelo debe aprender. El modelo parte de doce campos numéricos del registro de flujo, tal como los define CICIDS2017: la duración, los paquetes y bytes en cada sentido, el tamaño medio de paquete por sentido, los tiempos medios entre llegadas de paquetes (global y por sentido), la desviación estándar del tamaño en sentido directo y el protocolo. Las direcciones y los puertos no ingresan como campos: viajan en un identificador de conexión que solo sirve de insumo para derivar características y que se elimina antes del clasificador. Sobre esa base se derivan trece características más, con lo que el espacio de entrada suma veinticinco; cinco de ellas se descartan en el Capítulo 6 por describir artefactos del corpus, de modo que el modelo desplegado usa veinte. Cabe advertir que varios de esos campos —los tiempos entre llegadas y la dispersión de tamaños— provienen de la observación de cada paquete y no tienen equivalente directo en NetFlow v9, que exporta cada sentido por separado y solo con totales; en operación se aproximan o se imputan, una diferencia que el Capítulo 5 examina.

Las derivaciones las ejecutan dos transformadores propios, construidos con la interfaz de scikit-learn [39] para integrarse al pipeline y serializarse junto al clasificador, lo que garantiza que el entrenamiento y la inferencia transformen los datos exactamente igual. El primero descompone el identificador de conexión para indicar si las direcciones son privadas, si los puertos pertenecen a un conjunto de servicios sensibles —administración remota, bases de datos y compartición de archivos— y cuál es la diferencia entre ambos puertos; al terminar, elimina las direcciones para que no lleguen al clasificador. El segundo calcula razones de comportamiento —bytes por paquete, proporción entre lo enviado y lo recibido, variabilidad del tamaño y tasas por unidad de tiempo— y tres atributos de calendario: hora del día, día de la semana y jornada laboral.

4.5 Pipeline de Entrenamiento y Serialización

El entrenamiento se estructuró como un pipeline, una secuencia declarada de transformaciones que termina en el clasificador y que se ajusta y aplica como una unidad. Se empleó el ImbPipeline de imbalanced-learn en lugar del estándar de scikit-learn porque aplica SMOTE solo durante el ajuste y lo omite al predecir; un pipeline convencional remuestrearía también los datos de validación y produciría métricas optimistas. Las variables numéricas se imputan con la mediana y se escalan con RobustScaler, que no se deja distorsionar por los valores extremos frecuentes en el tráfico, y el resultado alimenta a XGBClassifier, que emite la probabilidad de tráfico malicioso. Los hiperparámetros —profundidad de los árboles, tasa de aprendizaje y número de árboles— se

eligieron con una búsqueda en grilla y validación cruzada estratificada de cinco particiones, que reparte los datos en cinco partes con la misma proporción de clases y evalúa cada combinación cinco veces.

El pipeline completo, y no solo el clasificador, se serializa con joblib, de modo que la inferencia reproduce exactamente el preprocesamiento del entrenamiento. Se producen dos artefactos: el modelo de veinticinco características, que se conserva como referencia del estudio de validez, y el de veinte, que es el que carga el prototipo.

4.6 Prevención y Verificación de la Fuga de Información

La fuga de información —el acceso del modelo, durante el entrenamiento, a información que no estará disponible o no será informativa en operación— equivale a estudiar con el solucionario a la vista: el examen sale perfecto y nada garantiza que se haya aprendido la materia. Es una de las trampas catalogadas por Arp et al. [34], que la señalan como causa frecuente de los desempeños optimistas que abundan en la literatura de seguridad. Por eso su prevención se trató como requisito de diseño, con cuatro medidas: excluir las direcciones IP del espacio de entrada; conservar de la marca temporal solo la hora y el día, nunca el instante absoluto, que permitiría memorizar cuándo ocurrió cada ataque; aplicar SMOTE dentro del pipeline, y ajustar la imputación y el escalado solo con los datos de entrenamiento de cada partición.

Tras el entrenamiento, y antes de reportar cualquier métrica, una verificación automática comprueba que ninguna característica se llame como una variable de identidad (ip, connection, flow id o timestamp), que el clasificador reciba decenas y no miles de columnas —un recuento de miles delata que una variable de identidad se expandió— y que las características derivadas de la 5-tupla no sean todas constantes, lo que indicaría un fallo silencioso. Estas medidas previenen la fuga, pero no prueban su ausencia; esa prueba es el estudio de exclusión progresiva del Capítulo 6.

5 Validación del Diseño mediante Prototipo

Para demostrar que el diseño puede materializarse se construyó un prototipo funcional en laboratorio, con exportadores físicos de tres fabricantes —Cisco, Fortinet y Huawei— y una mezcla de estaciones de prueba, generadores controlados y el tráfico real que reciben los segmentos del laboratorio expuestos a Internet. Es una maqueta y no la red de producción de una organización, de modo que sus conclusiones son de viabilidad y no de desempeño productivo. El capítulo recorre qué se construyó y qué enseñó la telemetría real, cómo se midió sin etiquetas y qué logra el clasificador sobre ese tráfico, y termina con la anexión de una segunda capa de detección que sus límites hicieron necesaria. El despliegue, la configuración de los equipos y el detalle de las mediciones se recogen en el Anexo E.

5.1 Alcance del Prototipo y Estado de Implementación

La Tabla 4 señala qué partes del diseño del Capítulo 3 están construidas y cuáles permanecen especificadas, distinción que delimita el alcance de las conclusiones.

Tabla 4. Estado de implementación de los componentes del diseño

Componente	Estado	Observación
Stack de microservicios Docker	Implementado	Orquestado con Docker Compose; healthchecks y dependencias declarativas
Captura NetFlow v5/v9/IPFIX (nfcapd)	Implementado	Colector centralizado multi-exportador operativo en puerto 9995/UDP
Procesador de archivos de captura	Implementado (básico)	Interpreta archivos con nfdump e inserta registros; empareja los dos sentidos de cada sesión
Modelo XGBoost + pipeline de transformadores	Implementado	Entrenado en Google Colab; servido vía joblib en la API
API de predicción (FastAPI, /predict, /flows)	Implementado	Validación de entrada con Pydantic; umbral de decisión configurable
Regla de campaña	Implementado	Agrupa por puerto las alertas con cinco o más orígenes; marca sin borrar

Esquema PostgreSQL con índices optimizados	Implementado	Incluye función almacenada de depuración por retención
Dashboards Grafana + monitoreo Prometheus	Implementado	Datasource PostgreSQL directo; métricas de infraestructura
Identificación de exportador (exporter_ip)	Implementado	Cada registro conserva la dirección del exportador que lo envió
Normalización por perfil de fabricante	Parcial	Mapeo de campos y corrección del contador de retorno
Deduplicación y validación cruzada	Parcial	Deduplicación dentro de cada exportador; las copias entre exportadores no se puntúan dos veces; validación cruzada pendiente

El balance es que la cadena de captura, almacenamiento, inferencia y visualización funciona de extremo a extremo con exportadores reales de distintos fabricantes. De los mecanismos multi-dispositivo, en cambio, solo están construidas la identificación del exportador, la deduplicación dentro de cada uno y, en parte, la normalización; la validación cruzada sigue siendo diseño.

5.2 La Telemetría no es el Corpus

El registro que el sistema recibe no es el que el modelo vio al entrenarse, aunque compartan los nombres de campo. CICIDS2017 describe cada conversación como un registro bidireccional, con banderas TCP, resolución de microsegundos y la sesión completa. NetFlow v9 exporta, en cambio, cada sentido por separado, en tramos que corta un temporizador, con resolución de un segundo y, en el caso del cortafuegos, sin banderas. Es la diferencia entre aprender con la transcripción completa de una llamada y tener que juzgar después con dos facturas separadas, una por cada extremo.

De esa diferencia se siguen cuatro lecciones. Primera, el sentido de un flujo —quién inició la conversación— no viene dado y debe definirse una sola vez, en el procesador, para que todos los componentes lo lean igual; mientras cada uno lo resolvía por su cuenta, la respuesta de un servidor propio parecía un sondeo. Segunda, un campo presente puede no ser un dato: según el fabricante, el contador de paquetes de vuelta —un campo que algunos equipos incluyen aunque cada registro NetFlow v9 describe un solo sentido— llegaba copiado del de ida o en cero, y el procesador lo reemplaza por la respuesta observada al emparejar los dos sentidos de cada sesión, cuando el exportador permite observarla. Tercera, lo observable lo decide qué interfaces de los equipos exportan, no el colector. Cuarta, una dirección puede no ser un equipo: detrás de una traducción de direcciones (NAT), un solo origen resume a decenas de personas, y su perfil es atípico sin que haya incidente.

5.3 Conocimiento Declarado del Sitio

La detección se define como la comparación entre el tráfico observado y una expectativa de comportamiento local. Por ejemplo, un sondeo a un puerto de administración constituye un ataque si va dirigido a un host que no publica dicho servicio, pero representa tráfico legítimo si dicho servicio está habilitado. Para ello, el sistema requiere conocer el inventario de la red: qué direcciones son propias, qué servicios están publicados, cuáles sufren traducción de direcciones y qué segmentos carecen de equipos activos. El prototipo estructura este conocimiento mediante declaraciones de rango y rol con vigencia temporal; en lugar de eliminar los registros obsoletos, cada declaración se acota con una fecha de cierre. Esto garantiza que las alertas históricas puedan interpretarse de manera fidedigna según el estado real que tenía la red en el momento de su emisión.

5.4 Medir sin Etiquetas

Sobre tráfico real nadie dice qué flujo es ataque, y la verdad de terreno que reemplace a las etiquetas no debe construirse a partir de la decisión del propio sistema evaluado, porque en ese caso la medición solo confirmaría lo que el sistema ya cree. El prototipo usa dos instrumentos complementarios. El censo responde qué fracción de lo alertado es amenaza, es decir, la precisión: toma todas las alertas de una ventana cerrada y las adjudica una por una con criterios fijados de antemano, sin mostrar la puntuación del modelo. Una alerta es benigna si su origen es propio, si se dirige a un servicio que la organización presta o si responde a una conexión iniciada desde dentro; el resto, sondeos no solicitados a servicios que no se prestan, se declara amenaza.

El telescopio —un instrumento pasivo que observa un bloque de direcciones públicas asignadas, pero sin uso— responde a la pregunta inversa, qué fracción de las amenazas se alerta, es decir, la cobertura o exhaustividad. Aprovecha cinco direcciones públicas propias que no tienen ningún equipo: todo lo que llega a ellas es, por definición, no solicitado, y cada dirección externa que les envía tráfico es un origen no solicitado cuya cobertura puede medirse. Validarlo enseñó que casi todo lo que parecía salir de esas direcciones lo respondía el propio cortafuegos, cuyo módulo de telefonía completa el saludo TCP en nombre de cualquier destino permitido. Haber respuesta no prueba que exista el equipo, y un instrumento de medida debe validarse antes de usarse. Y no solicitado no equivale a malicioso: el origen más persistente eran agentes de un sitio remoto que seguían enviando registros a un colector retirado.

5.5 El Clasificador Entrenado frente al Tráfico Real

El prototipo carga el modelo de veinte características del Capítulo 4, que sobre el corpus, como detalla el Capítulo 6, reconoce más del 99 % de las amenazas de familias conocidas con una precisión de 0,94. Esta sección examina cuánto de ese desempeño sobrevive al tráfico real, medido en una sola instalación y sobre ventanas de horas a pocos días.

Dos resultados se sostienen. El primero es la precisión: el censo de las 562 alertas que el clasificador emitió en veinticuatro horas arrojó un 84,5 % de amenazas, y un 88,7 % si cada par de origen y destino se cuenta una sola vez, aunque se repita. El segundo es la latencia: con los relojes de los exportadores sincronizados, un sondeo TCP queda alertado del orden de 200 s después de ocurrir, una demora que explican la rotación del colector y los temporizadores del exportador y no la inferencia. Son minutos, frente a las semanas de permanencia del atacante que reporta la literatura.

El primer límite es que el modelo depende del puerto. En un día de tráfico entrante no solicitado, todas sus alertas recayeron sobre los siete puertos que una de sus características codifica como sensibles —los mismos de la regla R1— y ninguna sobre los miles de puertos restantes, pese a que el tráfico contenía reconocimiento equivalente sobre otros servicios. Dos sondeos de la misma forma, uno alertado y otro no, solo se distinguían en esa componente, de modo que ningún umbral lo remedia: desplegado, el clasificador se comporta como un refinamiento aprendido de una regla estática.

El segundo límite es la brecha de dominio. En CICIDS2017 el objetivo de un sondeo contestaba con un reinicio de la conexión; en la red real el cortafuegos deniega y nadie responde. Cuando el procesador reconstruyó la respuesta verdadera, el sondeo denegado apareció como lo que es, un flujo sin respuesta, un caso que el modelo nunca vio, y dejó de reconocerlo: la puntuación mediana de doscientos sondeos reales cayó de 0,895 a 0,336. El telescopio lo confirma con una verdad independiente del puerto, pues de 20.140 orígenes externos que contactaron direcciones sin equipos en setenta y dos horas, el clasificador alertó al 3,1 %. El modelo es preciso porque lo que aprendió sigue siendo cierto para los casos que reconoce, y poco exhaustivo porque los que no reconoce no se parecen a nada que haya visto; ninguno de esos límites se corrige ajustando el umbral.

5.6 La Carga de Triaje en Operación

En el prototipo, la carga de triaje se calcula con la misma fórmula que en el corpus —alertas presentadas divididas por las que resultan ser amenazas reales—, pero con otra fuente de verdad: las amenazas reales no las indican etiquetas, sino la adjudicación del censo. Con ella, la carga es de 1,18 alertas por flujo revisadas por cada amenaza hallada, o de 1,13 si cada par de origen y destino se cuenta una vez, valores cercanos a los del corpus con el mismo umbral de 0,7 (1,046 y 1,107), aunque no directamente comparables: en el prototipo el clasificador alerta casi solo sobre los puertos sensibles de segmentos expuestos a Internet, donde el sondeo es frecuente, y su cobertura es baja. La cifra resiste el criterio del adjudicador: según cuánta evidencia de sondeo se exija para declarar amenaza una alerta y cómo se resuelvan los casos dudosos, oscila entre 1,15 y 1,28.

Lo que no pudo medirse en el prototipo es la comparación con las reglas estáticas. Las reglas R3 y R4 suponen registros bidireccionales y pierden su sentido sobre los del cortafuegos: con una resolución de un segundo, el percentil 1 de la duración es cero y R3 nunca se cumple, y una vez emparejados los sentidos, la ausencia de respuesta de R4 se cumple en la mayor parte del tráfico. Además, la línea base resultante emite casi

todas las alertas del clasificador, y el criterio del censo depende del puerto igual que R1, de modo que la verdad de terreno no es independiente de la línea base. Queda, por último, la segunda mitad del criterio: una carga baja con una cobertura casi nula no es el resultado que la hipótesis describe. La hipótesis queda verificada sobre el corpus y pendiente sobre tráfico real, porque la comparación con las reglas no pudo medirse y la cobertura del clasificador resultó insuficiente.

5.7 La Segunda Capa: Detección por Agregados

Los límites del clasificador tienen una parte que ningún reentrenamiento por flujo resolvería. El sondeo no se reconoce en un flujo aislado, sino en su acumulación: un intento de conexión no dice nada, pero cientos de intentos del mismo origen en pocos minutos sí. Un clasificador que juzga flujo a flujo es, por construcción, ciego a esa señal. Por eso el prototipo incorporó un detector complementario que cambia la unidad de observación y, en lugar de juzgar flujos, juzga orígenes. No forma parte del diseño del Capítulo 3 ni es la línea base de reglas, que solo sirve de comparación: es un componente que la operación hizo necesario.

El detector reúne por origen los flujos que este inició en las últimas veinticuatro horas, en todos los exportadores, y construye un perfil con los destinos y puertos contactados, el volumen, la diversidad de protocolos y la dispersión horaria. Sobre esos perfiles aplica un bosque de aislamiento (Isolation Forest) [21], un algoritmo no supervisado que señala los orígenes más fáciles de separar del resto, es decir, los más atípicos, y dos reglas explícitas, una de sondeo de puertos y otra de denegación de servicio. Su volumen lo fija un presupuesto de cien alertas por ejecución, de modo que su precisión es la de las primeras alertas del ordenamiento y no la de un umbral.

Sobre los mismos 20.140 orígenes del telescopio, el sistema completo alertó al 17,8 %, frente al 3,1 % del clasificador, y la cobertura crece con la intensidad del origen: el 8,8 % de los que enviaron un solo flujo, el 21,5 % de los que enviaron entre seis y veinte y la totalidad de los que superaron los cien; sobre los de más de veinte flujos alcanza el 79,5 %. Lo no cubierto es, en su mayor parte, radiación de fondo de Internet de baja intensidad, pero también marca el punto ciego del sistema: un atacante lento que envía pocos flujos espaciados. Las dos capas se complementan, pues el clasificador aporta precisión sobre cada flujo y el detector, cobertura sobre cada origen.

6 Evaluación y Resultados

6.1 Metodología de Evaluación

Este capítulo evalúa el modelo tal cual fue entrenado, sobre el corpus etiquetado, donde cada flujo trae su respuesta correcta y las métricas pueden calcularse con exactitud. La evaluación se realiza en la misma plataforma en que se entrenó el modelo (Google Colab): una vez entrenado, se evalúa de inmediato bajo dos regímenes de partición, aleatoria y temporal. Se emplean cuatro métricas. La precisión es la fracción de las alertas que son amenazas reales; la cobertura, o exhaustividad, la fracción del conjunto de amenazas reales que el modelo alerta; el F1-Score combina ambas en un solo valor, que solo es alto si las dos lo son, y el AUC-ROC mide, entre 0,5 y 1, la capacidad de ordenar los flujos maliciosos por sobre los benignos con independencia del umbral. Las métricas del clasificador se reportan con el umbral neutro de 0,5, y la carga de triaje, además, con el umbral operativo de 0,7. El Anexo C detalla el procedimiento reproducible y los resultados complementarios.

Los dos regímenes son deliberadamente contrapuestos: su diferencia es la que separa examinar a un estudiante con los ejercicios que practicó de examinarlo con ejercicios nuevos del mismo temario. El primero es la partición aleatoria estratificada, convención dominante en la literatura: los 2.660.377 flujos se reparten al azar en un 70 % de entrenamiento y un 30 % de prueba —1.862.263 y 798.114 registros—, preservando la proporción entre clases. Mide qué desempeño alcanza el modelo sobre familias que ya conoce, y describe la operación prevista de un detector que se reentrena con regularidad.

El segundo es la partición temporal: el modelo se entrena con los tres primeros días —1.668.530 flujos— y se evalúa con los dos últimos —991.847—. Como CICIDS2017 concentró cada familia en una sola jornada,

equivale a evaluar con familias nunca vistas y responde a la pregunta operacionalmente relevante de qué ocurre cuando el sistema enfrenta una amenaza nueva. Pendlebury et al. [40] demostraron que la partición aleatoria produce estimaciones optimistas que no se sostienen al desplegar, y por eso este segundo régimen es indispensable.

6.2 Desempeño bajo Partición Aleatoria

Con sus veinticinco características y bajo partición aleatoria, el modelo alcanza una precisión de 0,9997, una cobertura de 0,9999, un F1-Score de 0,9998 y un AUC-ROC de 1,0000: sobre 798.114 flujos de prueba comete cuarenta y nueve falsos positivos y dieciséis falsos negativos. Un resultado así exige auditarlo. La primera sospecha, el sobreajuste, queda descartada, porque el F1-Score es de 0,9999 sobre el entrenamiento y de 0,9998 sobre la prueba. Ese descarte agudiza la cuestión: si el modelo generaliza sin pérdida y aun así acierta casi siempre, la explicación no está en el algoritmo, sino en que el problema que resuelve es más simple que el planteado.

6.3 Diagnóstico de Validez y Selección del Modelo Desplegable

La primera pista la da la importancia por ganancia. Cada vez que un árbol de XGBoost divide los datos según una característica, el ajuste del modelo mejora en alguna medida; la ganancia de una característica es la suma de esas mejoras en todos los árboles, expresada como fracción del total, de modo que las veinticinco suman 100 %. Tres características concentran el 73,3 % y ninguna describe el comportamiento del tráfico: el protocolo de transporte (44,3 %) y los indicadores de dirección privada de origen y de destino (15,1 % y 13,9 %), que describen la topología del banco de pruebas. La primera característica de comportamiento aparece recién en cuarto lugar, con un 4,8 %, y los tres atributos de calendario —hora del día, día de la semana e indicador de jornada laboral— suman entre los tres un 3,4 %.

Cuatro comprobaciones confirmaron la sospecha. La combinación de direcciones privadas concentra el 99,62 % de los ataques del conjunto de prueba, y una única dirección, la máquina atacante del banco de pruebas, origina el 99,6 % de los flujos maliciosos. La proporción de tráfico malicioso va del 0,01 % del jueves al 41,02 % del viernes, de modo que el día de la semana transporta información de la etiqueta. Y el protocolo aparecía duplicado, porque un transformador lo reconstruía desde la 5-tupla, defecto que se corrigió. Para cuantificar el efecto, se reentrenó el modelo retirando bloques de características de forma acumulativa y se evaluaron los modelos de veinticinco y de veinte características bajo ambos regímenes. La Tabla 5 reúne los resultados.

Tabla 5. Exclusión progresiva de características y desempeño del modelo desplegable bajo ambos regímenes de partición

Escenario	Régimen	F1-Score	Precisión	Cobertura	AUC-ROC
A. Todas las características (25)	Aleatorio	0,9998	0,9997	0,9999	1,0000
B. Sin indicadores de topología (23)	Aleatorio	0,9979	—	—	—
C. Sin topología ni calendario (20, desplegable)	Aleatorio	0,9681	0,9419	0,9957	0,9994
D. Solo comportamiento del flujo (16)	Aleatorio	0,9590	—	—	—
A. Todas las características (25)	Temporal	0,4394	1,0000	0,2815	0,9474
C. Sin topología ni calendario (20, desplegable)	Temporal	0,4285	0,8873	0,2825	0,7535

La tabla admite tres lecturas. En B y D solo se registró el F1-Score; A y C son los modelos evaluados en detalle, que en el estudio de exclusión, entrenado por separado, obtuvieron 0,9997 y 0,9674, diferencias inferiores a una milésima que no alteran las conclusiones. La primera es que la fuga existe y es medible: retirar la topología resta 0,0019 al F1-Score, y retirar además el calendario resta 0,0298. Ahí se aclara una aparente paradoja, pues el calendario reunía solo el 3,4 % de la ganancia y es, sin embargo, el bloque cuya retirada más daña. La ganancia mide cuánto aprovecha el modelo entrenado cada característica, no cuánto se perdería sin ella: una característica que transporta la etiqueta puede intervenir en pocas divisiones y aun así resolver casos que ninguna otra resuelve.

La segunda lectura es que la fuga no explica el resultado: con solo las dieciséis características de comportamiento, el modelo conserva un F1-Score de 0,9590. Como prueba inversa, un modelo alimentado únicamente con el protocolo y la marca de tiempo alcanzó 0,915 sin observar comportamiento alguno.

La tercera lectura fija qué se despliega. Retirar las cinco características contaminadas cuesta 0,0317 bajo partición aleatoria, de 0,9998 a 0,9681, y esa diferencia es la magnitud de la fuga, no capacidad de detección trasladable a otra red. Bajo partición temporal, la ventaja del modelo de veinticinco características se reduce a 0,0109 y es aparente: un defecto en la conversión de la marca temporal lo llevó a emitir apenas dos falsos positivos, con una precisión perfecta, la misma cobertura y un AUC-ROC más alto que obedece a la misma causa, no a una mejor discriminación. Lo que enseñan los indicadores de topología —que el tráfico interno es sospechoso—, por su parte, se invierte en una red empresarial. De ahí la decisión: el prototipo emplea el modelo de veinte características, cuyo F1-Score de 0,968 sobre familias conocidas es la cota superior que este trabajo reporta.

6.3.1 Robustez ante Familias No Observadas

La partición temporal examina qué ocurre ante familias enteramente nuevas: el entrenamiento contiene fuerza bruta, cuatro variantes de denegación de servicio y Heartbleed, y la prueba contiene infiltración, DDoS, sondeo de puertos y botnet, sin una sola familia compartida. Como el corpus no admite una partición cronológica menos severa, el resultado no estima una degradación gradual, sino una cota inferior en el peor caso. El F1-Score cae a 0,4285, y la forma de la caída importa más que su magnitud: la precisión se mantiene en el 88,7 % y lo que se derrumba es la cobertura, hasta el 28,3 %.

El desglose por familia separa dos comportamientos. El modelo generaliza cuando dispone de un análogo conductual: nunca vio DDoS, pero sí denegación de servicio, y como ambas producen flujos cortos, de alta tasa y con poca respuesta, detecta el 63,57 % de la DDoS, lo que acredita que aprendió comportamiento y no ejemplos. No detecta, en cambio, el sondeo de puertos (0,14 %) ni la botnet, cuya señal reside en el agregado de muchos flujos de un mismo origen y no en un flujo aislado. Ajustar el umbral no lo remedia: bajarlo de 0,50 a 0,02 eleva la cobertura apenas del 28,25 % al 29,01 % mientras la precisión cae al 66,9 %, porque las amenazas no detectadas reciben una probabilidad cercana a cero. El techo es estructural, y la respuesta está en cambiar la unidad de observación, como hace el detector por agregados descrito en el Capítulo 5.

6.4 Cuantificación de la Hipótesis: la Carga de Triaje sobre el Corpus

Esta sección contrasta la hipótesis sobre el corpus, donde las etiquetas indican con exactitud cuáles de las alertas presentadas son amenazas reales. La carga de triaje se lee cuanto menor mejor, con 1 como valor ideal, y siempre junto a la cobertura, porque un detector que presentara un único evento y acertara tendría una carga de 1 sin haber detectado casi nada. La Tabla 6 compara el modelo desplegable con dos líneas base, las reglas R1–R4 y la revisión exhaustiva de todos los flujos, en ambos regímenes.

Tabla 6. Carga de triaje sobre el corpus: eventos presentados por cada amenaza real hallada, frente a las líneas base, en ambos regímenes

Escenario	Eventos presentados	% de los flujos	Amenazas halladas	Cobertura	Carga de triaje
Partición aleatoria (familias conocidas): 798.114 flujos de prueba, 166.640 amenazas reales					
Revisión exhaustiva (sin filtro)	798.114	100,00 %	166.640	100,00 %	4,789
Reglas estáticas R1–R4	143.660	18,00 %	39.479	23,69 %	3,639
Modelo XGBoost (umbral 0,5)	176.151	22,07 %	165.922	99,57 %	1,062
Modelo XGBoost (umbral 0,7)	172.266	21,58 %	164.640	98,80 %	1,046
Partición temporal (familias nunca vistas): 991.847 flujos de prueba, 288.959 amenazas reales					
Revisión exhaustiva (sin filtro)	991.847	100,00 %	288.959	100,00 %	3,432

Reglas estáticas R1–R4	178.904	18,04 %	49.682	17,19 %	3,601
Modelo XGBoost (umbral 0,5)	91.994	9,28 %	81.622	28,25 %	1,127
Modelo XGBoost (umbral 0,7)	90.209	9,10 %	81.506	28,21 %	1,107

Ante familias conocidas, la carga desciende a 1,046 alertas por amenaza hallada —prácticamente una alerta por amenaza— con una cobertura del 98,8 %, una reducción del 71,3 % respecto de las reglas estáticas. Un matiz contradice la intuición: el modelo presenta más eventos que las reglas, el 21,58 % de los flujos frente al 18,00 %, y aun así su carga es menos de un tercio; lo que agota al analista no es el número de alertas, sino el de alertas inútiles por hallazgo. Ante familias nunca vistas, la reducción se mantiene en el 69,3 %, de 3,601 a 1,107, y la cobertura sube del 17,19 % al 28,21 %.

Se cumple, por tanto, el criterio fijado en el Capítulo 1: sobre un mismo conjunto de flujos, el clasificador exhibe una carga de triaje estrictamente menor que la de la línea base y una cobertura mayor, en ambos regímenes.

Queda por saber qué anticipa esta cifra para una red en operación. Por ser un cociente, la carga es insensible al volumen de tráfico, pero no a la proporción de tráfico malicioso, que en la prueba temporal es del 29,1 % y en producción rara vez supera el 1 %. Es el problema de la tasa base que Axelsson identificó como dificultad central de la detección de intrusiones [41]: con el mismo clasificador, la carga crece a medida que los ataques se vuelven más raros, como muestra la Tabla 7, calculada con las tasas de detección y de falsas alarmas de la prueba temporal y el umbral de 0,5.

Tabla 7. Efecto de la proporción de tráfico malicioso sobre la carga de triaje, a igualdad de desempeño del clasificador

Tráfico malicioso en el conjunto evaluado	**Precisión del detector**	**Carga de triaje**
29,1 % (prueba temporal del corpus, umbral 0,5)	88,7 %	1,13
10 %	68,0 %	1,47
5 %	50,2 %	1,99
1 % (orden de magnitud habitual en producción)	16,2 %	6,17
0,1 %	1,9 %	53,19

Los valores absolutos de la carga no se trasladan, por tanto, a otra red. La comparación, en cambio, conserva su sentido e incluso se acentúa: con las tasas medidas en la prueba temporal, a una proporción del 1 % la carga de las reglas sería de unas 107 alertas por amenaza y la del clasificador de 6,17, porque las reglas alertan sobre una fracción mucho mayor del tráfico benigno. Antes de comprometer una cifra en otro entorno basta con volver a medirla allí.

6.5 Del Corpus a la Operación: Lectura Conjunta

Los resultados sobre el corpus y los del prototipo describen el mismo modelo desde dos lados. Lo que el corpus permitía anticipar se cumplió: en el modelo desplegable, el protocolo y los atributos de puerto concentran el 53,61 % de la ganancia, y en operación el modelo dependió del puerto; la partición temporal mostraba ceguera ante el sondeo, y el telescopio la confirmó; la precisión resistió ante familias nuevas y se sostuvo en tráfico real. Lo que el corpus no podía anticipar provino de la telemetría, pues el sondeo sin respuesta no existe en CICIDS2017 y los registros unidireccionales del cortafuegos no se parecen a sus conversaciones completas. La cota de 0,968 describe lo que el modelo aprendió; las mediciones en operación, cuánto de ese aprendizaje corresponde a lo que el sistema observa. Esa distancia es la brecha de dominio, y reducirla es el primer objetivo de la hoja de ruta del Capítulo 7.

7 Conclusiones

7.1 Validación de la Hipótesis

La hipótesis se operacionalizó mediante la carga de triaje, y las tres afirmaciones que reúne tienen un estado distinto. La primera, que el sistema reduce la carga respecto de las reglas estáticas sobre datos etiquetados, queda verificada: con el modelo desplegable, la carga desciende a 1,046 alertas por amenaza hallada ante familias conocidas, una reducción del 71,3 %, y ante familias nunca vistas la reducción sigue siendo del 69,3 %, con mayor cobertura que las reglas. La segunda, que esa mejora se sostiene sobre tráfico real, queda pendiente. En otras palabras, no verificada: en el prototipo el 84,5 % de las alertas resultaron amenazas, con una carga de 1,18, pero la comparación con las reglas no pudo medirse (sección 5.6) y el clasificador alertó solo al 3,1 % de los orígenes no solicitados; esas cifras acreditan precisión y viabilidad, no eficiencia operacional. La tercera, que el resultado se mantiene con exportadores de distintos fabricantes, no fue contrastada: la cadena integra los equipos de los tres fabricantes de extremo a extremo, pero hoy el clasificador puntúa solo los registros del cortafuegos, porque en los de los demás exportadores el emparejamiento aún no puede establecer si hubo respuesta, y no hay aún base para comparar la detección entre ellos. Contrastar ambas exige extender la inferencia a todos los exportadores y aplicar en el laboratorio el protocolo común de evaluación (M6) y, después, un despliegue piloto en una red en producción (sección 7.5).

En calidad de clasificación, el trabajo reporta dos cotas y no una cifra: un F1-Score de 0,9681 sobre familias conocidas y de 0,4285 sobre familias no observadas, evaluadas en el corpus. En el prototipo, el sistema alerta un sondeo TCP del orden de 200 s después de ocurrido. En la captura multi-dispositivo, la contribución es de diseño: están construidas la identificación del exportador y la deduplicación dentro de cada uno, pero no la validación cruzada, de la que depende el experimento que compara un exportador con todos.

7.2 Utilidad del Aprendizaje Automático para la Detección de Amenazas

La pregunta que dejan los resultados es directa: si el clasificador alcanza un F1-Score de 0,968 sobre el corpus pero alerta solo al 3,1 % de los orígenes no solicitados en el prototipo, ¿es útil el aprendizaje automático para detectar amenazas? La respuesta de este trabajo es afirmativa, como mecanismo de priorización dentro de un sistema más amplio. Priorizar significa aquí ordenar: el modelo asigna a cada flujo una probabilidad de amenaza, y ordenar las alertas por ella hace que el tiempo escaso del analista vaya primero a lo más probable; la carga de triaje mide cuánto esfuerzo exige revisar lo que el modelo presenta. El sistema más amplio lo completan la capa que correlaciona los flujos de un mismo origen y el conocimiento declarado del sitio.

Los resultados débiles del prototipo no provienen del algoritmo, sino de la distancia entre lo que el modelo aprendió y lo que observa, y cada falla tiene una causa identificada: depende de una lista de puertos porque esa lista era un atajo eficaz en el corpus; no reconoce el sondeo sin respuesta porque ese caso no existe en el corpus; aprendió de conversaciones completas y recibe registros unidireccionales, y juzga flujos aislados cuando la señal del sondeo está en el agregado. Otro algoritmo entrenado con los mismos datos heredaría las cuatro causas, y todas admiten corrección con los mismos métodos: entrenar con los campos que el sistema observará, retirar los atajos, incluir el caso ausente y cambiar la unidad de observación.

Hay, además, evidencia a favor. El modelo reconoció la DDoS sin haberla visto, lo que indica que aprendió comportamiento; su precisión se sostuvo en las tres mediciones —0,94 y 0,89 sobre el corpus, ante familias conocidas y nuevas, y 0,85 en el prototipo—; la cobertura que el sistema alcanzó provino de un bosque de aislamiento (Isolation Forest), también aprendizaje automático, y las reglas estáticas no salieron mejor paradas del cambio de telemetría. Lo que no se sostiene es usar un clasificador entrenado sobre un corpus público, como es el CICIDS2017, como detector listo para instalar.

7.3 Cumplimiento de los Objetivos Específicos

El Capítulo 1 asoció cada objetivo específico a un resultado verificable. La Tabla 8 recoge, para cada uno, dónde se aborda y qué resultado lo acredita.

Tabla 8. Cumplimiento de los objetivos específicos

Objetivo	Dónde se aborda	Resultado que lo acredita	Estado
1.4.1 Caracterizar NetFlow/IPFIX y sus alternativas	Capítulo 2; Tabla A.8	NetFlow/IPFIX seleccionado por satisfacer los cinco criterios de evaluación sin renovar el equipamiento instalado	*Cumplido*
1.4.2 Diseñar la arquitectura de microservicios	Capítulo 3; Figuras 1 y 2; Anexo D	Cadena completa de captura, normalización, inferencia, almacenamiento y visualización operativa; identificación del exportador y deduplicación por exportador construidas (Tabla 4)	*Cumplido en diseño; parcial en construcción*
1.4.3 Construir un flujo de entrenamiento reproducible	Capítulo 4; Anexo C	Pipeline con transformadores de dominio y SMOTE, semilla fija y cuaderno ejecutable que reproduce todas las cifras reportadas	*Cumplido*
1.4.4 Verificar que el desempeño no provenga de fugas de información	Capítulos 4 y 6; Tabla 5	Controles automáticos de entrada y exclusión progresiva de características; cota superior defendible F1 = 0,968 del modelo desplegable, tras descartar cinco características contaminadas	*Cumplido*
1.4.5 Cuantificar la hipótesis	Capítulo 6; Tabla 6	Carga de triaje sobre el corpus reducida de 3,601 a 1,107 (−69,3 %) con cobertura ×1,6, bajo familias de ataque no observadas	*Cumplido*
1.4.6 Materializar el prototipo de laboratorio	Capítulo 5; Anexo E	Operación de extremo a extremo verificada con exportadores de tres fabricantes; MTTD, precisión y exhaustividad medidos sobre tráfico real	*Cumplido; medición operacional parcial*

7.4 Contribuciones Principales

Este trabajo realizó cinco contribuciones. Las tres primeras son técnicas: una arquitectura de referencia reproducible, en contenedores, para detectar amenazas sobre NetFlow con aprendizaje automático y un clasificador intercambiable, que operó de forma continua sobre tráfico real de Internet en un laboratorio; un procedimiento para verificar la validez de los resultados de un modelo entrenado sobre un corpus público —controles de entrada, exclusión progresiva y dos regímenes de partición—, que descartó cinco de veinticinco características y es reutilizable con cualquier corpus, y el diseño de un mecanismo de captura y correlación multi-dispositivo independiente del fabricante.

La cuarta es conceptual: detectar es comparar lo observado con una expectativa local, de modo que la personalización es una condición de la tarea y no un ajuste opcional. Las plataformas de seguridad comerciales, como los SIEM, lo resuelven pidiendo esa información al implantarse [42], y herramientas de código abierto como SiLK exigen declararla [31]; y este trabajo lo formula como conocimiento declarado del sitio, por rango y rol y con vigencia. La quinta es metodológica: un censo de alertas y un telescopio de direcciones sin equipos permiten medir precisión y cobertura sin etiquetas, con una verdad de terreno construida sin mirar la puntuación del detector y que, a su vez, debe validarse. Lo que se transfiere a otra red no es el detector entrenado, sino la arquitectura, el procedimiento de verificación y la forma de medir.

7.5 Alcances y Limitaciones

El modelo se entrenó exclusivamente con CICIDS2017, generado en 2017 y con defectos de construcción documentados que el trabajo acota pero no corrige, lo que limita su capacidad ante familias posteriores; como cada familia se concentra en una sola jornada, la cota inferior reportada corresponde a familias excluidas y no a una deriva gradual. La carga de triaje depende además de la proporción de tráfico malicioso [41], lo que no invalida la comparación, pero impide trasladar su valor absoluto.

El prototipo opera en un laboratorio, en una sola instalación y sobre ventanas de horas a días, de modo que generalizar exige un despliegue piloto en producción. El muestreo de algunos equipos introduce un error irreducible en los contadores, los que no exportan estadísticas por sentido (dirección) obligan a imputar características, el cortafuegos que concentra el tráfico no exporta banderas TCP y es hoy el único exportador cuyos registros puntúa el clasificador, y la configuración del sitio —puertos sensibles y umbrales del detector— aún reside en el código.

Subsiste una limitación estructural: la señal del sondeo y del tráfico de baliza (beaconing traffic) reside en el agregado de muchos flujos de un mismo origen, y ningún reentrenamiento con el mismo espacio de características lo corrige. De ahí la capa agregada y la línea de trabajo que propone clasificar también por origen y ventana.

7.6 Trabajos Futuros

La línea principal es mejorar el modelo, que observa de manera distinta a como fue entrenado, conserva atajos del corpus y mira una unidad que no corresponde a ciertos ataques. Las líneas de mejora se identifican como M1 a M6: paridad de características entre entrenamiento y operación (M1), retiro de atajos (M2), sondeo sin respuesta (M3), clasificación por origen y ventana (M4), calibración local (M5) y protocolo común de evaluación (M6). Se organizan en una hoja de ruta en fases, porque cada fase se apoya en la anterior y se cierra con un criterio medible. Los marcos de referencia de la industria —el NIST Cybersecurity Framework [36] y MITRE ATT&CK [43]— se incorporan a medida que el sistema adquiere las capacidades que describen, como resume la Tabla 9.

Tabla 9. Hoja de ruta de mejora del modelo y del sistema

Horizonte y fase	Objetivo	Líneas de trabajo y marcos	Criterio de salida
Corto plazo · Fase 1	Un modelo que generalice a la telemetría real	M1 a M3 y M6; corpus NetFlow NF-UQ-NIDS-v2; ATT&CK como vocabulario de las alertas	Supera al modelo actual en precisión del censo y en cobertura del telescopio, con el mismo protocolo
Mediano plazo · Fase 2	Contexto y confianza	M4 y M5; ATT&CK como métrica de cobertura; ubicación en las funciones Identificar y Detectar de NIST CSF	Puntajes calibrados y un mapa de cobertura por técnica
Mediano plazo · Fase 3	Verdad de terreno local	Emulación de técnicas ATT&CK en un segmento de laboratorio autorizado	Tasa de detección por técnica medida, no inferida
Largo plazo · Fase 4	Operación sostenible	Retroalimentación del analista, detección de deriva y reentrenamiento gobernado; funciones Gobernar y Responder de NIST CSF; NIST AI RMF	Un modelo que se mantiene vigente sin intervención artesanal
Largo plazo · Fase 5	Más allá del flujo	Correlación con fuentes de host y registros de eventos	Cobertura de técnicas que NetFlow no observa

En el corto plazo, la fase 1 ataca las causas de la brecha medida: entrenar con lo que se observará (M1), reprocesando las capturas de CICIDS2017 por la misma cadena de exportación del prototipo o adoptando un corpus nacido en NetFlow, como NF-UQ-NIDS-v2 [32], verificado campo por campo frente a lo que exporta cada fabricante; reemplazar la lista fija de puertos por la condición de servicio declarado y tratar el protocolo como categoría (M2); incluir el sondeo sin respuesta en el entrenamiento (M3), y evaluar cada versión con un protocolo común —ambos regímenes de partición, censo y telescopio— sobre un corpus ampliado con versiones reprocesadas o corregidas de CICIDS2017 [37], [44], [45] o capturadas en otra infraestructura [46], [47] (M6). ATT&CK entra aquí como vocabulario: una tabla declarada asocia cada tipo de alerta a una táctica y una técnica, separando lo que un registro de flujo puede observar de lo que no.

En el mediano plazo, la fase 2 clasifica también por origen y ventana de tiempo (M4) y calibra el puntaje con la muestra adjudicada (M5), y ATT&CK pasa de vocabulario a métrica de cobertura por técnica; la fase 3 emula técnicas ATT&CK en un segmento de laboratorio autorizado para obtener verdad de terreno local. En el largo plazo, la fase 4 hace sostenible la operación —corpus propio a partir de los veredictos del analista, detección de deriva y reentrenamiento gobernado, en línea con el NIST CSF y con el marco de gestión de riesgos de la inteligencia artificial [48]—, y la fase 5 correlaciona las alertas con fuentes de host para cubrir técnicas que NetFlow no observa. El orden no es arbitrario: sin paridad de características, la calibración ajustaría un modelo equivocado.

Referencias

[1] Unión Internacional de Telecomunicaciones (UIT), "Internet traffic," en Measuring Digital Development: Facts and Figures 2025, Ginebra, Suiza: ITU Publications, nov. 2025. Disponible: https://www.itu.int/itu-d/reports/statistics/2025/10/15/ff25-internet-traffic/

[2] R. Hofstede, P. Čeleda, B. Trammell, I. Drago, R. Sadre, A. Sperotto y A. Pras, "Flow Monitoring Explained: From Packet Capture to Data Analysis With NetFlow and IPFIX," IEEE Communications Surveys & Tutorials, vol. 16, no. 4, pp. 2037-2064, 2014.

[3] T. Chen y C. Guestrin, "XGBoost: A Scalable Tree Boosting System," en Proc. 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, San Francisco, CA, EE. UU., pp. 785-794, 2016.

[4] L. Grinsztajn, E. Oyallon y G. Varoquaux, "Why do tree-based models still outperform deep learning on typical tabular data?," en Advances in Neural Information Processing Systems 35 (NeurIPS 2022), Datasets and Benchmarks Track, pp. 507-520, 2022.

[5] M. Sarhan, S. Layeghy, N. Moustafa y M. Portmann, "NetFlow Datasets for Machine Learning-Based Network Intrusion Detection Systems," en Big Data Technologies and Applications (BDTA/WiCON 2020), Lecture Notes of the Institute for Computer Sciences, Social Informatics and Telecommunications Engineering (LNICST), vol. 371, Springer, pp. 117-135, 2021.

[6] IDC, "The Voice of the Analysts: Improving Security Operations Center Processes Through Adapted Technologies," informe patrocinado por FireEye, Inc., ene. 2021.

[7] L. Yang, Z. Chen, C. Wang, Z. Zhang, S. Booma, P. Cao, C. Adam, A. Withers, Z. Kalbarczyk, R. K. Iyer y G. Wang, "True Attacks, Attack Attempts, or Benign Triggers? An Empirical Measurement of Network Alerts in a Security Operations Center," en Proc. 33rd USENIX Security Symposium, Philadelphia, PA, EE. UU., pp. 1525-1542, 2024.

[8] B. A. Alahmadi, L. Axon e I. Martinovic, "99% False Positives: A Qualitative Study of SOC Analysts' Perspectives on Security Alarms," en Proc. 31st USENIX Security Symposium, Boston, MA, EE. UU., pp. 2783-2800, 2022.

[9] Mandiant, "M-Trends 2026: Data, Insights, and Strategies From the Frontlines," Google Cloud, mar. 2026. Disponible: https://cloud.google.com/security/resources/m-trends

[10] IBM Security, "Cost of a Data Breach Report 2026," IBM Corporation, Armonk, NY, EE. UU., jul. 2026. Disponible: https://www.ibm.com/reports/data-breach

[11] FastAPI, "FastAPI Documentation," documentación oficial del proyecto, 2024. Disponible: https://fastapi.tiangolo.com/

[12] PostgreSQL Global Development Group, "PostgreSQL 15 Documentation," documentación oficial del proyecto, 2022. Disponible: https://www.postgresql.org/docs/15/

[13] Grafana Labs, "Grafana Documentation," versión 10.4, documentación oficial del proyecto, 2024. Disponible: https://grafana.com/docs/grafana/v10.4/

[14] S. Newman, Building Microservices: Designing Fine-Grained Systems, 2.ª ed. Sebastopol, CA, EE. UU.: O'Reilly Media, 2021.

[15] I. Sharafaldin, A. Habibi Lashkari y A. A. Ghorbani, "Toward Generating a New Intrusion Detection Dataset and Intrusion Traffic Characterization," en Proc. 4th International Conference on Information Systems Security and Privacy (ICISSP), Funchal, Portugal, pp. 108-116, 2018.

[16] N. V. Chawla, K. W. Bowyer, L. O. Hall y W. P. Kegelmeyer, "SMOTE: Synthetic Minority Over-sampling Technique," Journal of Artificial Intelligence Research, vol. 16, pp. 321-357, 2002.

[17] A. Sperotto, G. Schaffrath, R. Sadre, C. Morariu, A. Pras y B. Stiller, "An Overview of IP Flow-Based Intrusion Detection," IEEE Communications Surveys & Tutorials, vol. 12, no. 3, pp. 343-356, 2010.

[18] J. Jung, V. Paxson, A. W. Berger y H. Balakrishnan, "Fast Portscan Detection Using Sequential Hypothesis Testing," en Proc. IEEE Symposium on Security and Privacy, Oakland, CA, EE. UU., pp. 211-225, 2004.

[19] B. Claise, Ed., G. Sadasivan, V. Valluri y M. Djernaes, "Cisco Systems NetFlow Services Export Version 9," RFC 3954, IETF, oct. 2004.

[20] B. Claise, Ed., B. Trammell, Ed., y P. Aitken, "Specification of the IP Flow Information Export (IPFIX) Protocol for the Exchange of Flow Information," RFC 7011 (STD 77), IETF, sep. 2013.

[21] F. T. Liu, K. M. Ting y Z.-H. Zhou, "Isolation Forest," en Proc. 8th IEEE International Conference on Data Mining (ICDM), Pisa, Italia, pp. 413-422, 2008.

[22] R. Panigrahi y S. Borah, "A detailed analysis of CICIDS2017 dataset for designing Intrusion Detection Systems," International Journal of Engineering & Technology, vol. 7, no. 3.24, pp. 479-482, 2018.

[23] M. Ring, S. Wunderlich, D. Scheuring, D. Landes y A. Hotho, "A survey of network-based intrusion detection data sets," Computers & Security, vol. 86, pp. 147-167, 2019.

[24] J. H. Friedman, "Greedy Function Approximation: A Gradient Boosting Machine," The Annals of Statistics, vol. 29, no. 5, pp. 1189-1232, 2001.

[25] N. Capuano, G. Fenza, V. Loia y C. Stanzione, "Explainable Artificial Intelligence in CyberSecurity: A Survey," IEEE Access, vol. 10, pp. 93575-93600, 2022.
[26] NETSCOUT Systems, "Arbor Sightline: DDoS Attack Detection Solution" y "Why NetFlow Is Still the Best Mechanism for Detecting DDoS Attacks," documentación de producto, consultada en 2026. Disponible: https://www.netscout.com/product/arbor-sightline
[27] Nokia, "Deepfield Defender: Advanced DDoS Security," documentación de producto, consultada en 2026. Disponible: https://www.nokia.com/ip-networks/deepfield/defender/
[28] Genie Networks, "DDoS Defense Solution: GenieATM," documentación de producto, consultada en 2026. Disponible: https://www.genie-networks.com/ddos-protection/
[29] Radware, "DefenseFlow: SDN Based Adaptive Attack Mitigation Solution," hoja de datos del producto, consultada en 2026. Disponible: https://www.radappliances.com/defenseflow.asp
[30] P. Odintsov, "FastNetMon: Very Fast DDoS Sensor with sFlow/NetFlow/IPFIX/SPAN Support," repositorio del proyecto, consultado en 2026. Disponible: https://github.com/pavel-odintsov/fastnetmon
[31] CERT Division, Software Engineering Institute, Carnegie Mellon University, "SiLK: System for Internet-Level Knowledge," versión 3.24.2, documentación oficial del proyecto, consultada en 2026. Disponible: https://tools.netsa.cert.org/silk/
[32] M. Sarhan, S. Layeghy y M. Portmann, "Towards a Standard Feature Set for Network Intrusion Detection System Datasets," Mobile Networks and Applications, vol. 27, 2022, doi: 10.1007/s11036-021-01843-0. Disponible: https://staff.itee.uq.edu.au/marius/NIDS_datasets/
[33] R. Sommer y V. Paxson, "Outside the Closed World: On Using Machine Learning for Network Intrusion Detection," en Proc. IEEE Symposium on Security and Privacy, Oakland, CA, EE. UU., pp. 305-316, 2010.
[34] D. Arp, E. Quiring, F. Pendlebury, A. Warnecke, F. Pierazzi, C. Wressnegger, L. Cavallaro y K. Rieck, "Dos and Don'ts of Machine Learning in Computer Security," en Proc. 31st USENIX Security Symposium, Boston, MA, EE. UU., pp. 3971-3988, 2022.
[35] Prometheus Authors, "Prometheus Documentation," documentación oficial del proyecto, 2024. Disponible: https://prometheus.io/docs/
[36] C. Pascoe, S. Quinn y K. Scarfone, "The NIST Cybersecurity Framework (CSF) 2.0," NIST Cybersecurity White Paper NIST CSWP 29, National Institute of Standards and Technology, Gaithersburg, MD, EE. UU., feb. 2024.
[37] G. Engelen, V. Rimmer y W. Joosen, "Troubleshooting an Intrusion Detection Dataset: the CICIDS2017 Case Study," en Proc. IEEE Security and Privacy Workshops (SPW), San Francisco, CA, EE. UU., 2021, doi: 10.1109/SPW53761.2021.00009.
[38] M. Lanvin, P.-F. Gimenez, Y. Han, F. Majorczyk, L. Mé y É. Totel, "Errors in the CICIDS2017 Dataset and the Significant Differences in Detection Performances It Makes," en Risks and Security of Internet and Systems (CRiSIS 2022), Lecture Notes in Computer Science, vol. 13857, Springer, pp. 18-33, 2023, doi: 10.1007/978-3-031-31108-6_2.
[39] F. Pedregosa et al., "Scikit-learn: Machine Learning in Python," Journal of Machine Learning Research, vol. 12, pp. 2825-2830, 2011.
[40] F. Pendlebury, F. Pierazzi, R. Jordaney, J. Kinder y L. Cavallaro, "TESSERACT: Eliminating Experimental Bias in Malware Classification across Space and Time," en Proc. 28th USENIX Security Symposium, Santa Clara, CA, EE. UU., pp. 729-746, 2019.
[41] S. Axelsson, "The Base-Rate Fallacy and the Difficulty of Intrusion Detection," ACM Transactions on Information and System Security, vol. 3, no. 3, pp. 186-205, 2000.
[42] IBM, "Network hierarchy," IBM QRadar SIEM Documentation, consultada en 2026. Disponible: https://www.ibm.com/docs/en/qsip/7.4?topic=phase-network-hierarchy
[43] B. E. Strom, A. Applebaum, D. P. Miller, K. C. Nickels, A. G. Pennington y C. B. Thomas, "MITRE ATT&CK: Design and Philosophy," The MITRE Corporation, McLean, VA, EE. UU., mar. 2020. Disponible: https://www.mitre.org/news-insights/publication/mitre-attck-design-and-philosophy
[44] M. Shafi, A. Habibi Lashkari y A. Haghighian Roudsari, "NTLFlowLyzer: Towards generating an intrusion detection dataset and intruders behavior profiling through network and transport layers traffic analysis and pattern extraction," Computers & Security, vol. 148, art. 104160, 2025. Disponible: https://www.yorku.ca/research/bccc/ucs-technical/cybersecurity-datasets-cds/
[45] R. Panigrahi y S. Borah, "IDS2025 (Balanced Intrusion Detection Evaluation Dataset)," Mendeley Data, v1, 2025, doi: 10.17632/pkskt3fv3v.1.
[46] Communications Security Establishment y Canadian Institute for Cybersecurity, "A Realistic Cyber Defense Dataset (CSE-CIC-IDS2018)," Registry of Open Data on AWS, 2018. Disponible: https://registry.opendata.aws/cse-cic-ids2018/
[47] L. K. Vashishtha y K. Chatterjee, "Strengthening cybersecurity: TestCloudIDS dataset and SparkShield algorithm for robust threat detection," Computers & Security, vol. 151, art. 104308, 2025, doi: 10.1016/j.cose.2024.104308.

[48] National Institute of Standards and Technology, "Artificial Intelligence Risk Management Framework (AI RMF 1.0)," NIST AI 100-1, Gaithersburg, MD, EE. UU., ene. 2023, doi: 10.6028/NIST.AI.100-1.
[49] P. Phaal, S. Panchen y N. McKee, "InMon Corporation's sFlow: A Method for Monitoring Traffic in Switched and Routed Networks," RFC 3176 (Informational), IETF, sep. 2001.
[50] OpenConfig, "gRPC Network Management Interface (gNMI) Specification," v0.10.0, may. 2023. Disponible: https://www.openconfig.net/docs/gnmi/gnmi-specification/
[51] E. Voit, A. Clemm, A. Gonzalez Prieto, E. Nilsen-Nygaard y A. Tripathy, "Subscription to YANG Notifications," RFC 8639, IETF, sep. 2019.
[52] A. Clemm y E. Voit, "Subscription to YANG Notifications for Datastore Updates," RFC 8641, IETF, sep. 2019.
[53] B. Trammell y E. Boschi, "Bidirectional Flow Export Using IP Flow Information Export (IPFIX)," RFC 5103, IETF, ene. 2008.

Anexo A: Campos NetFlow por Fabricante

Este anexo reúne el material de referencia sobre el protocolo NetFlow que el Capítulo 2 resume. La evolución de sus versiones, la estructura del mensaje con plantillas, el detalle de las combinaciones anómalas de banderas TCP empleadas en reconocimiento, y una descripción comparativa de los campos que exportan los fabricantes más relevantes del mercado, con énfasis en los utilizados para la detección de amenazas mediante aprendizaje automático.

A.1 Evolución del Protocolo y Estructura del Mensaje

A.1.1 Versiones y un poco de historia del Protocolo

NetFlow surgió a mediados de la década de 1990 como subproducto de una línea de trabajo distinta de aquella a la que terminó sirviendo. Los ingenieros de Cisco buscaban acelerar el procesamiento de las listas de control de acceso y la conmutación de paquetes, y para ello agruparon los paquetes que compartían origen, destino y puertos, tratándolos como una unidad continua de comunicación a la que denominaron flujo. La técnica fue descartada como mecanismo de conmutación y reemplazada por Cisco Express Forwarding, pero dejó tras de sí un residuo valioso. La memoria del enrutador conservaba un registro preciso de qué máquinas se comunicaban entre sí, información que los administradores de red no lograban obtener mediante SNMP con la granularidad requerida. Cisco habilitó entonces su exportación por UDP hacia un colector externo, y de esa reconversión nació NetFlow como instrumento de monitoreo.

La evolución posterior admite un resumen en cuatro etapas. NetFlow v1, liberado en 1996, constituyó el prototipo inicial y presentaba limitaciones considerables, pues operaba únicamente sobre IPv4 y omitía tanto las máscaras de subred como los números de sistema autónomo del protocolo BGP. Las versiones v2 a v4 permanecieron confinadas a los laboratorios del fabricante y nunca alcanzaron distribución comercial. NetFlow v5, publicado en 1998, corrigió esas carencias e incorporó los identificadores de sistema autónomo de origen y destino, dato indispensable para los proveedores de servicio porque permite reconstruir la trayectoria de tránsito de los datos y no solo el par de direcciones involucradas. Añadió además números de secuencia y fijó los siete campos que definen canónicamente un flujo, a saber, direcciones IP de origen y destino, puertos de origen y destino, protocolo, interfaz de entrada y tipo de servicio. Su robustez la convirtió en estándar de facto adoptado por el conjunto de la industria. Las variantes v7 y v8 fueron desarrollos menores orientados a los conmutadores Catalyst y a la agregación de datos para economizar memoria, y cayeron pronto en desuso.

El cambio cualitativo llegó con NetFlow v9, documentado por la IETF en el RFC 3954 [19] e introducido comercialmente a fines de 2004, que abandonó la estructura de campos fijos en favor de un esquema de plantillas dinámicas. Esa flexibilidad habilitó la exportación de tráfico IPv6, etiquetas MPLS y atributos orientados a seguridad. El éxito de la versión 9 llevó a la IETF a adoptar su arquitectura como base del estándar

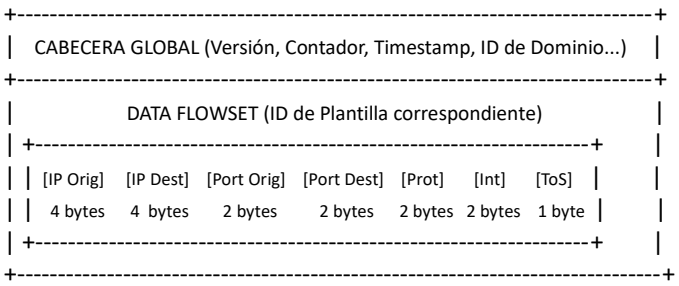
abierto IPFIX, especificado en el RFC 7011 [20] y conocido también como NetFlow v10. Sobre ese mecanismo de plantillas se construyen hoy las implementaciones de los demás fabricantes, entre ellas J-Flow de Juniper, NetStream de Huawei, la exportación NetFlow v9 de los cortafuegos FortiGate de Fortinet y el soporte IPFIX de los conmutadores empresariales de Aruba.

A.1.2 Estructura del mensaje

Tanto NetFlow v9 como IPFIX organizan la información de manera jerárquica. Un mensaje se compone de una cabecera global seguida de uno o más conjuntos de flujo. La cabecera transporta la versión del protocolo —9 para NetFlow v9 y 10 para IPFIX—, el recuento de registros, la marca de tiempo, el número de secuencia y el identificador del dominio de observación. Los conjuntos de flujo, ubicados a continuación, admiten tres formas. El conjunto de plantilla define la estructura y el orden de los campos que tendrán los registros posteriores. El conjunto de plantilla de opciones describe características del propio proceso de medición. El conjunto de datos transporta los valores medidos, dispuestos según la plantilla previamente anunciada.

De esta arquitectura se desprende una consecuencia operativa relevante. La séptupla que identifica el flujo no reside en la cabecera fija sino dentro del conjunto de datos, y su orden lo determina dinámicamente la plantilla correspondiente. El dispositivo exportador anuncia primero en qué disposición enviará los campos y solo después transmite los registros con esa misma disposición. Un colector que no haya recibido la plantilla vigente resulta por tanto incapaz de interpretar los datos que recibe, circunstancia que explica la necesidad de reenviarlas de forma periódica y que se retoma al describir la configuración de los exportadores.

Diagrama general de la estructura del paquete Netflow y las 7-tupla



A.2 Combinaciones Anómalas de Banderas TCP

El establecimiento normal de una sesión TCP sigue la secuencia SYN, SYN-ACK y ACK. Una sucesión masiva de paquetes SYN sin la confirmación correspondiente revela el intento de mapear los puertos abiertos de un segmento. Más reveladoras aún resultan las combinaciones que el estándar no contempla, empleadas para sondear defensas perimetrales que solo vigilan el patrón SYN convencional. Todas explotan la misma propiedad: un puerto cerrado responde con RST y uno abierto guarda silencio, de modo que el atacante infiere el estado del puerto sin completar ninguna conexión. La Tabla A.1 recoge las combinaciones más frecuentes.

Tabla A.1. Combinaciones anómalas de banderas TCP empleadas en sondeos de reconocimiento

Técnica	Banderas activas	Fundamento del sondeo
Xmas scan	FIN, PSH, URG	Exige procesar datos urgentes y entregarlos de inmediato a la aplicación mientras ordena cerrar la conexión, requerimiento contradictorio que ningún cliente legítimo formula
Null scan	Ninguna	Cabecera desprovista de toda señalización, situación imposible en una comunicación real, pues cualquier interacción requiere al menos una bandera de estado

SYN + FIN	SYN, FIN	Solicita abrir y cerrar la conexión en el mismo instante, con el propósito de confundir el seguimiento de estado de los cortafuegos de inspección
FIN scan	FIN	Ordena cerrar una sesión que nunca llegó a establecerse
SYN + RST	SYN, RST	Solicita la sincronización de la sesión y ordena abortarla de manera simultánea

A.3 Cisco NetFlow v5

NetFlow v5 define un formato fijo de 48 bytes por registro de flujo, sin mecanismo de plantillas y limitado a IPv4, cuyos campos recoge la Tabla A.2. Su vigencia merece precisión, porque condiciona el alcance del colector. Cisco no ha publicado una declaración formal de obsolescencia del formato, pero lo ha restringido en la práctica: en IOS-XE la exportación en v5 solo es admisible cuando el monitor de flujo emplea uno de los tres registros predefinidos que reproducen el NetFlow original, de modo que cualquier registro personalizado —requisito de todo caso de uso que involucre IPv6, MPLS, VLAN o clasificación de aplicación— obliga a v9 o IPFIX. Las plataformas Cisco que exportaban v5 mediante la sintaxis tradicional pertenecen además a generaciones ya fuera de soporte.

En consecuencia, este trabajo trata NetFlow v5 como modo de retrocompatibilidad del colector y no como formato primario del diseño, que se apoya en v9 e IPFIX. Mantener su soporte se justifica por dos motivos concretos: constituye el denominador común que acepta prácticamente todo colector desplegado, y sigue siendo exportado de forma nativa por equipos vigentes de gama de entrada, señaladamente MikroTik RouterOS.

Tabla A.2. Campos principales de Cisco NetFlow v5

Campo	Bytes	Descripción
srcaddr	4	Dirección IP de origen (IPv4)
dstaddr	4	Dirección IP de destino (IPv4)
nexthop	4	IP del siguiente salto de enrutamiento
input	2	Índice de interfaz SNMP de entrada
output	2	Índice de interfaz SNMP de salida
dPkts	4	Número total de paquetes del flujo
dOctets	4	Número total de bytes del flujo
First	4	Timestamp de inicio (ms desde boot del sistema)
Last	4	Timestamp de fin del flujo
srcport	2	Puerto de origen (TCP/UDP) o tipo ICMP
dstport	2	Puerto de destino (TCP/UDP) o código ICMP
tcp_flags	1	Unión (OR) de todos los flags TCP del flujo
prot	1	Número de protocolo IP (6=TCP, 17=UDP, 1=ICMP)
tos	1	Tipo de servicio IP del primer paquete
src_as	2	Número AS de origen (BGP)
dst_as	2	Número AS de destino (BGP)
src_mask	1	Longitud prefijo de dirección de origen
dst_mask	1	Longitud prefijo de dirección de destino

A.4 Cisco NetFlow v9 e IPFIX (RFC 7011)

NetFlow v9 introdujo un mecanismo basado en plantillas (templates) que permite la definición dinámica de campos exportados. El exportador envía FlowSets de plantillas describiendo la estructura de datos, seguidos de DataFlowSets con los valores. IPFIX (RFC 7011, IETF 2013) estandarizó este mecanismo con soporte para campos de empresa (Enterprise fields) y un registro centralizado de Information Elements (IEs).

Tabla A.3. Campos IPFIX relevantes para detección de amenazas (RFC 7011)

IE ID	Nombre	Descripción para Seguridad
1	octetDeltaCount	Bytes del flujo — indicador de volumen de transferencia
2	packetDeltaCount	Paquetes — ratio bytes/paquetes para caracterización
4	protocolIdentifier	Protocolo IP — filtra tráfico ICMP/UDP sospechoso
6	tcpControlBits	Flags TCP acumulados — detecta SYN floods, RST inusuales
7	sourceTransportPort	Puerto de origen — puertos ephemeral vs. bien conocidos
8	sourceIPv4Address	IP origen — análisis de reputación y geolocalización
11	destinationTransportPort	Puerto destino — detección de PortScan multi-puerto
12	destinationIPv4Address	IP destino — detección DDoS por concentración
21	flowEndSysUpTime	Timestamp fin — cálculo de duración del flujo
22	flowStartSysUpTime	Timestamp inicio — ventana temporal del flujo
32	icmpTypeCodeIPv4	Tipo y código ICMP — detección de ICMP flooding
150	flowStartSeconds	Timestamp inicio en segundos (época Unix)
231	initiatorPackets	Paquetes en dirección iniciador→respondedor (forward)
232	responderPackets	Paquetes en dirección respondedor→iniciador (backward)
233	initiatorOctets	Bytes en dirección forward
234	responderOctets	Bytes en dirección backward

A.5 Juniper J-Flow

Juniper Networks denomina J-Flow a su implementación de exportación de flujos. La documentación actual de Junos OS emplea preferentemente el término inline active flow monitoring, que designa el modelo vigente: el muestreo y la exportación se ejecutan en el motor de reenvío (PFE), sin requerir el hardware de servicios del esquema clásico. Los formatos soportados en ese modelo son IPFIX y NetFlow v9, y está disponible en las series MX, PTX, ACX, EX y SRX. J-Flow admite muestreo configurable (1:N paquetes) para acotar la carga en interfaces de alta velocidad, y provee extensiones propietarias para información de VRF y etiquetas MPLS, relevantes en entornos de proveedores de servicio.

Tabla A.4. Extensiones J-Flow de Juniper relevantes para seguridad

Campo	Descripción
jnxJflowLogSrcPrefix	Prefijo de red de origen con máscara — agrupación por subredes
jnxJflowLogDstPrefix	Prefijo de red de destino
jnxJflowLogMplsLabel1	Etiqueta MPLS top-of-stack — trazabilidad en redes MPLS
jnxJflowLogVrf	Nombre del VRF — segmentación por cliente multi-tenant
jnxJflowLogDirection	Dirección del flujo (ingress/egress) relativa a la interfaz

A.6 Huawei NetStream

Huawei implementa NetStream en sus equipos de las series NetEngine AR, NE y CE. La familia de acceso vigente es NetEngine AR —series AR600, AR6100, AR6200, AR6300 y AR8000—, que ejecuta VRP V300R0xx; la generación anterior AR G3, asociada a VRP V200R0xx, se encuentra en fin de vida. NetStream es compatible con NetFlow v5 y v9 y soporta IPFIX. A diferencia de lo ocurrido en otros fabricantes, la estructura de comandos de configuración se mantuvo estable entre ambas generaciones de VRP. Huawei añade extensiones para estadísticas de calidad de servicio (QoS) y diferenciación por clase de servicio (CoS), útiles

para correlacionar degradación de QoS con ataques de saturación que afectan el rendimiento de la red antes de ser detectados como anomalías de seguridad.

Tabla A.5. Extensiones propietarias Huawei NetStream relevantes para seguridad

Campo	Descripción
hwIfIndex	Índice de interfaz según estándar Huawei — mapeo a topología
hwSamplingInterval	Intervalo de muestreo configurado — afecta representatividad estadística
hwFlowSamplingMode	Modo de muestreo (determinístico/aleatorio) — impacta detección de ataques breves
hwVrfName	Nombre del VRF — correlación de flujos por segmento de red
hwQoSRemark	Reclasificación QoS aplicada — evidencia de manipulación de marcas CoS

A.7 Fortinet FortiGate (NetFlow v9)

Los firewalls FortiGate exportan NetFlow v9 desde FortiOS 5.2. Las versiones vigentes al momento de este trabajo son FortiOS 7.6, con soporte de largo plazo, y FortiOS 8.0, liberada en abril de 2026; la rama 5.x está fuera de soporte desde hace varios años y no corresponde considerarla en un despliegue de producción. La sintaxis de configuración cambió además de manera relevante: desde FortiOS 7.4.2 la definición del colector se anida en un bloque config collectors que admite varios destinos simultáneos, según se documenta en el Anexo E. Su implementación está orientada a la contabilidad de tráfico por política de seguridad, por lo que enriquece los registros con el identificador de la política aplicada y la clasificación de aplicación del motor de inspección, pero omite parte de las estadísticas direccionales presentes en los exportadores de enrutamiento puro.

Tabla A.6. Campos característicos de la exportación NetFlow de Fortinet FortiOS

Campo	Descripción	Valor para detección
fgtPolicyId	Identificador de la política de seguridad aplicada	Correlación entre flujo y regla que lo permitió
fgtAppId	Aplicación identificada por el motor de inspección	Detección de aplicaciones no autorizadas en la red
fgtSrcIntf / fgtDstIntf	Interfaces lógicas de entrada y salida	Trazabilidad de zona de seguridad origen y destino
fgtVdom	Dominio virtual del firewall	Segmentación en despliegues multi-tenant
octetTotalCount	Bytes totales del flujo	Volumetría; sin desglose por dirección
packetTotalCount	Paquetes totales del flujo	Ratio bytes/paquetes para caracterización

A.8 HPE Aruba (IPFIX y sFlow)

Corresponde advertir aquí una distinción que suele pasarse por alto al planificar la instrumentación. Los switches modulares HPE Aruba de la serie 5400R zl2, pertenecientes a la línea ProVision/AOS-S, no exportan NetFlow ni IPFIX: su mecanismo de telemetría de flujo es sFlow, sobre el puerto 6343/UDP, con la semántica de muestreo de paquetes analizada en la sección A.12. Un despliegue que los incorpore requiere en consecuencia un colector sFlow independiente, o bien un colector capaz de ambos protocolos, y no puede integrarlos sin más al socket NetFlow del diseño aquí descrito.

La familia vigente de HPE Aruba Networking es AOS-CX, que sí soporta IPFIX de forma nativa además de sFlow. Sus equipos de agregación y core —series CX 6400, sucesora modular del 5400R zl2, CX 8320, CX 8360 y CX 9300— se apegan estrechamente al registro de Information Elements de IANA, sin extensiones propietarias de relevancia, lo que simplifica la normalización: el perfil aplicable coincide en lo esencial con el del estándar IPFIX genérico. En las series de mayor densidad de puertos el muestreo es configurable, con factores habituales entre 1:100 y 1:10000, lo que obliga a aplicar el escalado descrito en la sección 2.5.

A.9 MikroTik RouterOS (Traffic Flow)

MikroTik implementa exportación de flujos bajo la denominación Traffic Flow, configurable en la rama /ip traffic-flow de RouterOS 7.x, con soporte de NetFlow v5, v9 e IPFIX. Es habitual en redes de menor escala y en proveedores regionales de servicios de Internet, y constituye el caso más relevante de equipo actualmente en producción que exporta NetFlow v5 de forma nativa, circunstancia que justifica mantener la compatibilidad con ese formato en el colector. Su particularidad para la normalización es que RouterOS exporta timestamps relativos al arranque del sistema aun en modo IPFIX, lo que obliga a la reconstrucción temporal descrita en la sección 2.5, y que el temporizador de flujos activos tiene por defecto un valor de treinta minutos que debe reducirse explícitamente para que la detección opere con latencia útil.

A.10 Cuadro Comparativo de Fabricantes

Tabla A.7. Comparación de capacidades de exportación por fabricante

Fabricante	Denominación	Versiones	Estadísticas direccionales	Muestreo	Perfil de normalización
Cisco	NetFlow / Flexible NetFlow	v5 (restringido), v9, IPFIX	Completas (FNF)	Opcional	Directo
Juniper	J-Flow / inline active flow monitoring	v9, IPFIX	Completas	Habitual 1:N	Escalado por muestreo
Huawei	NetStream	v5, v9, IPFIX	Completas	Opcional	Directo + IAT bwd imputado
Fortinet	NetFlow	v9	Parciales	No	Imputación de 2 campos
HPE Aruba (AOS-CX)	IPFIX	IPFIX	Completas	Configurable	Escalado por muestreo
MikroTik	Traffic Flow	v5, v9, IPFIX	Completas	Opcional	Reconstrucción temporal
HPE Aruba (5400R, AOS-S)	sFlow	—	No aplica	Obligatorio	Requiere colector sFlow independiente

A.11 Familias de Campos Relevantes para la Detección

Temporalidad —duración y marcas de tiempo—. Caracteriza el ritmo de la comunicación, dimensión en la que el software automatizado se distingue con nitidez de la actividad humana. El caso más ilustrativo es el beaconing: flujos muy breves, de escaso volumen y separados por intervalos muy regulares, patrón que ninguna persona reproduce al navegar. En el extremo opuesto, una duración anormalmente prolongada hacia un destino externo sugiere extracción sostenida de información.

Volumen y direccionalidad —paquetes y bytes por sentido—. Exponen las desviaciones respecto del caudal habitual y, sobre todo, distinguen una sesión conversacional legítima de una comunicación esencialmente unidireccional, propia tanto del barrido de reconocimiento como de la exfiltración.

Banderas de control TCP. Acumulan los indicadores observados durante la vida del flujo y separan las conexiones legítimas de los sondeos: una sucesión masiva de paquetes SYN sin la confirmación correspondiente revela el mapeo de puertos abiertos. La Tabla A.1 recoge además las combinaciones que el estándar no contempla y que persiguen el mismo fin eludiendo las defensas que solo vigilan el patrón SYN.

Textura —tiempo entre llegadas y dispersión de tamaños—. El tráfico originado por una persona es irregular por naturaleza; el generado por un proceso automático tiende a la regularidad. La minería de criptomonedas no autorizada ilustra el principio con un tamaño de paquete casi constante y varianza próxima a cero.

Métricas derivadas. Condensan en un solo valor relaciones que los campos individuales solo expresan por separado, y son por ello las de mayor rendimiento. El cociente entre bytes y paquetes caracteriza la naturaleza del flujo con independencia de su magnitud: valores bajos acompañan a la fuerza bruta y a la inundación SYN, valores altos sostenidos apuntan a transferencia masiva. El número de puertos distintos contactados desde una misma dirección es el indicador más directo del reconocimiento interno, y la asimetría del tráfico bidireccional distingue exfiltración de barrido según el sentido en que se produzca.

A.12 Alternativas de Telemetría Evaluadas

sFlow (RFC 3176) [49] muestrea uno de cada N paquetes en lugar de mantener estado por flujo, lo que lo hace sencillo en hardware y barato sobre el plano de control, pero le impide entregar duración, contadores completos y tiempos entre llegadas, que son precisamente las magnitudes sobre las que opera el clasificador. La telemetría de streaming, apoyada en gNMI [50] y YANG con notificaciones de los RFC 8639 [51] y 8641 [52], entrega información en milisegundos, pero solo está disponible en equipamiento reciente de gama alta y su objeto natural es el estado del dispositivo antes que el registro por flujo. La inspección profunda de paquetes (DPI) es la alternativa de mayor poder informativo y la más costosa en tres dimensiones a la vez: hardware dedicado, degradación del equipamiento en línea e implicancias legales por acceder al contenido. La captura íntegra en formato PCAP preserva cada paquete y es la referencia forense, pero su almacenamiento crece linealmente con el tráfico cursado y comparte las implicancias de privacidad de la anterior. Esta sección desarrolla cada alternativa.

Tabla A.8. Comparación de tecnologías de telemetría de red para detección de amenazas

Criterio	NetFlow / IPFIX	sFlow	Telemetría de streaming	DPI
Objeto de observación	Metadatos del tráfico IP: quién se comunica con quién, cuánto y cuándo (capas 3 y 4)	Muestra estadística de paquetes individuales	Estado y desempeño del propio dispositivo de red	Contenido y carga útil del paquete (capa 7)
Compatibilidad de hardware	Universal: routers, switches y firewalls de generaciones antiguas y actuales	Amplia, con predominio en equipos de conmutación de centro de datos	Limitada a equipamiento reciente con sistema operativo modular	Requiere procesadores dedicados, y cortafuegos de nueva generación
Costo de adquisición	Bajo: funcionalidad nativa, rara vez sujeta a licenciamiento adicional	Bajo: nativo en la mayoría del equipamiento compatible	Medio: habitualmente incluido en licencias base o intermedias	Alto: licencias por suscripción para actualización continua de firmas
Impacto sobre el equipo	Mínimo: el procesamiento de cabeceras consume pocos recursos	Mínimo y constante: independiente del número de flujos activos	Bajo: transporte eficiente mediante gRPC y Protocol Buffers	Elevado: la apertura de paquetes y el descifrado degradan el rendimiento
Latencia de la información	Diferida: minutos, según expiración del flujo o del temporizador activo	Cuasi inmediata para las muestras; agregación diferida	Tiempo real: milisegundos, con notificación ante cambios	Tiempo real: inspección en línea, paquete a paquete
Suficiencia para el modelo de interés	Alta: provee duración, volumen, direccionalidad y temporalidad por flujo	Parcial: los contadores por flujo deben estimarse desde muestras	Baja: no caracteriza conversaciones entre pares de máquinas	Muy alta, con información redundante respecto de la requerida
Riesgos a la Privacidad	Baja: no accede al contenido de las comunicaciones	Baja: solo fragmentos de cabecera	Muy baja: no observa tráfico de usuarios	Alta: exige interceptar y descifrar contenido
Uso característico	Auditoría de tráfico, detección de anomalías de comportamiento, contabilidad de consumo	Caracterización volumétrica en redes de alta velocidad	Automatización, detección de microcortes, monitoreo de infraestructura crítica	Seguridad perimetral, prevención de intrusiones, control de aplicaciones

Limitación principal	No observa el contenido ni identifica aplicaciones que enmascaran su puerto	El muestreo degrada la detección de flujos de bajo volumen	Incompatible con hardware antiguo; ajena a la caracterización del tráfico	Costo continúo elevado y exigencias legales del descifrado

Tabla A.9. Plataformas comerciales y de código abierto para detección de amenazas sobre telemetría de flujo

Plataforma	**Telemetría**	**Detección declarada**	**Mitigación**	**Mercado objetivo**	**Licenciamiento**
NETSCOUT Arbor Sightline + TMS	NetFlow, IPFIX, sFlow (muestreo ~1:1.000)	Líneas base estadísticas por objeto protegido; inteligencia ATLAS asistida por ML	Equipo TMS de limpieza; BGP Flowspec	Operadores, ISP, grandes empresas	Comercial, por capacidad; hardware dedicado
Nokia Deepfield Defender	Flujo, BGP, DNS de cualquier fabricante	Analítica a escala de petabytes; modelos ML de tráfico normal; contexto Secure Genome	Filtrado en los propios routers; sistema de mitigación 7750	Operadores, IXP, nube	Comercial; integrado al ecosistema Nokia
Genie Networks GenieATM	Telemetría de flujo de routers y conmutadores	Motor de anomalías de comportamiento con aprendizaje de líneas base; hasta 24 M flujos/s	BGP / Flowspec hacia centro de limpieza	Operadores e ISP	Comercial, por capacidad
Radware DefenseFlow + DefensePro	NetFlow, estadísticas xFlow	Líneas base adaptativas por objeto protegido; motor de comportamiento patentado	Equipos DefensePro en línea o en centro de limpieza; BGP / SDN	Operadores, ISP, nube, empresas	Comercial; hardware dedicado
Cisco Secure Network Analytics	NetFlow, IPFIX (ecosistema Cisco)	Aprendizaje no supervisado de líneas base por entidad	Integración con el ecosistema Cisco	Empresas	Comercial, por flujos/s
Kentik	NetFlow, sFlow, IPFIX como servicio	Detección de anomalías asistida por ML	Orquestación hacia terceros	Equipos de red	Suscripción SaaS
FastNetMon / ntopng	sFlow, NetFlow, IPFIX, espejo de puerto	Umbrales configurables por host	BGP blackhole / Flowspec	ISP pequeños, hosting	Código abierto
Sistema propuesto	NetFlow v5/v9, IPFIX sin muestreo	Clasificador supervisado XGBoost por flujo, con importancias de características auditables	Alertamiento y tableros; mitigación no incluida	Organización mediana	Sin licencia; código propio desplegable con Docker

Anexo B: Esquema de Base de Datos

El esquema de base de datos fue diseñado para soportar tanto el almacenamiento eficiente de flujos NetFlow en tiempo real como las consultas analíticas de los dashboards Grafana y las APIs de seguridad.

B.1 Tabla Principal: netflow_data

Tabla B.1. Definición de la tabla netflow_data

Columna	Tipo PostgreSQL	Descripción
id	BIGSERIAL PK	Identificador único autoincremental
start_time	TIMESTAMPTZ NOT NULL	Inicio del flujo con zona horaria
end_time	TIMESTAMPTZ	Fin del flujo
created_at	TIMESTAMPTZ DEFAULT NOW()	Timestamp de inserción en BD
src_ip	INET NOT NULL	IP de origen (soporta IPv4 e IPv6)
dst_ip	INET NOT NULL	IP de destino
src_port	INTEGER DEFAULT 0	Puerto de origen
dst_port	INTEGER DEFAULT 0	Puerto de destino
protocol	TEXT DEFAULT unknown	Protocolo (TCP, UDP, ICMP, etc.)
flags	TEXT	Flags TCP del flujo (OR acumulativo)
bytes	BIGINT DEFAULT 0	Bytes totales del flujo
packets	INTEGER DEFAULT 0	Paquetes totales del flujo
duration	FLOAT	Duración en segundos (check >= 0)
bytes_per_second	FLOAT	Tasa de transferencia derivada
packets_per_second	FLOAT	Tasa de paquetes derivada
avg_packet_size	FLOAT	Tamaño medio de paquete en bytes
is_anomaly	BOOLEAN DEFAULT false	Indicador de anomalía detectada por ML
threat_score	FLOAT [0,1]	Score de amenaza del modelo XGBoost
total_fwd_packets	INTEGER	Paquetes en dirección forward (para ML)
total_backward_packets	INTEGER	Paquetes en dirección backward
total_length_of_fwd_packets	BIGINT	Bytes totales en dirección forward
total_length_of_bwd_packets	BIGINT	Bytes totales en dirección backward
fwd_packet_length_mean	FLOAT	Tamaño medio paquetes forward
bwd_packet_length_mean	FLOAT	Tamaño medio paquetes backward
flow_iat_mean	FLOAT	IAT medio entre paquetes del flujo
fwd_iat_mean	FLOAT	IAT medio entre paquetes forward
bwd_iat_mean	FLOAT	IAT medio entre paquetes backward
fwd_packet_length_std	FLOAT	Desv. estándar tamaño paquetes forward
labels	JSONB DEFAULT {}	Metadatos adicionales de clasificación
connection_id	TEXT	ID de conexión para correlación multi-fuente
exporter_ip	INET	Dirección del exportador que envió el registro
file_source	TEXT	Archivo nfcapd de origen del flujo

B.2 Índices Optimizados sobre netflow_data

Tabla B.2. Índices implementados y su justificación

Índice	Tipo	Columnas	Uso Principal
netflow_time_brin_idx	BRIN	start_time	Consultas temporales sobre series de tiempo; 100x más compacto que B-Tree
netflow_src_ip_hash_idx	HASH	src_ip	Búsquedas de igualdad exacta sobre IP origen (más eficiente que B-Tree)
netflow_dst_ip_hash_idx	HASH	dst_ip	Búsquedas de igualdad exacta sobre IP destino
netflow_network_flow_idx	B-Tree compuesto	src_ip, dst_ip, protocol, start_time	Consultas de correlación de conversaciones bidireccionales con filtro temporal
netflow_protocol_idx	B-Tree parcial	protocol WHERE IN (TCP,UDP,ICMP,HTTP,HTTPS)	Filtrado por protocolos comunes (reducción de índice para protocolos frecuentes)
netflow_anomaly_idx	B-Tree parcial	is_anomaly, threat_score WHERE is_anomaly=true	Recuperación eficiente de amenazas (<5% del total de flujos)

B.3 Política de Retención

La política de retención enunciada en la sección 3.5 se aplica por tipo de dato conforme a la Tabla B.3, y se ejecuta mediante una función almacenada en PL/pgSQL que elimina por lotes los registros anteriores al umbral correspondiente.

Tabla B.3. Política de retención aplicada por tipo de dato

Tipo de dato	Retención	Justificación
Flujos individuales (netflow_data)	90 días	Ventana suficiente para investigación forense; controla el crecimiento de la tabla
Amenazas confirmadas (detected_threats)	1 año	Insumo para análisis de tendencias y reentrenamiento del modelo
Agregaciones horarias	1 año	Base de comparación para detección de desviaciones estacionales
Archivos binarios nfcapd	7 días	Solo necesarios para reprocesamiento ante fallos del pipeline
Métricas de infraestructura (Prometheus)	200 horas	Diagnóstico operacional de corto plazo

Anexo C: Pipeline ML — Descripción Técnica

Este anexo reúne el material técnico que el cuerpo del informe resume, a saber, la taxonomía de los algoritmos de aprendizaje automático aplicables al análisis de tráfico, la descripción detallada de los transformadores personalizados, la estructura del objeto serializado que se despliega en operación, los hiperparámetros finalmente adoptados y el conjunto definitivo de características. El desbalance de clases se corrige mediante SMOTE dentro del ImbPipeline, razón por la cual no se emplea el hiperparámetro scale_pos_weight de XGBoost: ambos mecanismos abordan el mismo problema y su aplicación simultánea sobrecompensaría la clase minoritaria.

C.1 Familias de Algoritmos de Aprendizaje Automático

La Inteligencia Artificial y el Aprendizaje Automático (Machine Learning) se dividen a grandes rasgos en cuatro grandes familias según la forma en que los algoritmos procesan los datos y aprenden de ellos. A continuación, se presenta la taxonomía general de la IA para aterrizar posteriormente en los algoritmos específicos de ciberseguridad con los que hemos abordado la problemática de ciberseguridad de este trabajo.

Las Cuatro Familias Generales de Algoritmos

1. Aprendizaje Supervisado: El algoritmo aprende utilizando un conjunto de datos histórico que ya contiene etiquetas de las respuestas correctas. Se le muestran miles de ejemplos que dicen "esto es tráfico normal" y "esto es un ataque". Su objetivo es aprender la regla matemática para clasificar correctamente los datos nuevos en el futuro.

2. Aprendizaje No Supervisado: El algoritmo recibe datos brutos sin ninguna etiqueta ni respuesta correcta. Su trabajo es explorar la estructura de los datos por sí mismo, agrupando elementos similares o detectando cuáles piezas de información se alejan del comportamiento común de la masa de datos.

3. Aprendizaje por Refuerzo: El algoritmo aprende a tomar decisiones interactivas mediante un sistema de recompensas y castigos en un entorno dinámico (es el método utilizado para que las IA aprendan a jugar ajedrez o a manejar autos autónomos). Su aplicación en telemetría de red pura es menos frecuente.

4. Aprendizaje Profundo (Deep Learning): Es una subcategoría avanzada que utiliza estructuras llamadas Redes Neuronales Artificiales que imitan vagamente el cerebro humano. Son ideales cuando los datos de origen no son estructurados (como imágenes, videos o audio) o cuando los patrones en los números son extremadamente complejos y sutiles

C.2 Algoritmos Relevantes para Detección de Amenazas en Redes

Cuando aplicamos este mapa general de algoritmos al análisis de tráfico IP (NetFlow/IPFIX), las derivaciones más eficientes son las siguientes:

A. Desde el Aprendizaje Supervisado (Especialidad: Clasificación)
   Cuando una empresa tiene un registro histórico de ataques conocidos (como bases de datos de escaneos previos, alertas confirmadas de ransomware o DDoS), utilizas algoritmos de Clasificación. Su tarea es binaria al determinar si un flujo es Seguro o Malicioso.
   Por ejemplo, Random Forest y XGBoost toman las métricas estructuradas de NetFlow, como la combinación anómala de flags TCP (FIN+PSH+URG) de un escaneo Xmas, o el ratio bajo de bytes/paquetes, y las procesan a través de múltiples árboles de decisión para dar un veredicto ultra rápido y preciso.

B. Desde el Aprendizaje No Supervisado (Especialidad: Detección de Anomalías)
En redes reales el tráfico legítimo cambia constantemente y los atacantes introducen técnicas nuevas de manera continua (ataques de día cero). Al no disponerse de etiquetas que anticipen lo desconocido, el algoritmo se configura para aprender la noción de normalidad propia de la red observada. Por tanto, todo aquello que rompa ese patrón se señala como anomalía.

Uno de los algoritmos es Isolation Forest, el cual aísla de forma matemática y veloz los flujos que tienen un comportamiento extraño, como una IP interna que repentinamente contacta a cientos de puertos únicos en un intento de movimiento lateral. Es el algoritmo que se utiliza en el prototipo como detector complementario.

Otro destacable es One-Class SVM (Support Vector Machines) que se vuelve experto en la "firma de normalidad" de una red corporativa. Si un malware de comando y control (C2) genera un beaconing con un Timestamp de inicio rígidamente automatizado, el algoritmo lo detecta de inmediato por no encajar en la rutina caótica humana.

C. Desde el Aprendizaje Profundo (Especialidad: Patrones Complejos)
Autoencoders son un tipo de redes neuronales no supervisadas que comprimen los datos de tráfico normal. Si el tráfico entrante es un ataque avanzado que usa Jitter (deslizamientos aleatorios de tiempo) para camuflarse, la red no logra procesarlo de forma limpia y genera un error de reconstrucción que activa la alerta en el centro de operaciones de seguridad (SOC).

C.3 Transformadores Personalizados: Descripción Detallada

En Ingeniería de características el uso de transformadores ha sido fundamental. El primer transformador, NetworkFeatureTransformer, opera sobre el identificador de conexión —la concatenación de dirección IP de origen, dirección IP de destino, puerto de origen, puerto de destino y protocolo— y lo descompone en sus cinco componentes para derivar cuatro grupos de características fundadas en conocimiento del dominio de seguridad.

En primer lugar, determina si cada dirección IP pertenece a los rangos privados definidos por el RFC 1918 (10.0.0.0/8, 172.16.0.0/12 y 192.168.0.0/16), generando los indicadores binarios src_is_private y dst_is_private. La combinación de origen público con destino privado constituye una señal potencial de acceso externo hacia la red interna, mientras que la combinación inversa puede corresponder a exfiltración.

En segundo lugar, verifica si los puertos de origen o de destino figuran en un conjunto de puertos históricamente asociados a servicios de administración remota, a compartición de archivos y a bases de datos: SSH (22/TCP), Telnet (23/TCP), RDP (3389/TCP), SMB (445/TCP), MSSQL (1433/TCP) y HTTP alternativo (8000/TCP y 8080/TCP). El resultado se expresa en los indicadores src_port_suspicious y dst_port_suspicious.

En tercer lugar, calcula port_difference, la diferencia absoluta entre ambos puertos. Su poder discriminativo se explica por la asimetría del modelo cliente-servidor convencional. El cliente abre un puerto efímero de numeración alta —habitualmente entre 49152 y 65535— y contacta un puerto de servicio bajo y fijo, como el 80 o el 443, de modo que la diferencia absoluta tiende a ser grande. En cambio, ciertas técnicas de reconocimiento fijan el puerto de origen en un valor bajo y recorren secuencialmente puertos de destino contiguos, produciendo diferencias pequeñas y una distribución concentrada que el modelo puede distinguir del patrón de navegación habitual.

Por último, codifica el protocolo como valor numérico (protocol_type) y elimina la columna de conexión original, cuya información ya quedó representada en las características derivadas. Cabe anticipar que esta derivación resulta redundante en el escenario de entrenamiento. El transformador reconstruye el protocolo desde la 5-tupla porque en operación la API puede recibir el identificador de conexión sin el campo de protocolo por separado; al entrenar sobre CICIDS2017, en cambio, ambos están presentes, de modo que incorporar protocol_type al espacio de entrada equivaldría a entregar el mismo valor por dos vías. La verificación de la sección 6.3 confirma la coincidencia, y por esa razón el atributo se calcula pero no se incorpora al conjunto de

características del modelo. Cabe destacar que este diseño evita deliberadamente incorporar las direcciones IP concretas como característica. Hacerlo permitiría al modelo memorizar los hosts atacantes del conjunto de entrenamiento en lugar de aprender el comportamiento que los caracteriza, forma de fuga de información advertida en la sección 4.6.

El segundo transformador, FeatureEngineeringTransformer, deriva métricas de comportamiento a partir de las relaciones entre los campos ya normalizados. En el plano cuantitativo calcula tres razones. La primera, bytes_per_packet, expresa el volumen medio por paquete y separa los flujos de control —numerosos paquetes pequeños, característicos de la fuerza bruta y de la inundación SYN— de las transferencias masivas propias de la exfiltración. La segunda métrica, fwd_bwd_ratio, mide la proporción entre el tráfico enviado y el recibido; su desequilibrio extremo delata comunicaciones esencialmente unidireccionales, ajenas al patrón conversacional de una sesión legítima. La tercera, packet_size_variation, es el cociente entre la desviación estándar y la media del tamaño de los paquetes en sentido directo, y distingue el tráfico interactivo humano, de alta variabilidad, del tráfico automatizado de tamaño rígido. Calcula, además, dos tasas por unidad de tiempo, paquetes y bytes por segundo de duración (flow_packets_per_duration y flow_bytes_per_duration).

En el plano temporal, cuando el registro incluye marca de tiempo, deriva tres atributos de granularidad gruesa: la hora del día (HourOfDay), el día de la semana (DayOfWeek) y el indicador IsBusinessHours, que sitúa el flujo dentro o fuera del horario laboral definido entre las 08:00 y las 18:00. El propósito de estos atributos es permitir que el modelo aprenda una distinción que ningún umbral de volumen captura por sí solo, pues un mismo volumen de transferencia puede ser rutinario a media mañana y anómalo de madrugada. Se deriva deliberadamente la hora y el día, y nunca el instante absoluto ni el minuto, por las razones que se exponen en la sección 4.6. Debe advertirse, con todo, que el análisis de la sección 6.3 obligó finalmente a descartar también estos tres atributos: en CICIDS2017 el día de la semana está asociado de manera casi determinista a la presencia de ataque, de modo que transporta información de la etiqueta antes que del comportamiento del tráfico.

Todas las derivaciones se calculan de forma condicional, únicamente si sus insumos están presentes en el registro. Esta decisión de implementación confiere al transformador tolerancia frente a exportadores con formatos incompletos —requisito directo de la arquitectura multi-dispositivo del Capítulo 3— y evita que la ausencia de un campo interrumpa el procesamiento del lote completo.

C.4 Arquitectura de la Pipeline

El pipeline de inferencia se despliega como un único objeto serializado con joblib, construido con ImbPipeline de imbalanced-learn, que encapsula todas las etapas y el modelo final. Su estructura encadena los dos transformadores personalizados, NetworkFeatureTransformer y FeatureEngineeringTransformer; un ColumnTransformer que imputa por mediana y escala con RobustScaler las variables numéricas; la etapa SMOTE, que actúa solo durante el ajuste, y el clasificador XGBClassifier. La serialización del pipeline completo hace innecesario reproducir el código de preprocesamiento en el servicio de producción: el archivo .pkl contiene tanto los parámetros aprendidos —estadísticos de imputación y escalado— como la lógica de transformación de las clases personalizadas.

C.5 Hiperparámetros del Modelo XGBoost

Tabla C.1. Hiperparámetros del modelo XGBoost final (optimizados por GridSearchCV)

Hiperparámetro	Grilla explorada / valor	Justificación
n_estimators	{200, 300} → 300	Número de árboles; equilibrio entre desempeño y velocidad de inferencia
max_depth	{4, 6} → 6	Profundidad máxima; controla la complejidad del modelo
learning_rate (eta)	{0,05, 0,1} → 0,1	Tasa de aprendizaje; convergencia estable con 300 estimadores

subsample	0,8	Fracción de muestras por árbol; regularización por submuestreo
colsample_bytree	0,8	Fracción de características por árbol; regularización de características
objective	binary:logistic	Clasificación binaria con salida en probabilidad
eval_metric	logloss	Log-pérdida binaria; penaliza el error emitido con alta confianza
tree_method	hist	Agrupación en histogramas; reduce tiempo y memoria sobre 1,8 M de filas
n_jobs	−1	Uso de todos los núcleos disponibles
random_state	42	Semilla de los procesos estocásticos; reproducibilidad exacta

C.6 Conjunto Definitivo de Características

El modelo desplegable opera sobre las veinte características que sobrevivieron al estudio de exclusión progresiva de la sección 6.3. Son los doce campos tomados de la exportación —flow duration, total fwd packets, total backward packets, total length of fwd packets, total length of bwd packets, fwd packet length mean, bwd packet length mean, flow IAT mean, fwd IAT mean, bwd IAT mean, fwd packet length std y protocol— junto a ocho derivadas: src_port_suspicious, dst_port_suspicious, port_difference, bytes_per_packet, fwd_bwd_ratio, packet_size_variation, flow_packets_per_duration y flow_bytes_per_duration. Quedan excluidas src_is_private y dst_is_private, por describir la topología del banco de pruebas del corpus, y HourOfDay, DayOfWeek e IsBusinessHours, por transportar información del calendario del experimento. El atributo protocol_type, que el transformador deriva desde la 5-tupla, tampoco forma parte del espacio de entrada del modelo: reconstruye el mismo valor que el campo protocol ya aporta, de modo que se calcula por compatibilidad con el escenario de producción, pero no se entrega al clasificador.

A los atributos temporales HourOfDay, DayOfWeek e IsBusinessHours concurre una segunda razón de exclusión, de orden técnico. Su cálculo se apoya en la inferencia del formato de fecha a partir del primer registro del subconjunto procesado, de modo que el valor obtenido para un mismo flujo puede variar según el lote en que se lo evalúe. La sección C.6.1 documenta la medición de ese efecto y sus consecuencias sobre las métricas. Un atributo cuyo valor no es determinista respecto del lote de inferencia no admite despliegue en producción.

C.6.1 Examen de las Cinco Características Descartadas

Bajo partición temporal, el modelo de veinticinco características supera al de veinte en +0,0109 de F1-Score, con dos cifras llamativas: una precisión de 1,0000, con solo dos falsos positivos, y un AUC-ROC de 0,9474. El examen del mecanismo muestra que esa ventaja no debe tomarse al pie de la letra, porque la causa es un defecto de parseo y no una mejora discriminativa. El transformador infiere el formato de fecha a partir del primer registro del subconjunto que recibe, de modo que el formato inferido depende de qué subconjunto se le entregue. Al ajustar sobre los tres primeros días la inferencia no resuelve la mayoría de los registros y DayOfWeek toma solo dos valores: el −1 de relleno, presente en 1.138.612 flujos de los cuales el 23,4 % es malicioso, y el 0, presente en 529.918 flujos entre los que no hay un solo ataque. El modelo aprende, en consecuencia, que el valor 0 equivale a tráfico benigno. Al predecir sobre los dos últimos días la inferencia resuelve de otro modo y el atributo toma los valores 3 y 4, que el modelo nunca observó; los árboles los encaminan por la rama de los valores altos, la que aprendió del 0, y el modelo declara benigno casi todo lo que evalúa. Ese sesgo explica las dos cifras llamativas: los dos únicos falsos positivos no expresan precisión excepcional sino renuencia a señalar amenazas, y el AUC-ROC se obtiene sobre un ordenamiento que un atributo fuera de distribución desplazó en bloque. La mejora de +0,0109 no es aporte informativo, sino el efecto colateral de un artefacto que en

producción se manifestaría como ceguera ante el tráfico malicioso. El hallazgo refuerza, por una vía no prevista, la decisión de excluir los atributos temporales: a su condición de portadores de información de la etiqueta se añade que su cálculo no es determinista respecto del subconjunto evaluado, propiedad inadmisible en producción.

Un examen análogo se aplicó a los dos indicadores de topología, con resultado distinto. Se comprobó primero si actúan como sustituto de la jornada, lo que haría de ellos un segundo canal de fuga temporal: la distancia de variación total entre la distribución de combinaciones del jueves y la del viernes es de 0,126, magnitud pequeña que descarta esa hipótesis. Lo que sí ocurre es que la asociación que el modelo aprende de ellos es propia del corpus: dentro del viernes, la combinación privado→privado concentra un 55,83 % de tráfico malicioso frente a valores del orden del 1 % en las restantes, con lo cual el modelo aprende que el tráfico interno es sospechoso. En una red empresarial la relación se invierte, porque el tráfico interno es el legítimo y la amenaza proviene habitualmente del exterior. Su exclusión se funda, por tanto, no en una fuga de la etiqueta sino en la inversión del signo de lo aprendido al cambiar de entorno, y de ahí que, de reincorporarse, deban reexpresarse como dirección respecto del perímetro declarado.

El régimen de partición temporal de la sección 6.3.1 descompone la cobertura del clasificador por familia de ataque. La Tabla C.2 recoge ese desglose, que distingue las familias para las que el entrenamiento ofrecía un análogo conductual de aquellas cuya firma solo se manifiesta en el agregado de muchos flujos.

Tabla C.2. Cobertura del clasificador por familia de ataque bajo partición temporal

Familia de ataque	Flujos en prueba	Detectados	Cobertura
DDoS	128.027	81.393	63,57 %
PortScan	158.930	229	0,14 %
Bot	1.966	0	0,00 %
Infiltration	36	0	0,00 %
Total	288.959	81.622	28,25 %

C.7 Reproducibilidad del Entrenamiento

El procedimiento completo de entrenamiento y evaluación está documentado en un cuaderno ejecutable que reproduce, en el orden en que se exponen en este informe, la carga y depuración del corpus, la construcción del pipeline, la verificación anti-fuga de la sección 4.6, el estudio de exclusión progresiva, la evaluación bajo ambos regímenes de partición y el cálculo de la carga de triaje. Todas las cifras consignadas en el Capítulo 6 provienen de su ejecución. Los procesos estocásticos —partición, remuestreo y clasificador— comparten la semilla 42, de modo que la ejecución del cuaderno reproduce los valores reportados. El cuaderno incorpora además una parte final dedicada al contraste entre los modelos de veinticinco y de veinte características, al diagnóstico del parseo no determinista de la marca temporal, al recorrido del umbral de decisión y a la serialización del artefacto desplegable, cuyos resultados se exponen en las secciones 6.3 y 6.3.1.

C.7.1 Serialización del Pipeline y Módulo de Transformadores

El modelo final —el pipeline completo, con todas sus etapas y parámetros aprendidos— se serializó mediante joblib. Serializar el pipeline y no únicamente el clasificador es lo que garantiza que la inferencia reproduzca exactamente el preprocesamiento del entrenamiento, incluidos el orden de los transformadores y los estadísticos de imputación y escalado. Una precisión de implementación resulta necesaria por sus consecuencias operacionales: las bibliotecas de serialización de Python guardan las clases personalizadas por referencia al módulo en que están definidas, de modo que si los transformadores se declaran dentro del propio cuaderno su módulo es __main__ y el artefacto no puede reabrirse en otro proceso. Por eso ambos se declaran en un módulo independiente, custom_transformers_v2.py, que se despliega junto al artefacto en el contenedor de inferencia.

El procedimiento incluye una verificación posterior a la escritura: el archivo se recarga y se comprueba que reproduce, predicción por predicción, las del modelo en memoria.

C.8 Notación de la Función Objetivo de XGBoost

La sección 2.3 del cuerpo presenta la función objetivo de XGBoost en su forma conceptual, expresión (2), como suma del error de predicción y del costo por complejidad del modelo. Escrita con la notación habitual de la literatura, esa misma expresión adopta la forma:

$$\text{Obj} = \Sigma_i \, l(y_i, \hat{y}_i) \; + \; \Sigma_k \, \Omega(f_k) \qquad (3)$$

donde el término de complejidad de cada árbol se define como:

$$\Omega(f) = \gamma \cdot T + \tfrac{1}{2} \cdot \lambda \cdot \|w\|^2 \qquad (4)$$

El primer término de la expresión (3) empuja al modelo a ser preciso y el segundo lo empuja a mantenerse simple, de modo que el entrenamiento consiste en encontrar el equilibrio entre ambas fuerzas. La Tabla C.3 detalla el significado de cada símbolo.

Tabla C.3. Significado de los símbolos de las expresiones (3) y (4)

Símbolo	Nombre	Qué representa en términos simples
y_i	Etiqueta real	La respuesta correcta conocida para la muestra i: 0 si el flujo es benigno, 1 si es malicioso
$\hat{y}_i$	Predicción del modelo	Lo que el modelo estima para esa misma muestra; el acento circunflejo indica "valor estimado"
$l(y_i, \hat{y}_i)$	Función de pérdida	Medida de cuánto se equivocó el modelo en la muestra i: vale cero si acertó y crece a medida que se aleja de la respuesta correcta
f_k	Árbol k-ésimo	Cada uno de los árboles de decisión que componen el conjunto
$\Omega(f_k)$	Penalización de complejidad	El "precio" que se cobra al árbol k por lo elaborado que es
T	Número de hojas	Cuántas conclusiones distintas puede emitir el árbol; más hojas equivale a un árbol con más ramificaciones y más detalle
w	Pesos de las hojas	El valor numérico que el árbol asigna en cada una de sus conclusiones
$\|w\|^2$	Suma de los pesos al cuadrado	Crece rápidamente si alguna hoja emite un valor desproporcionadamente grande
γ (gamma)	Peaje (costo) por hoja	Cuánto se cobra por cada hoja nueva; a mayor γ, más conservador el árbol
λ (lambda)	Freno a valores extremos	Cuánto se penaliza que una hoja emita valores muy grandes; a mayor λ, predicciones más moderadas

C.9 Mecanismo de Regularización y Eficiencia de XGBoost

Un refinamiento adicional explica la eficiencia del algoritmo: para decidir los puntos de corte agrupa previamente los valores numéricos en histogramas en lugar de evaluar cada valor posible, lo que reduce sustancialmente el tiempo de entrenamiento sobre millones de registros. A ello se suman tres mecanismos que refuerzan la generalización: el shrinkage (learning_rate), que reduce la contribución de cada árbol nuevo, y el submuestreo de filas (subsample) y de columnas (colsample_bytree), que introducen variabilidad entre árboles.

Tabla C.4. Importancia por ganancia de las características del modelo entrenado con partición aleatoria (cinco primeras)

Característica	Importancia	Naturaleza de la información
protocol	0,443	Protocolo de transporte
src_is_private	0,151	Topología del banco de pruebas
dst_is_private	0,139	Topología del banco de pruebas

fwd packet length mean	0,048	Comportamiento del flujo
bwd packet length mean	0,039	Comportamiento del flujo

Tabla C.5. Lectura operacional de la carga de triaje

Carga de triaje	**Qué significa para el analista**
1,00	Situación ideal: todo evento presentado es una amenaza real y no se descarta ninguno
1,11	De cada diez hallazgos, se descarta aproximadamente un evento sin valor (sistema propuesto, partición temporal del corpus)
2,00	La mitad del tiempo de revisión se invierte en eventos que no son amenazas
3,43	Dos de cada tres revisiones son infructuosas; equivale a revisar todo el tráfico sin filtro alguno
3,60	Reglas estáticas, partición temporal del corpus: revisar sin filtro resulta ligeramente más eficiente
10,00	Nueve de cada diez revisiones no conducen a ningún hallazgo

Tabla C.6. Síntesis de las dos cotas de desempeño del modelo desplegable de veinte características

Dimensión	**Cota superior (partición aleatoria)**	**Cota inferior (partición temporal)**
Supuesto operacional	Las amenazas pertenecen a familias conocidas	Las amenazas pertenecen a familias no observadas
Características empleadas	20 (sin topología ni atributos temporales)	20 (sin topología ni atributos temporales)
F1-Score	0,9681	0,4285
Precisión	0,9419	0,8873
Cobertura	0,9957	0,2825
Carga de triaje (umbral 0,7)	1,046	1,107
Reducción frente a reglas estáticas	71,3 %	69,3 %

Anexo D: Verificación y Diagnóstico de la Captura Multi-Dispositivo

Este anexo documenta el procedimiento operativo de verificación y diagnóstico de la captura, aplicado en el laboratorio en su parte ya operativa y especificado como procedimiento previsto en la parte que depende de mecanismos aún en diseño (deduplicación entre exportadores y validación cruzada). El esquema del prototipo identifica al exportador por su dirección, exporter_ip, porque el identificador interno que asigna el colector por orden de llegada (exporter_id) cambia al reiniciarlo; así deben leerse las consultas que siguen. Las que emplean observation_count corresponden al esquema extendido de la siguiente iteración.

D.1 Verificación de Recepción por Exportador

El primer control consiste en confirmar que el colector recibe tráfico desde cada dirección IP esperada. La siguiente consulta agrupa los flujos recibidos en la última hora por exportador de origen, permitiendo detectar de inmediato un dispositivo que dejó de exportar:

```
SELECT exporter_id,
COUNT(*)              AS flujos,
MIN(start_time)       AS primer_flujo,
MAX(start_time)       AS ultimo_flujo
FROM   netflow_data
WHERE  start_time >= NOW() - INTERVAL '1 hour'
GROUP  BY exporter_id
ORDER  BY flujos DESC;
```

Un exportador ausente de este resultado indica pérdida de telemetría, cuyas causas más frecuentes son el bloqueo del puerto 9995/UDP por una ACL intermedia, el reinicio del equipo con pérdida de la configuración no guardada, o el vencimiento del temporizador de plantillas en exportadores v9 sin reenvío periódico.

D.2 Verificación de Deduplicación

La siguiente consulta cuantifica cuántos flujos fueron observados por más de un exportador, métrica que valida el funcionamiento del mecanismo de correlación cruzada:

```
SELECT observation_count,
COUNT(*)  AS flujos,
ROUND(100.0 * COUNT(*) / SUM(COUNT(*)) OVER (), 2) AS porcentaje
FROM   netflow_data
WHERE  start_time >= NOW() - INTERVAL '24 hours'
GROUP  BY observation_count
ORDER  BY observation_count;
```

Una vez implementada la deduplicación, la proporción de flujos con observation_count mayor o igual a dos constituirá el indicador directo del grado de solapamiento entre los puntos de captura, y su valor esperado depende de la topología instrumentada. Una caída abrupta de ese porcentaje respecto de su línea base señalará desincronización horaria entre exportadores, que impide que la ventana de tolerancia agrupe correctamente las observaciones.

D.3 Diagnóstico de Sincronización Temporal

La deriva entre relojes se detecta comparando, para flujos con la misma clave de identidad, la dispersión de los timestamps reportados por cada exportador. Una dispersión sistemáticamente superior al segundo indica que el equipo afectado ha perdido la sincronización NTP y requiere intervención antes de que su telemetría vuelva a ser utilizable para correlación.

En el laboratorio el diagnóstico siguió tres pasos que se describen aquí en términos genéricos. Primero se midió, por exportador, la diferencia entre el fin del último flujo de cada archivo y el instante en que el colector lo cerró. Esa diferencia no distingue un reloj de pared desviado de una desviación nacida al estampar los flujos, de modo que, segundo, se comparó la hora que cada exportador escribe en la cabecera de sus paquetes con la hora de captura en el colector. La coincidencia de ambos instrumentos, que midieron el mismo desfase de forma independiente, estableció que la causa eran los relojes de pared. Tercero, se corrigió cada causa. Uno de los equipos tenía la hora fijada a mano, y su cliente NTP fallaba porque la consulta salía con la dirección privada de una interfaz sin traducción, lo que se resolvió declarando una dirección de origen explícita para NTP.

La misma revisión reveló otros tres defectos de la telemetría, ajenos al reloj. Un router exportaba en IPFIX con el campo de protocolo en cero, porque su plantilla no lo incluía, y la corrección exigió crear una plantilla nueva y reasignarla al monitor, pues el equipo no admite modificar una plantilla en uso. Parte del tráfico es observada por dos exportadores a la vez, lo que confirma en la práctica la necesidad de la deduplicación de la sección D.12.1. Y los temporizadores de expiración difieren entre fabricantes, lo que explica que la latencia sea mayor en UDP que en TCP. Todos los cambios se registran con su hora en una bitácora de captura en la base de datos, de modo que cada medición puede referirse al régimen de captura vigente.

La operación continuada permitió además tipificar cuatro modos de falla distintos del tiempo, que no se detectan del mismo modo. Un reloj fijado a mano, que ningún cliente corrige porque no hay cliente. Un cliente activo que no corrige, porque el error inicial excede lo que ajusta por deslizamiento: el cortafuegos permaneció veintisiete segundos adelantados durante días declarándose sincronizado, y solo un salto manual lo resolvió. Una fuente inestable, que engancha y suelta, dejando al equipo en deriva libre por intervalos. Y dos equipos sincronizados contra referencias distintas, cada uno correcto y discrepantes entre sí, que es lo que la correlación no tolera. De los cuatro se sigue una recomendación de diseño: disponer un servidor de tiempo en la propia red de gestión, al que apunten todos los exportadores, de modo que compartan una referencia común con retardo de milisegundos y sin depender de la salida a Internet ni de la traducción de direcciones. Y una de método: el estado que el equipo declara de sí mismo no es evidencia; la evidencia es la comparación con un instrumento externo.

D.4 Problemas Frecuentes Durante la Puesta en Servicio

Tabla D.1. Problemas típicos de la puesta en servicio y su resolución

Síntoma	Causa raíz	Resolución aplicada
Un exportador no aparece en la base de datos	ACL intermedia bloquea UDP/9995	Habilitación explícita del flujo de gestión hacia el colector
Registros con campos vacíos en v9	Plantilla no recibida por el colector tras reinicio	Reducción de template data timeout a 60 segundos
Volumen de bytes inconsistente entre exportadores	Factor de muestreo no declarado en el perfil de normalización	Registro explícito del factor por exportador (exporter_ip)
Detecciones duplicadas del mismo evento	Ventana de deduplicación insuficiente por deriva horaria	Sincronización NTP contra referencia común
Latencia de detección superior a 30 minutos	Temporizador de flujo activo con valor por defecto	Ajuste de timeout active a 60 segundos en todos los equipos
Backlog creciente en el procesador	Volumen agregado superior a la capacidad de parseo	Reducción temporal de exportadores; paralelización pendiente

Las secciones D.5 a D.9 desarrollan los cuatro desafíos que la sección 2.5 del cuerpo del informe enuncia y que la Tabla 2 sintetiza: la normalización de formatos entre fabricantes, la sincronización temporal, la deduplicación de flujos y el muestreo diferencial.

D.5 Normalización de la Heterogeneidad de Formatos entre Fabricantes

Cada fabricante implementa el estándar con matices propios. Un router Cisco IOS clásico puede exportar NetFlow v5 con campos fijos; un equipo IOS-XE moderno exporta Flexible NetFlow sobre plantillas v9; un firewall FortiGate exporta NetFlow v9 con un subconjunto reducido de campos; un equipo Huawei exporta NetStream. Aunque en sus configuraciones por defecto difieren en la disponibilidad de estadísticas direccionales —cuya exportación normalizada está especificada en el RFC 5103 [53] pero no implementada de manera uniforme por todos los fabricantes—, en la resolución de los timestamps y, de forma independiente, en su sistema de referencia (tiempo absoluto frente a milisegundos transcurridos desde el arranque del sistema), y en la semántica del campo de dirección del flujo, pues bien, la capa de normalización debe conciliar estas diferencias antes de que los registros alcancen el modelo, dado que XGBoost espera un vector de características de estructura fija y semántica homogénea.

D.6 Sincronización Temporal

La correlación entre dispositivos exige que sus relojes estén alineados. Una deriva de pocos segundos entre dos exportadores basta para que el sistema interprete como eventos distintos dos observaciones del mismo flujo, o para que asocie erróneamente flujos no relacionados. El problema se agrava en los exportadores que reportan timestamps relativos al arranque del sistema (sysUpTime) en lugar de tiempo absoluto, lo que obliga a reconstruir el tiempo real a partir de la marca de tiempo del propio paquete de exportación.

D.7 Deduplicación de Flujos

Un mismo flujo que atraviesa tres dispositivos, con exportación habilitada en su trayectoria, es exportado tres veces, una por cada dispositivo en la trayectoria. Sin deduplicación, el sistema contabilizaría el triple del volumen real, distorsionaría las métricas agregadas por dirección IP y —lo que resulta más grave para la detección— multiplicaría artificialmente las alertas generadas por un único evento. La deduplicación requiere definir una clave de identidad de flujo estable entre exportadores y una ventana de tolerancia temporal, puesto que los distintos dispositivos no exportan el mismo flujo en el mismo instante.

Conviene precisar que la deduplicación no persigue descartar la redundancia, sino reinterpretarla. El objetivo es colapsar las observaciones múltiples en un registro único que preserve, como atributo derivado, el número y la identidad de los exportadores que observaron el flujo. De este modo la multiplicidad deja de inflar los conteos y pasa a constituir una característica informativa sobre la trayectoria del tráfico.

D.8 Muestreo Diferencial

Los equipos de alta capacidad pueden exportar la totalidad de los flujos observados o únicamente una muestra estadística, con tasas configurables (1:10, 1:100, 1:1000). La decisión responde a restricciones operativas: el costo de procesamiento en el plano de control del exportador, el ancho de banda consumido por la propia telemetría y la capacidad del colector para absorber ráfagas de registros en enlaces de alta velocidad. En consecuencia, es habitual que un mismo despliegue combine exportadores sin muestreo en segmentos de acceso con exportadores fuertemente muestreados en el core o el borde.

Esta heterogeneidad afecta al procesamiento en tres niveles de gravedad creciente. Los contadores dejan de ser comparables entre fuentes y deben escalarse por el factor de muestreo; el escalado, sin embargo, no restituye los flujos que nunca fueron muestreados, y esa pérdida no es uniforme, pues degrada selectivamente la señal de baja intensidad —justamente la que caracteriza las fases tempranas de un compromiso—; por último, si cada exportador mantiene una tasa estable en el tiempo, el factor de escalado se convierte en un identificador encubierto del dispositivo de origen y habilita una forma de fuga de información difícil de detectar. La sección D.9 desarrolla los tres niveles y las verificaciones que permiten acotarlos.

D.9 Efecto del Muestreo Diferencial sobre la Detección

Esta heterogeneidad afecta al procesamiento en tres niveles de gravedad creciente. En el primero, los contadores de bytes y paquetes dejan de ser directamente comparables entre fuentes y deben escalarse por el factor de muestreo antes de cualquier agregación; omitir este ajuste introduce un sesgo sistemático que el modelo interpretaría como variación de comportamiento.

En el segundo, y de forma más problemática, el escalado recupera una estimación razonable del volumen agregado, pero no restituye los flujos que nunca fueron muestreados. La pérdida no es uniforme: con una tasa de 1:1000, una transferencia masiva sostenida sobrevive a la selección con alta probabilidad, mientras que un barrido de reconocimiento compuesto por centenares de flujos de escasa duración y pocos paquetes puede desaparecer casi por completo del registro. El muestreo degrada, por tanto, de manera selectiva la señal de baja intensidad —precisamente aquella que caracteriza las fases tempranas de un compromiso y que motivaba la captura multi-dispositivo.

En el tercero, si cada exportador mantiene una tasa de muestreo estable en el tiempo, el factor de escalado se convierte en un identificador encubierto del dispositivo de origen. El modelo puede entonces aprender a discriminar en función de la procedencia de la telemetría antes que del comportamiento del tráfico, una forma de fuga de información difícil de detectar porque ninguna característica del vector de entrada declara explícitamente el exportador. Mitigarlo exige, como mínimo, verificar que el desempeño del clasificador se mantenga estable al evaluarlo por separado sobre subconjuntos provenientes de exportadores con tasas distintas.

D.10 Puntos de Captura Considerados en el Diseño

La Tabla D.2 recoge las posiciones topológicas contempladas para la instrumentación de exportadores, con el tipo de tráfico que cada una observa y el fabricante del equipo correspondiente en el laboratorio.

Tabla D.2. Puntos de captura considerados en el diseño de instrumentación

Dispositivo tipo	Rol topológico	Protocolo	Muestreo	Tráfico observado
Router de borde (Cisco Catalyst 8300)	Frontera con el proveedor de Internet	NetFlow v9	1:1	Norte-sur completo
Switch de core (Cisco Catalyst 9500)	Interconexión entre segmentos internos	Flexible NetFlow v9	Configurable	Este-oeste de alto volumen
Firewall perimetral (FortiGate)	Segmentación DMZ y publicación de servicios	NetFlow v9	1:1	Tráfico hacia y desde la DMZ
Router de sucursal (Huawei NetEngine AR)	Enlace WAN con sede remota	NetStream (v9)	1:1	Tráfico intersede

D.11 Especificación de los Mecanismos de Correlación Multi-Dispositivo

Esta sección desarrolla los cuatro mecanismos que la sección 3.5 del cuerpo del informe enuncia: la identificación del exportador de origen, la normalización por perfil de fabricante, la deduplicación de flujos y la validación cruzada entre puntos de observación. La identificación del exportador está construida, la normalización y la deduplicación lo están en parte y la validación cruzada permanece especificada, conforme al estado declarado en la Tabla 4.

El diseño establece que cada registro de flujo se etiquete con el identificador del dispositivo que lo exportó. Ese identificador se deriva de la dirección IP de origen del paquete de exportación UDP, información que nfcapd conserva en la cabecera de cada archivo de captura y que, por tanto, no exige configuración adicional en los equipos de red. En el prototipo la trazabilidad se materializó primero de forma indirecta, en la columna file_source, que registra el archivo de captura de procedencia, y hoy en columnas propias con la dirección y el identificador del exportador, pobladas durante el procesamiento.

La trazabilidad del origen cumple tres funciones y es prerrequisito de todo lo que sigue: habilita el filtrado por dispositivo en los tableros, permite aplicar el factor de muestreo correcto durante la normalización y hace posible la validación cruzada, que por definición exige distinguir observaciones procedentes de puntos independientes.

D.12 Normalización a Esquema Común

La capa de normalización contempla un perfil por fabricante que resuelva las diferencias de formato analizadas en la sección 2.5. El perfil se seleccionaría automáticamente a partir del identificador del exportador y de la versión de protocolo detectada por nfdump, y ejecutaría cuatro operaciones sobre cada registro:

- Mapeo de campos hacia un esquema unificado, el cual resuelve las divergencias terminológicas entre fabricantes. Por ejemplo, las variables initiatorOctets de IPFIX y dOctets de NetFlow v5 se homologan bajo el atributo estandarizado total_length_of_fwd_packets.
- Escalado por factor de muestreo. Los contadores de bytes y de paquetes procedentes de exportadores con muestreo 1:N se multiplican por N para restituir la magnitud estimada real antes de cualquier agregación entre fuentes. Omitir este paso haría que un exportador con muestreo 1:1000 apareciera como marginal frente a otro que exporta la totalidad de los flujos.
- Conversión temporal a tiempo universal (UTC). Los exportadores que reportan marcas de tiempo relativas al arranque del equipo obligan a reconstruir el instante absoluto de cada flujo. La operación resta al instante de exportación consignado en la cabecera del paquete el sysUpTime del equipo en ese mismo momento, con lo que se obtiene el instante de arranque en tiempo absoluto, y sobre esa base se sitúan los marcadores de inicio y término del flujo. Esta transformación a un marco temporal común es indispensable para comparar y correlacionar los flujos provenientes de dispositivos distintos.
- Imputación de campos ausentes. Cuando un exportador no provee estadísticas direccionales —caso característico de NetFlow v5— el diseño prevé estimar los campos correspondientes a partir de los contadores agregados disponibles y marcar el registro con un indicador de calidad reducida que permite ponderar su peso en la inferencia y en el análisis posterior.

De estas cuatro operaciones, el prototipo implementa el mapeo de campos al esquema de la tabla netflow_data y la corrección del contador de retorno que se describe a continuación. El escalado por muestreo, la reconstrucción temporal propia y la imputación con indicador de calidad quedan especificados como diseño.

La operación sobre tráfico real obligó a una corrección que el diseño no preveía: descartar contadores que el exportador no observa sino que fabrica. NetFlow v9 exporta cada sentido de una sesión por separado; el cortafuegos perimetral rellena el contador de paquetes de vuelta copiando el de ida, y los otros dos exportadores lo dejan en cero. El procesador ya no toma ese campo como observación y reconstruye la respuesta emparejando cada flujo con el de sentido opuesto de la misma sesión, según la regla que la sección E.8 describe. Es la operación inversa de la imputación, pues no completa un campo ausente sino que descarta uno presente que no es dato, y debe formar parte del perfil de cada fabricante.

La reconstrucción temporal tiene, por su parte, un modo de falla predecible. El sysUpTime es un contador de 32 bits en milisegundos que se desborda cada 2^32 ms, unos 49,71 días, y si el cálculo no contempla la vuelta del contador los flujos quedan fechados exactamente un período en el futuro. Así se observó en el prototipo, en un episodio de 1.273 flujos fechados unos cincuenta días después de su inserción por un equipo cuyo tiempo de actividad superaba ese período. La comprobación es inmediata, un desplazamiento igual al período del contador, y conviene incluirla en la validación de cada perfil. La Tabla E.3 del Anexo E reúne los defectos de normalización observados por exportador.

D.12.1 Deduplicación de Flujos

La deduplicación propuesta opera sobre una clave de identidad de flujo formada por la 5-tupla canónica —dirección IP de origen, dirección IP de destino, puerto de origen, puerto de destino y protocolo— y el instante de inicio redondeado a una ventana de tolerancia configurable. El valor inicial de diseño para esa ventana es de dos segundos, escogido como orden de magnitud plausible de la dispersión entre exportaciones del mismo flujo.

Su calibración definitiva exige medir dicha dispersión sobre la maqueta de laboratorio, medición que forma parte del plan de evaluación operacional de la sección E.7.

Conforme al diseño, ante registros duplicados el sistema no descartaría sin más las copias redundantes. Conservaría un único registro canónico —el procedente del exportador con menor factor de muestreo, por ser el de mayor fidelidad estadística— e incrementaría un contador observation_count que refleje cuántos dispositivos independientes observaron ese mismo flujo. Ese contador no es un subproducto contable, sino el insumo directo del mecanismo de validación cruzada que se describe a continuación. Ni el contador ni la validación cruzada están construidos; la deduplicación sí lo está dentro de cada exportador, en la forma que se describe a continuación.

La dispersión se midió sobre dos horas de tráfico, emparejando la misma 5-tupla registrada por exportadores distintos, y resultó bimodal. El 78,9 % de los pares difiere en dos segundos o menos, con desfase mediano entre −1 y +1 s y dispersión inferior a un segundo; son flujos cortos, de duración mediana nula, y la tolerancia de diseño los capta enteros. Solo el 0,7 % cae entre dos y diez segundos, de modo que los dos segundos coinciden con un corte natural de la distribución. El 20,4 % restante difiere entre diez y sesenta segundos, pero no por desfase de reloj: su dispersión, de unos 41 s, corresponde a un reparto casi uniforme, su signo varía entre pares de exportadores y lo forman flujos largos, de duración mediana de 80 s, que cada exportador divide en tramos según su temporizador activo, con límites que no coinciden. Una medición posterior sobre una hora encontró 1.819 pares de copias entre exportadores, del orden del 1,7 % de los flujos, y ninguno con ambas copias puntuadas por el clasificador, de modo que la duplicación entre exportadores infla los volúmenes y los agregados del detector, pero no las alertas ni las métricas del modelo.

De ello se sigue una corrección del diseño. La clave por proximidad del instante de inicio vale para flujos cortos, pero los que superan el temporizador activo deben emparejarse por solapamiento de intervalos. El procesador adoptó ese criterio dentro de cada exportador, donde elimina las copias repetidas de un mismo flujo, pero no lo aplica entre exportadores, y la consecuencia es medible: un único barrido registrado por varios exportadores conservó del orden de 72.000 observaciones canónicas, una por flujo y exportador.

D.12.2 Validación Cruzada entre Puntos de Observación

La validación cruzada propuesta ajusta la severidad de una detección en función del grado de acuerdo entre los dispositivos que observaron el flujo. Conviene deslindar las dos magnitudes que intervienen, pues operan en planos distintos. El threat_score es una propiedad del flujo: la probabilidad, en el intervalo [0, 1], que el modelo le asigna de corresponder a tráfico malicioso, obtenida exclusivamente a partir de las características del propio registro. La severidad, en cambio, es una propiedad de la alerta: la categoría operativa —ALTA, MEDIA o INFORMATIVA— con que el hallazgo se presenta al analista, y que expresa no cuán malicioso parece el flujo, sino cuánta urgencia de revisión amerita. La distinción importa porque un mismo threat_score puede justificar severidades diferentes según el contexto de observación: una puntuación de 0,8 confirmada por dos exportadores independientes constituye evidencia más sólida que la misma puntuación registrada por un único dispositivo con muestreo 1:1000, donde el hallazgo podría ser un artefacto de la instrumentación antes que un comportamiento real.

La regla es, en consecuencia, deliberadamente conservadora: no modifica el threat_score emitido por el modelo, que se conserva íntegro para efectos de auditoría y de reentrenamiento, sino que actúa exclusivamente sobre el nivel de severidad. Esta separación tiene tres consecuencias prácticas. Primera, preserva la trazabilidad del clasificador, ya que la puntuación almacenada sigue siendo la que el modelo efectivamente produjo y no una cifra corregida a posteriori. Segunda, permite ajustar la política de severidad —las reglas de la Tabla D.3— sin reentrenar ni revalidar el modelo, por tratarse de componentes independientes. Tercera, evita introducir en los datos de reentrenamiento una realimentación que confundiría la evidencia del tráfico con el juicio operacional sobre ella.

Tabla D.3. Reglas de ajuste de severidad por validación cruzada entre exportadores

Condición observada	Severidad asignada	Fundamento
threat_score > 0,7 confirmado por ≥ 2 exportadores	Alta	Dos instrumentos independientes coinciden; la anomalía es del tráfico y no de la medición
threat_score > 0,7 en un único exportador con muestreo 1:1	Media	Sin observación redundante disponible, pero fidelidad de captura completa
threat_score > 0,7 en un único exportador con muestreo 1:N	Informativa	Posible artefacto de muestreo; requiere confirmación
threat_score entre 0,5 y 0,7 confirmado por ≥ 2 exportadores	Media	La concordancia compensa la puntuación intermedia
Discrepancia entre exportadores sobre el mismo flujo	Informativa	Señal de posible anomalía de instrumentación; se registra para diagnóstico

Conviene subrayar una limitación estructural de este mecanismo. La validación cruzada solo es aplicable a los flujos que efectivamente atraviesan más de un exportador instrumentado. El tráfico confinado a un único segmento —por ejemplo, el intercambiado entre dos servidores conectados a un mismo switch de acceso no instrumentado— no admite confirmación redundante y permanece necesariamente en severidad media. Ampliar la cobertura de instrumentación reduce esa zona ciega, pero a costa de incrementar el volumen de telemetría que el colector debe absorber; el punto de equilibrio entre ambos efectos es específico de cada red y debe determinarse empíricamente.

La zona ciega se midió. Sobre seis horas de tráfico, de 575.414 flujos lógicos solo 9.486, alrededor del 1,6 %, los registra más de un exportador. La cifra es una cota inferior, porque el emparejamiento exacto no reconoce los tramos desalineados descritos en la sección D.12.1 ni las traducciones de dirección entre puntos de observación (sección D.13). Hay además un conflicto con el procesador actual. Las reglas de la Tabla D.3 presuponen que varios exportadores puntúan el mismo flujo, y hoy solo se puntúa la copia del cortafuegos, porque en los registros de los demás exportadores el emparejamiento aún no puede establecer si hubo respuesta y la regla de la sección E.8 los excluye de la inferencia. Mientras eso no cambie, la condición de confirmación por dos o más exportadores no puede cumplirse, y la capa de aprendizaje automático observa en la práctica solo el perímetro.

D.13 Efecto del Punto de Observación sobre los Atributos de Dirección

La posición del exportador determina además el significado de los atributos derivados de las direcciones. En una red con traducción de direcciones, un mismo flujo presenta direcciones privadas reales si se observa en la interfaz interna, y aparece ya traducido —con decenas de estaciones colapsadas en una sola dirección pública— si se observa en la externa. Los indicadores de pertenencia a rango privado significan por tanto cosas distintas según el punto de observación. De ello se siguen dos decisiones: instrumentar preferentemente la interfaz interna de los equipos que traducen, y tratar la dirección del flujo respecto del perímetro como un atributo que la capa de normalización declare explícitamente, en lugar de inferirlo del rango observado. La sección 6.3 recoge la consecuencia de esta ambigüedad sobre el conjunto de características.

Las mediciones sobre tráfico real mostraron que la visibilidad de un segmento depende de tres condiciones independientes. La primera es que el proveedor entregue el tráfico al equipo, condición ajena al laboratorio; uno de los segmentos públicos declarados no recibe tráfico alguno, lo que se verificó capturando paquetes en la interfaz. La segunda es que la política lo acepte, pues el cortafuegos exporta sesiones y el tráfico denegado no genera sesión ni registro. La tercera es que la interfaz tenga activo el muestreo, que se configura interfaz por interfaz. Al activarlo en cuatro segmentos que no lo tenían, cada uno pasó a registrar un volumen de sondeo comparable al del segmento ya observado, del orden de un millar de orígenes distintos por hora, sin duplicación de registros. La captura previa observaba, por tanto, una fracción del sondeo real, lo que no invalida las mediciones, que se refieren a lo observado, pero delimita su alcance. Los segmentos se declararon como espacio propio después de los veredictos de la adjudicación, y ninguna ficha adjudicada los involucraba. Se adoptó como regla que toda interfaz con espacio propio público exporte, y al completarla se verificó que el cortafuegos

exporta cada sesión una sola vez, cualquiera sea el número de interfaces con exportación activa. El tráfico entre dos interfaces no exportadas —por ejemplo, entre una red privada y un servidor interno— sigue fuera de la observación y se declara como límite de alcance.

Anexo E: Implementación y Verificación del Prototipo de Laboratorio

Este anexo reúne la documentación técnica del prototipo construido para validar la viabilidad del diseño expuesto en el Capítulo 3. Se recoge aquí, y no en el cuerpo del informe, porque constituye material de implementación y de operación antes que desarrollo conceptual. Su valor es doble. Permite reproducir el despliegue sobre equipamiento equivalente, y documenta las verificaciones que sustentan la afirmación de viabilidad técnica del Capítulo 5.

E.1 Despliegue de la Plataforma

Dado que la plataforma comprende varios contenedores, su puesta en marcha se coordina con Docker Compose, herramienta que permite levantar simultáneamente todos los servicios a partir de un único archivo de configuración. El despliegue se ejecuta mediante la orden docker-compose up -d, que crea e inicia en segundo plano los servicios declarados y respeta el orden impuesto por las dependencias entre ellos. Así, PostgreSQL debe alcanzar el estado saludable antes de que arranquen el procesador de flujos y la API. Esa condición se comprueba con pg_isready, utilidad de línea de comandos incluida en la distribución de PostgreSQL que consulta el estado del servidor y devuelve un código indicando si acepta conexiones, sin necesidad de abrir una sesión completa.

E.2 Configuración de los Exportadores del Laboratorio

La puesta en servicio del prototipo requirió configurar la exportación de flujos en los dispositivos del laboratorio. Dado que ninguno de ellos admite configuración automatizada desde el colector, esta etapa se ejecutó manualmente sobre cada equipo. A continuación, se documentan las configuraciones aplicadas, que constituyen el procedimiento reproducible para replicar el despliegue sobre equipos equivalentes.

En todos los casos la dirección de destino corresponde al host que ejecuta el stack Docker y el puerto de destino es el 9995/UDP expuesto por nfcapd. La sincronización horaria mediante NTP (Network Time Protocol, el protocolo estándar que alinea los relojes de los dispositivos contra una referencia común) es condición indispensable para la correlación temporal descrita en el Capítulo 3; la operación mostró que no basta con configurarla y que debe verificarse, como se expone en la sección E.8.

Las configuraciones se expresan en la sintaxis vigente de cada fabricante. En el caso de Cisco, ello significa Flexible NetFlow —bloques flow record, flow exporter y flow monitor aplicados a la interfaz con ip flow monitor—, que es el único modelo disponible en IOS-XE y, por tanto, en las plataformas actuales de borde y de core, como las familias Catalyst 8000 y Catalyst 9000. La sintaxis tradicional de IOS clásico, basada en los comandos ip flow-export e ip flow ingress, no existe en esas plataformas y subsiste únicamente en generaciones de equipamiento ya fuera de soporte del fabricante.

E.2.1 Cisco IOS-XE — Router de Borde

```
flow record FR-BORDE
 match ipv4 protocol
 match ipv4 source address
 match ipv4 destination address
 match transport source-port
 match transport destination-port
```

```
 collect counter bytes long
 collect counter packets long
 collect transport tcp flags
 collect timestamp absolute first
 collect timestamp absolute last
!
flow exporter FE-COLECTOR
 destination 10.20.0.15
 source Loopback0
 transport udp 9995
 export-protocol netflow-v9
 template data timeout 60
!
flow monitor FM-BORDE
 exporter FE-COLECTOR
 record FR-BORDE
 cache timeout active 60
 cache timeout inactive 15
!
interface GigabitEthernet0/0/0
 ip flow monitor FM-BORDE input
 ip flow monitor FM-BORDE output
```

El parámetro cache timeout active 60 fuerza la exportación de los flujos de larga duración cada sesenta segundos, en lugar del valor por defecto de treinta minutos. Este ajuste resultó determinante para la latencia de detección: sin él, un flujo de exfiltración sostenida no se reportaría hasta media hora después de haberse iniciado. Conviene notar que la unidad del temporizador difiere entre generaciones —en la sintaxis tradicional de IOS clásico el valor se expresaba en minutos y en Flexible NetFlow se expresa en segundos—, diferencia que ha originado errores de configuración al migrar plataformas.

E.2.2 Cisco IOS-XE — Switch de Core con Flexible NetFlow

```
flow record FR-SEGURIDAD
match ipv4 source address
match ipv4 destination address
match transport source-port
match transport destination-port
match ipv4 protocol
collect counter bytes long
collect counter packets long
collect transport tcp flags
collect timestamp absolute first
collect timestamp absolute last
!
flow exporter FE-COLECTOR
destination 10.20.0.15
transport udp 9995
export-protocol netflow-v9
template data timeout 60
!
flow monitor FM-SEGURIDAD
exporter FE-COLECTOR
```

```
record FR-SEGURIDAD
cache timeout active 60
!
interface TenGigabitEthernet1/0/1
 ip flow monitor FM-SEGURIDAD input
 ip flow monitor FM-SEGURIDAD output
```

La cláusula collect transport tcp flags es la que habilita la detección de los patrones de escaneo descritos en el marco teórico —Xmas, Null y FIN scan— que dependen íntegramente de la disponibilidad de los flags TCP en el registro exportado. El parámetro template data timeout 60 garantiza que el colector reciba las plantillas con frecuencia suficiente para interpretar los registros aun después de un reinicio del servicio nfcapd.

E.2.3 Fortinet FortiGate — Firewall Perimetral

```
config system netflow
    config collectors
        edit 1
            set collector-ip 10.20.0.15
            set collector-port 9995
            set source-ip 10.20.1.1
        next
    end
    set active-flow-timeout 60
    set inactive-flow-timeout 15
end
config system interface
    edit "port1"
        set netflow-sampler both
    next
end
```

La configuración anterior corresponde a FortiOS 7.4.2 y posteriores, versiones en las que el colector se define dentro del bloque config collectors, que admite varios destinos simultáneos; en versiones anteriores los mismos parámetros se declaraban directamente bajo config system netflow. FortiOS exporta NetFlow v9 con un subconjunto de campos más reducido que los equipos de enrutamiento: no provee estadísticas separadas de tamaño medio de paquete por dirección. El perfil de normalización previsto para este exportador estimaría esos campos a partir de los contadores agregados y marcaría los registros con un indicador de calidad reducida, mecanismo especificado en la sección D.12 y aún no construido.

El muestreo se activa por interfaz, y una interfaz sin netflow-sampler no exporta las sesiones que la atraviesan. Cuando el tráfico entra por una interfaz sin muestreo y sale por otra que lo tiene, la sesión se exporta una sola vez. Conviene además declarar la dirección de origen de los servicios propios del equipo, como NTP, porque el tráfico generado por el propio cortafuegos no se traduce.

Tres propiedades de la exportación de FortiOS, observadas en operación, condicionan la detección. No exporta las banderas TCP, de modo que la cláusula collect transport tcp flags de los equipos Cisco no tiene contraparte y los sondeos de la Tabla A.1 no se distinguen en sus registros. Rellena el contador de paquetes de vuelta copiando el de ida, defecto que el procesador corrige (sección D.12). Y el instante en que exporta una sesión lo fijan sus temporizadores de sesión, no solo los de exportación: en la configuración del laboratorio una sesión TCP se exporta entre uno y diez segundos después de su cierre, una sesión UDP solo al expirar su temporizador de inactividad, de 180 s, y una sesión de tráfico bloqueado a los 30 s. El temporizador UDP explica la diferencia de latencia entre protocolos de la sección E.8. Conviene, por último, verificar el temporizador activo efectivamente configurado: en el laboratorio estuvo en 1.800 s hasta que se redujo a 60 s, y con el valor anterior la duración máxima registrada crecía a razón de un día por día, artefacto que desapareció con el cambio.

E.2.4 Huawei NetEngine AR — Router de Sucursal (NetStream)

```
ip netstream export version 9
ip netstream export host 10.20.0.15 9995
ip netstream export source 10.30.1.1
ip netstream timeout active 1
ip netstream timeout inactive 15
#
interface GigabitEthernet0/0/1
ip netstream inbound
ip netstream outbound
```

E.3 Verificación Funcional del Prototipo

Tabla E.1. Métricas operacionales objetivo del sistema

Métrica	Valor objetivo	Fundamento del objetivo
Throughput de procesamiento	> 400 flujos/seg	Estimación de carga pico de una red mediana con margen de seguridad
Latencia media de detección (MTTD)	< 6 minutos	Ventana de rotación de 5 min + sondeo de 30 s + inferencia
Disponibilidad del sistema	> 99%	Reinicio automático por healthchecks; sin alta disponibilidad multi-nodo
Tiempo de respuesta API /predict (p95)	< 200 ms	Inferencia XGBoost del orden de microsegundos; margen para red y validación
Tasa de falsos positivos	Reducción ≥ 30% con captura multi-dispositivo	Efecto esperado de la validación cruzada, por verificar empíricamente

La validación funcional ejecutada sobre el laboratorio confirmó la operación de extremo a extremo del pipeline: los exportadores entregan flujos al colector, los archivos de captura se generan y rotan conforme a la ventana configurada, el procesador los interpreta e inserta los registros en la base de datos, la API responde a solicitudes de clasificación con el modelo cargado, y los dashboards reflejan los datos con latencia de actualización inferior a cinco segundos. Esta verificación se apoyó en los scripts de diagnóstico desarrollados junto al prototipo, que comprueban la salud de cada contenedor, la presencia de archivos de captura recientes, el crecimiento de la tabla de flujos y la respuesta del endpoint de predicción. De las métricas de la tabla, el MTTD ya se midió sobre tráfico real (sección E.8); el caudal sostenido y la disponibilidad de largo plazo se obtendrán al ejecutar el plan de evaluación de la sección E.7, y hasta entonces deben leerse como objetivos de diseño verificables, no como resultados medidos.

E.4 Detección Complementaria por Anomalías

Además del clasificador supervisado, el prototipo incorpora un detector complementario no supervisado, implementado como módulo independiente que opera directamente sobre la base de datos. Su función es compensar la debilidad estructural del clasificador por flujo ante los ataques cuya señal reside en el agregado, como el sondeo y el tráfico de baliza. No debe confundirse con la línea base de reglas estáticas R1–R4, que solo sirve de término de comparación, y no interviene en el entrenamiento del modelo.

El detector agrega los flujos de las últimas 24 horas por dirección IP de origen y construye para cada una un perfil de comportamiento: número de destinos y puertos únicos contactados, estadísticas de bytes y paquetes, diversidad de protocolos, dispersión horaria de la actividad y ratios derivados como el indicador de escaneo de puertos. Sobre esos perfiles aplica Isolation Forest [21] —el algoritmo de aislamiento presentado en la sección

2.3— para asignar a cada IP una puntuación de anormalidad, que se traduce en niveles de amenaza (alto, medio, bajo, normal). De forma complementaria, dos reglas heurísticas explícitas identifican patrones inequívocos: el escaneo de puertos (más de diez puertos únicos contactados en menos de una hora) y la denegación de servicio (más de cinco orígenes concurrentes contra un mismo destino con volumen superior al umbral en ventana corta). Las detecciones se persisten en la tabla detected_threats con su tipo, nivel y detalle en formato JSON, quedando disponibles para los dashboards y para el análisis posterior.

La versión vigente incorpora tres correcciones que la medición sobre tráfico real hizo necesarias. Antes de agregar, descarta los registros que corresponden al sentido de respuesta de una sesión ya observada —el cortafuegos exporta cada sentido por separado—, de modo que el perfil de un origen refleja solo las conexiones que inicia y el destino de una alerta de sondeo es el objetivo real. La proporción de contaminación del bosque se fijó en el 1 %. Conviene advertir que ese parámetro no es un umbral de decisión sino un presupuesto de alertas, pues el algoritmo señala esa proporción de los orígenes observados cualquiera sea su comportamiento, de modo que fija a la vez la carga y el techo de cobertura del detector. Y las alertas de un origen ya señalado en la ventana vigente se omiten, porque las ejecuciones horarias sobre una ventana de veinticuatro horas repetían la misma alerta hasta catorce veces.

Esa supresión tenía un defecto que solo un ataque real puso de manifiesto. Identificaba la alerta previa por el origen y la restringía a las alertas activas, de modo que un origen ya señalado por un sondeo menor quedaba silenciado durante veinticuatro horas aunque después barriera decenas de miles de puertos, y una alerta cerrada por el analista dejaba de suprimir y se repetía. La versión vigente compara magnitudes: vuelve a emitir cuando la magnitud de la detección —puertos contactados en el sondeo, puertos distintos en la anomalía, orígenes en la denegación de servicio— triplica la de la alerta previa, cualquiera sea su estado. Con esa corrección el barrido real descrito en la sección E.8, invisible para el clasificador, quedó detectado. Se observó además que el bosque de aislamiento emite un número constante de anomalías por ejecución, cien en la configuración vigente, porque agota su presupuesto en cada una; el recuento de anomalías no informa, por tanto, sobre la actividad de la red, y solo la identidad y el orden de los orígenes señalados lo hacen.

Sobre el clasificador opera además una regla de campaña, que agrupa sus alertas por flujo por puerto de destino y protocolo en una ventana de veinticuatro horas. Un grupo con al menos cinco orígenes distintos constituye una campaña: se emite una única alerta con los agregados —orígenes, destinos y alertas contenidas— y las alertas por flujo se marcan como explicadas por ella, sin borrarse, de modo que siguen disponibles para la auditoría y para el censo de precisión. Una campaña vigente no se vuelve a emitir salvo que el número de orígenes triplique el de la anterior, el mismo criterio de escalada del detector por agregados. La clave es el puerto y no el puerto y la hora porque las campañas observadas son sostenidas: partidas por hora la compresión era de 2,8 veces, y por puerto es de unas 40, de 675 a 17 alertas en la ventana medida. La contrapartida es que la agregación oculta la recurrencia de cada origen, que sigue disponible en las alertas marcadas. La regla es idempotente; queda por observar su rama de escalada con tráfico nuevo y por fijar su cadencia de ejecución.

La coexistencia de ambos enfoques —clasificación supervisada por flujo y detección de anomalías por perfil de comportamiento— ilustra en el propio prototipo la complementariedad teórica discutida en el Capítulo 2: el primero aporta precisión sobre patrones de ataque conocidos; el segundo, sensibilidad frente a comportamientos nuevos que se aparten de la normalidad aprendida de la red observada.

La supresión se completó con la persistencia de condiciones. Cuando una alerta se omite por repetición, la alerta vigente de esa condición registra la última vez que se la vio y el número de veces, y la ventana de veinticuatro horas se cuenta desde esa última vista, con un tope de treinta días; la escalada por magnitud no cambia. Una condición continua conserva así una única alerta con su historia, en lugar de reaparecer cada día. Una estimación previa, que es una cota superior, indicaba que el 37,8 % de las anomalías agregadas y el 14,2 % de los sondeos de una semana eran continuaciones. Con presupuesto fijo el total diario de anomalías no baja, porque el cupo liberado pasa a orígenes nuevos, y ese es el efecto buscado: el mismo esfuerzo de revisión cubre más condiciones distintas. Verificada la persistencia, la proporción de alertas sobre orígenes propios cayó de cerca del 2,8 % al 0,5 %: eran las mismas sesenta y cuatro condiciones estables re-emitidas a diario, explicadas por caras de traducción, clientes intensivos en DNS, extremos de túneles y equipos de laboratorio (sección E.10).

La confianza que acompaña a cada alerta no tiene la misma semántica en todos los tipos. En la alerta por flujo es la probabilidad del clasificador, no calibrada para esta red; en la anomalía agregada es una

transformación con tope de la puntuación del bosque, que es relativa a cada ejecución; en el sondeo y la denegación de servicio es una constante. No es comparable entre tipos ni entre ejecuciones, y por eso cada anomalía conserva además su percentil dentro de la ejecución y el número de orígenes evaluados, información que permite cruzarla con el censo sin alterar la columna original.

E.5 Instrumentos de Visualización

Los dashboards Grafana implementados proveen cuatro vistas: (1) Resumen ejecutivo con métricas de amenazas detectadas en las últimas 24 horas y distribución por severidad de threat_score. (2) Análisis de tráfico por protocolo y dirección IP con drill-down a nivel de flujo individual. (3) Línea de tiempo de eventos de seguridad con filtrado por tipo de ataque, rango temporal y umbral de score ML. (4) Panel de monitoreo de infraestructura con métricas de CPU, memoria, I/O y rendimiento de la base de datos desde Prometheus. La integración directa con PostgreSQL mediante SQL nativo reduce la latencia de actualización de los dashboards a menos de 5 segundos.

E.6 Parámetros de Configuración de los Componentes

Esta sección recoge los parámetros operativos de cada componente, que la sección 3.3 del cuerpo del informe resume sin detallar.

El colector escucha en el puerto 9995/UDP y admite exportaciones NetFlow v5, v9 e IPFIX, lo que permite recibir telemetría de dispositivos de distinta generación y fabricante sin colectores intermedios. Opera con ventanas de rotación de cinco minutos (parámetro -t 300), organiza el almacenamiento por marca temporal y utiliza la zona horaria America/Santiago. Cada rotación cierra un archivo binario y abre el siguiente, de modo que la periodicidad de la rotación fija el primer sumando de la latencia de detección del sistema. El contenedor se ejecuta con un usuario no privilegiado (UID 1000) y monta el volumen nfcapd_data, que comparte con el procesador. Es el único componente que recibe tráfico procedente de la red monitoreada.

E.6.1 Procesador de Flujos (netflow-processor)

El procesador, implementado en Python, sondea el volumen nfcapd_data cada treinta segundos, detecta los archivos cerrados desde la iteración anterior y los decodifica con nfdump. Sobre cada registro obtenido ejecuta dos operaciones de normalización. Primero, proyecta los campos del exportador sobre el esquema común de la tabla netflow_data. Segundo, compone el identificador de conexión, una cadena que concatena con guiones la 5-tupla del flujo en el orden dirección IP de origen, dirección IP de destino, puerto de origen, puerto de destino y protocolo —por ejemplo, 10.20.3.14-192.168.1.7-49512-443-6—, y que constituye la entrada del transformador descrito en la sección 4.4.

Tres parámetros gobiernan su comportamiento: BATCH_SIZE, que fija el número de registros insertados por transacción; PROCESSING_INTERVAL, que define el período de sondeo; y MIN_FILE_SIZE_KB, que descarta los archivos vacíos producidos por ventanas sin tráfico. Sus valores por defecto —1000, 30 y 1, respectivamente— permiten ajustar el balance entre latencia de detección y sobrecarga de procesamiento según las características del entorno de despliegue.

E.6.2 Servicio de Inferencia (FastAPI + XGBoost)

El servicio se implementa mediante el framework FastAPI [11], diseñado para el desarrollo rápido y eficiente de API web en Python. Este componente se expone a través del puerto 8000 y publica dos puntos de acceso (endpoints): GET /flows, que recupera flujos almacenados aplicando paginación mediante los parámetros limit y skip; y POST /predict, que clasifica flujos de datos en tiempo real (en línea). La validación de entrada se apoya en modelos Pydantic (herramienta que FastAPI usa internamente para revisar y validar que los datos que entran y salen sean correctos), que garantizan el tipado de los campos del flujo y admiten alias para preservar

la compatibilidad con la nomenclatura del dataset CICIDS2017, en la que los nombres contienen espacios (por ejemplo, «total fwd packets»); de los campos validados, trece —los doce campos numéricos y el identificador de conexión— alimentan directamente el pipeline del modelo y el resto se conserva con fines de trazabilidad.

El modelo, entrenado previamente en la plataforma Google Colab, se almacena en el archivo modelo_xgboost_v2_desplegable.pkl, el artefacto de veinte características descrito en la sección 4.5, el cual contiene el pipeline completo de transformación y clasificación, junto al módulo custom_transformers_v2.py del que dependen los transformadores personalizados. Este archivo se carga una única vez durante el arranque del servicio FastAPI, optimizando el rendimiento al evitar el costo computacional de la deserialización en cada petición. El punto de acceso /predict admite tanto objetos individuales como listas de flujos y retorna, para cada uno, un threat_score. Este valor es la probabilidad que el modelo asigna a que el flujo corresponda a tráfico malicioso y se expresa, por tanto, en el intervalo [0, 1]. Una puntuación de 0,05 indica que el clasificador considera el flujo prácticamente indistinguible del tráfico benigno, mientras que una de 0,95 expresa confianza alta en lo contrario. Se trata de una medida continua de confianza y no de un veredicto binario, y la distinción es deliberada porque conservar la puntuación completa permite ordenar los flujos por prioridad de revisión, examinar la franja intermedia —donde se concentran los casos ambiguos— y reinterpretar a posteriori una decisión sin necesidad de reprocesar el tráfico. Por esa razón el threat_score se persiste íntegro en la tabla netflow_data.

La conversión de esa puntuación en una alerta constituye una decisión distinta, que el sistema mantiene separada de la inferencia. El parámetro THREAT_THRESHOLD define el punto de operación del clasificador, esto es, el valor a partir del cual una puntuación se promueve a alerta de severidad alta y se registra como amenaza en la tabla detected_threats. Se fijó en 0,7, por sobre el punto neutro de 0,5, con lo que se privilegia la precisión sobre la sensibilidad en la etapa de alertamiento. La justificación es operacional. Dado que el sistema conserva todo flujo procesado junto con su puntuación, los flujos maliciosos que quedan bajo el umbral no se pierden: permanecen almacenados y siguen siendo visibles en los tableros, de modo que elevar el umbral desplaza el evento de la cola de alertas al material de consulta, pero no lo elimina. La medición de la sección 6.4 respalda la elección: bajo partición temporal, al pasar de 0,5 a 0,7 la precisión sube de 0,887 a 0,904 y la carga de triaje baja de 1,127 a 1,107, mientras la cobertura apenas varía (de 28,25 % a 28,21 %). En cambio, un umbral más permisivo multiplicaría las alertas de severidad alta y erosionaría la atención del operador, fenómeno conocido como fatiga de alertas. El AUC-ROC (Area Under the Curve- Receiver Operating Characteristic; métrica estándar para medir qué tan eficiente es un modelo distinguiendo entre dos clases) del modelo desplegable, de 0,9994 ante familias conocidas, indica que su capacidad discriminativa no depende de un punto de corte particular, aunque ante familias no observadas desciende a 0,7535. Sin embargo, el punto de corte puede recalibrarse según la tolerancia a falsos positivos de cada organización sin reentrenar el modelo ni alterar ningún otro componente del sistema.

E.6.3 Almacenamiento Analítico (PostgreSQL)

PostgreSQL 15 [12] se seleccionó por tres razones. La primera es el soporte nativo de INET para direcciones IP, que permite expresar filtros por rango de red sin conversiones en la consulta. La segunda es la extensión pg_stat_statements, que hace posible instrumentar el costo de las consultas analíticas y detectar las que degradan el tablero del dashboard. La tercera es su comportamiento acreditado ante cargas mixtas de escritura sostenida y lectura concurrente, que es precisamente el perfil de un colector de flujos consultado en línea. La configuración del servidor ajusta shared_buffers a 256 MB, effective_cache_size a 1 GB y max_connections a 200.

La tabla principal netflow_data reúne sus campos agrupados en tres categorías: identificación y métricas del flujo, características derivadas para el modelo, y resultados del análisis (is_anomaly, threat_score y labels en formato JSONB). Los índices se eligieron según el patrón de acceso predominante de cada columna: BRIN sobre start_time, apropiado para datos insertados en orden temporal y de tamaño muy inferior al de un índice B-tree equivalente; HASH sobre las direcciones IP, orientado a búsquedas de igualdad; e índices parciales sobre is_anomaly, cuyo tamaño queda acotado a la fracción minoritaria de registros marcados. El Anexo B detalla la definición completa del esquema.

Conviene explicitar aquí una decisión de diseño que atraviesa todo el sistema y que se presta a confusión: lo que el modelo recibe como entrada y lo que el sistema registra y presenta son conjuntos distintos, gobernados por criterios distintos. El clasificador no recibe las direcciones IP, porque hacerlo le permitiría memorizar la identidad de los equipos observados en el entrenamiento en lugar de aprender el comportamiento que los caracteriza, según se argumenta en la sección 4.6. La base de datos, en cambio, conserva la 5-tupla completa junto a la puntuación emitida, porque una alerta que no identifica quién habla con quién es inaccionable para el analista. La restricción es del modelo, no del registro.

E.6.4 Visualización y Monitoreo (Grafana + Prometheus)

Grafana 10.x [13] se configuró con PostgreSQL como fuente de datos primaria, lo que permite formular los paneles directamente en SQL sobre la tabla de flujos, sin capa intermedia de agregación. El tablero principal reúne cinco vistas: volumen de tráfico en tiempo real, distribución de amenazas por severidad, direcciones IP más relevantes por volumen y por puntuación del modelo, línea de tiempo de eventos de seguridad y distribución de protocolos.

Los tableros explotan esa distinción. Junto a las vistas agregadas, el sistema publica el detalle accionable de cada evento: la 5-tupla completa, el exportador de origen, la puntuación del modelo y el tipo de amenaza cuando el detector complementario logra atribuirlo. Se incorporan asimismo dos vistas orientadas a la mitigación activa —el ranking de direcciones de origen por número de flujos con puntuación elevada y el de conversaciones origen-destino ordenadas por puntuación acumulada—, que son las que permiten pasar de la constatación de una amenaza a la decisión de bloquearla. Dado que ese detalle expone la topología interna de la organización, el acceso se segrega por rol: las vistas agregadas quedan disponibles para la operación de red, mientras que el detalle con direcciones se restringe al equipo de seguridad. Cabe consignar, además, que en determinados marcos regulatorios una dirección IP puede constituir dato personal, lo que refuerza tanto la segregación de acceso como la política de retención acotada de la sección 3.5.

En paralelo, Prometheus [35] recolecta métricas operacionales de la propia plataforma mediante node-exporter (uso de CPU, memoria y entrada/salida) y postgres-exporter (conexiones activas y rendimiento de consultas), que alimentan un tablero de infraestructura independiente. La separación entre ambos planos —detección y salud operacional— es deliberada porque así permite distinguir una caída en el número de amenazas detectadas atribuible a la ausencia de tráfico malicioso de otra causada por una degradación del pipeline, discriminación que un tablero único vuelve difícil.

E.6.5 Proxy Inverso (Nginx)

Nginx unifica el acceso a Grafana (puerto 3000) y a FastAPI (puerto 8000) bajo los puertos 80 y 443, de modo que ambos servicios se alcanzan desde un único nombre de host y ninguno queda expuesto de forma directa. Su configuración incorpora las cabeceras de seguridad HTTP habituales y soporte TLS, lo que habilita acceso HTTPS para todos los servicios de usuario final. Los registros de acceso se persisten en el volumen nginx_logs y quedan disponibles para auditoría con independencia del ciclo de vida del contenedor.

E.7 Diseño Experimental del Plan de Evaluación Operacional

Tabla E.2. Diseño experimental del plan de evaluación operacional

Experimento	Variable manipulada	Métrica de respuesta	Duración prevista
E1: Carga sostenida	Volumen de flujos generados (generador controlado)	Throughput, latencia de procesamiento, CPU/memoria	72 horas continuas
E2: Latencia de detección	Inyección de ataques simulados (PortScan, SYN flood)	MTTD desde inicio del ataque hasta alerta	20 repeticiones por tipo
E3: Fuente única vs. múltiple	Número de exportadores activos (1 vs. todos)	Tasa de falsos positivos; cobertura de tráfico este-oeste	24 horas por configuración

E4: Escalabilidad	Número de exportadores sintéticos adicionales	Punto de saturación del procesador; backlog	Incremental hasta saturación
E5: Calidad de normalización	Fabricante del exportador (tráfico sintético idéntico)	Error de contadores y campos imputados por fabricante	1 sesión por fabricante

La Tabla E.2 detalla el diseño de los cinco experimentos previstos. De ellos, E3 es el de mayor relevancia para la hipótesis y contrastará la operación con un único exportador de borde frente a la operación con todos los exportadores del laboratorio activos, midiendo tanto la tasa de falsos positivos como la cobertura de tráfico este-oeste. El resultado esperado —fundado en el razonamiento teórico de la sección 2.5 y en la literatura sobre monitoreo de flujos distribuido [2]— es una reducción de falsos positivos igual o superior al 30 % y la aparición de visibilidad sobre el tráfico lateral, estructuralmente ausente en la captura perimetral. El experimento E2, por su parte, empleará los ataques simulados como verdad de terreno para medir el MTTD (Mean Time to Detect) real del sistema. Cabe subrayar que E3 requiere la implementación previa de la deduplicación entre exportadores y de la validación cruzada descritas en el Capítulo 3, aún no construidas, lo que ordena naturalmente la secuencia de trabajo futuro.

La ejecución difiere del diseño en tres experimentos. E2 se ejecutó sobre tráfico real y no sobre ataques inyectados, tomando el inicio del flujo como instante de referencia; para un sondeo de un paquete ambos instantes coinciden, pero no para ataques de mayor duración, cuya medición sigue pendiente. E5 se ejecutó como caracterización observacional, pues la operación continua dejó ver, exportador por exportador, los defectos que el experimento buscaba medir, y la Tabla E.3 los resume. No aísla el efecto de cada fabricante con un estímulo idéntico, pero describe los defectos tal como el sistema los encuentra. E3 no es ejecutable con el procesador actual, porque solo se puntúa la copia del cortafuegos (sección D.12.2).

Tabla E.3. Defectos de normalización observados por exportador (E5 observacional)

Defecto	**Exportador**	**Evidencia**
Contador de retorno fabricado: replica el de ida	Cortafuegos perimetral	Idéntico al de ida en el 100 % de los registros examinados
Contador de retorno ausente	Equipo de borde y router de sucursal	Cero en el 100 % de los registros
Resolución de duración heterogénea	Los tres exportadores	1 s, 10 ms y 1 ms según el equipo
Banderas TCP no exportadas	Cortafuegos perimetral	Campo vacío en la totalidad de un barrido de 1,9 millones de flujos
Desbordamiento del sysUpTime de 32 bits	Cortafuegos perimetral	Episodio fechado exactamente un período del contador (49,71 días) en el futuro

E.7.1 Medición de la Carga de Triaje en el Entorno de Destino

La dependencia de la carga de triaje respecto de la proporción de tráfico malicioso, descrita en la sección 6.4, tiene dos consecuencias para un despliegue real. La carga debe volver a medirse en el entorno de destino antes de comprometer una cifra de mejora, para lo cual basta contabilizar los eventos presentados y los que el analista confirma como amenazas durante un período acotado. Y en entornos de muy baja proporción de tráfico malicioso el umbral de decisión adquiere una importancia mayor que en este experimento, porque al elevarlo sacrifica cobertura pero contiene el crecimiento de la carga, compensación que el operador debe calibrar según su capacidad de revisión.

E.8 Primeras Mediciones sobre Tráfico Real: Procedimientos y Detalle

Esta sección recoge el detalle de las mediciones que las secciones 5.4 a 5.7 del cuerpo resumen, incluidos el diagnóstico del desfase de los relojes de los exportadores y el examen del espacio de características que explica el techo de cobertura observado.

La primera medición corresponde al experimento E2, la latencia extremo a extremo. Antes de poder reportarla fue necesario corregir un defecto que la medición misma puso al descubierto: un 31 % de los flujos presentaba latencia negativa, es decir, afirmaban haber comenzado después de quedar almacenados. La causa se estableció comparando, para cada archivo del colector y cada exportador, la marca de fin del último flujo con el instante en que el colector cerró el archivo, magnitudes ambas ajenas a la base de datos. Dos de los exportadores resultaron desincronizados: el cortafuegos perimetral, con el 99,4 % de los registros, adelantado 239,0 s (desviación típica 2,0 s), y un router secundario, con el 0,6 %, atrasado 1.272,9 s; ninguno estaba sincronizado por NTP. La comprobación de cierre es que al corregir el desfase no subsiste ninguna latencia negativa y el mínimo pasa a ser +4,3 s.

Corregida esa desviación y restringiendo la medición al régimen estacionario, la latencia presenta una mediana de 297,7 s y un percentil 95 de 489,4 s sobre 212.421 flujos. La componente atribuible al sistema, medida entre el fin del flujo y su almacenamiento con puntuación, asciende a 285,2 s, equivalente a 0,95 ventanas de rotación del colector: la latencia queda así explicada por el parámetro de rotación y no por el tiempo de inferencia, lo que confirma por medición la composición que la sección 3.2 anticipaba y sitúa en el acortamiento de la ventana la vía para reducirla. Se reportan la mediana y el percentil 95, y no la media ni el máximo, porque estos resultan dominados por la cola de reproceso y por el exportador desincronizado.

El examen del techo de cobertura se realizó sobre 220.937 flujos entrantes no solicitados de un mismo día, el clasificador emitió 2.220 alertas, todas sobre los siete puertos de destino que una de sus características codifica como sensibles —los mismos de la regla R1 de la línea base— y ninguna sobre los 42.680 puertos restantes; la puntuación media fue de 0,4856 dentro de ese conjunto y de 0,0034 fuera. El tráfico observado contiene, sin embargo, reconocimiento equivalente sobre servicios no cubiertos por esa lista —bases de datos, escritorio remoto alterno, almacenes de datos, transferencia de archivos y correo— desde decenas de orígenes distintos en cada caso. La causa no reside en el entrenamiento sino en el espacio de características: restringida la comparación a sondeos equivalentes —un paquete por sentido, duración nula, protocolo TCP—, el vector de entrada de un sondeo detectado y el de uno no detectado coinciden en todas sus componentes salvo en la que indica si el puerto pertenece a la lista. El clasificador no dispone, por tanto, de información con la que distinguirlos, y ningún umbral lo remedia: los sondeos no cubiertos alcanzan puntuaciones máximas del orden de 0,036 mientras el tráfico legítimo de redes de distribución de contenido llega a 0,671, de modo que todo umbral que excluya a este último excluye también a aquellos. Es la misma manifestación del problema de la tasa base: la proporción de reconocimiento entre el tráfico no cubierto es del orden del 1,7 %, y a esa prevalencia una discriminación modesta resulta operacionalmente inservible.

Corregidos los relojes y verificados los tres por cabecera —desviaciones de −0,4 s, −0,6 s y +0,1 s—, la medición de E2 se repitió sobre 83.431 flujos en régimen estacionario, con una mediana de 230,4 s, un percentil 95 de 445,4 s y una componente del sistema de 228,1 s, sin latencias negativas y con un mínimo de +8,7 s. La descomposición por protocolo entrega 175,4 s en TCP, 326,1 s en UDP y 240,7 s en ICMP, y es la magnitud estable, pues la agregada es su mezcla: la mediana pasó de 302,9 s a 230,4 s entre dos ventanas sin que el sistema cambiara, solo porque la proporción de flujos TCP cortos aumentó al hacerse visibles nuevos segmentos. El modelo aditivo predice ambas medianas por protocolo a partir de media ventana de rotación y el temporizador de expiración del exportador, 165 s en TCP y 330 s en UDP. Conviene señalar que una versión anterior de este anexo declaraba sin explicar una separación estable de unos 155 s entre el tráfico dirigido al rango público propio y el resto; con relojes verificados esa diferencia es de 20 s y la explica la mezcla de protocolos de cada grupo, de modo que era un artefacto del desfase y no un fenómeno.

La carga de triaje sobre tráfico real se midió mediante adjudicación estratificada. Se declaró primero el espacio de direcciones propio, se extrajo una muestra por estrato —alertas del clasificador, alertas de las reglas y flujos no alertados— y cada ficha se resolvió con un procedimiento que aplica condiciones fijadas de antemano sin mostrar el veredicto del modelo; los casos que las condiciones no resuelven se reservan para decisión manual. El estimador pondera cada estrato por su tamaño en la población y reporta intervalos de Wilson. La carga resultó de 1,028 para el clasificador y de 4,356 para las reglas, una reducción del 76,4 % que no baja de un 60 % aproximadamente en el peor caso de los intervalos, con una cobertura del modelo del orden del 41 % de la alcanzada por las reglas. Esta medición es anterior a la corrección del contador de retorno y empleó una declaración de servicios más estrecha; la reemplaza el censo descrito al final de esta sección.

La primera versión del detector complementario produjo del orden de 27.000 alertas diarias. El 98,6 % correspondía a anomalías del bosque de aislamiento, efecto de la cuota fija de contaminación del 10 % repetida en cada ejecución horaria, y de las alertas de escaneo, el 89,7 % (intervalo del 85 % al 93 %) eran tráfico de respuesta de servidores externos. La segunda versión, descrita en la sección E.4, procesó en su primera ejecución 977.316 flujos, de los cuales excluyó 539.606 como respuestas, y sobre 14.145 orígenes señaló 141 anomalías y 5 sondeos. Durante el cambio de versión el detector estuvo fuera de servicio algunas horas sin que el procesador lo advirtiera, lo que aconseja que la orquestación registre como error la ausencia o el fallo de cualquier módulo.

La carga de triaje de la segunda versión se estimó con el procedimiento de la medición anterior, sobre una población de 266 alertas de una ventana de veinticuatro horas delimitada por marcas de tiempo absolutas. Se definieron tres estratos —sondeos, anomalías con origen externo y anomalías con origen en el espacio propio— y se adjudicaron 62 fichas, presentadas sin la puntuación del modelo. La precisión resulta de 1,000 en los sondeos, de 0,667 en las anomalías externas y de 0,000 en las internas, que en la jornada observada corresponden todas a tráfico de respuesta o a servicios propios. Ponderando cada estrato por su tamaño, la precisión global es de 0,654 y la carga de triaje de 1,53 eventos por amenaza hallada, con un intervalo de 1,30 a 1,95 derivado del intervalo de Wilson del estrato dominante. Una ficha quedó indeterminada y se excluyó del denominador; contarla como benigna eleva la carga a 1,55, de modo que la conclusión no depende de ese criterio. Dos veredictos se fundaron en la reputación documentada del origen y no en la forma del tráfico, distinción que se consigna porque esa evidencia no es observable desde los registros de flujo.

La cobertura se midió sin intervención humana. Un rango propio sin servicios publicados recibe, por construcción, solo tráfico no solicitado, de modo que los orígenes que contactan en él más de diez puertos en menos de una hora —el mismo criterio que el detector aplica— constituyen una verdad de terreno. De 54 orígenes así confirmados, el detector alertó 23, esto es, un 42,6 % con un intervalo del 31 % al 55 %. El techo es de diseño y no de discriminación, pues con una contaminación del 1 % sobre unos veinte mil orígenes el algoritmo dispone de un número fijo de alertas por ejecución, del orden de doscientas en esa medición y de cien en la configuración vigente descrita en la sección E.4. Elevar esa proporción aumenta la cobertura y la carga en la misma dirección, y es la perilla equivalente al umbral de decisión del clasificador. La medición exige además excluir las direcciones del rango en que se levantó un servicio durante el período, porque dejan de ser espacio sin servicios.

Esa medición se repitió con el telescopio de la sección E.10, que no exige el criterio del detector y admite cualquier forma de contacto. En setenta y dos horas, 20.140 orígenes externos enviaron flujos a direcciones sin equipos y el sistema alertó al 17,8 % de ellos por alguna de sus vías: la agregada a 3.103, el clasificador por flujo a 617 y la regla de sondeo a 37. La Tabla E.4 separa la cobertura por intensidad, y muestra que el techo de presupuesto descrito arriba se gasta en los orígenes más intensos.

Tabla E.4. Cobertura del sistema sobre los orígenes externos del telescopio según su intensidad (72 h)

Flujos al telescopio	Orígenes	Alertados	Cobertura	Direcciones medias	Puertos medios
1	6.325	554	8,8 %	1,0	1,0
2 a 5	7.468	851	11,4 %	2,6	2,4
6 a 20	4.953	1.065	21,5 %	4,1	8,7
21 a 100	1.133	847	74,8 %	3,9	24,6
Más de 100	261	261	100,0 %	4,6	168,1
Total	20.140	3.578	17,8 %		

Dos efectos transitorios quedan documentados. El primero es que, al reemplazar la versión del detector, las alertas de la versión anterior permanecen activas y suprimen por deduplicación a sus orígenes durante veinticuatro horas; al expirar produjeron un pico aislado de 118 alertas en una hora, frente a una mediana horaria de 9, y 108 de esos orígenes ya habían sido señalados por la versión previa, por lo que el pico se excluye de la población adjudicada y se recomienda marcar como resueltas las alertas de la versión saliente. El segundo es que los falsos positivos restantes comparten una propiedad, pues corresponden a agentes de monitoreo, a

mediciones de laboratorio y a sesiones hacia servicios propios tras la traducción de direcciones, todos dirigidos a puertos que la organización publica. Es la misma corrección que el clasificador requiere, lo que refuerza la prioridad de la característica de puerto declarado en el trabajo futuro.

Dos lecciones de observabilidad completan el registro, ambas surgidas de fallas reales. La primera es que el detector estuvo fuera de servicio varias horas mientras el orquestador registraba su ausencia como advertencia con texto de reintento: el registro existía, pero su severidad no correspondía a la consecuencia, que era la pérdida permanente de una de las dos vías de detección. La segunda es que los registros del contenedor se pierden al reconstruirlo, justo cuando se sustituye el componente que falló, de modo que el estado de cada módulo debe persistirse en el almacenamiento analítico y no solo en la bitácora del proceso. De ahí que la orquestación distinga la ausencia del módulo de su fallo en ejecución, registre ambos como error y deje constancia en una tabla de salud consultable, junto con una bitácora de cambios de captura que permite referir cada medición al régimen vigente cuando se tomó.

La corrección del contador de retorno partió de una comparación registro a registro: en el cortafuegos el contador de paquetes de vuelta es idéntico al de ida en todos los registros, y en los otros dos exportadores es cero. Ninguno de los dos valores describe la respuesta, que NetFlow v9 registra como un flujo aparte. La regla adoptada busca, para cada flujo, el de la 5-tupla invertida de la misma sesión dentro de una ventana acotada y distingue tres casos: el sentido opuesto existe y aporta la respuesta; el exportador permitiría verlo y no aparece, lo que constituye una ausencia observada; o el exportador no permite saberlo, y el registro se excluye de la inferencia en lugar de puntuarse con un cero que no es dato. En el prototipo, este último caso comprende hoy los registros de los dos exportadores distintos del cortafuegos; incorporarlos exige declarar, en el perfil de cada fabricante, cuándo el equipo observa ambos sentidos de una sesión, tarea que forma parte de la paridad de características (M1). Contra la hipótesis inicial de que el emparejamiento degradaría la detección, los flujos emparejados recibieron en conjunto puntuaciones del orden de nueve veces superiores; la excepción fueron los sondeos denegados, que al quedar correctamente sin respuesta perdieron puntuación, como expone la sección 5.5.

El experimento natural fue un barrido real desde un único origen externo, del orden de 1,9 millones de flujos en unos diecisiete minutos contra el espacio propio. Ningún flujo superó el umbral: los 203 dirigidos a puertos sensibles puntuaron 0,6469 o 0,6733 y el resto quedó próximo a cero. La distancia al umbral no se debe a la calibración sino a la información disponible, pues sin banderas TCP un sondeo no completado es indistinguible de una conexión fallida. El detector por agregados tampoco lo alertó en un principio, porque el origen tenía una alerta previa de menor magnitud; tras la corrección de la sección E.4 lo hizo en la ejecución siguiente.

La inyección controlada se ejecutó en dos pruebas desde la sede remota. En la primera, conexiones manuales a once puertos —21, 22, 23, 25, 53, 80, 110, 143, 443, 993 y 995— de un blanco propio que responde produjeron alertas solo en los puertos 22 (8 de 44 flujos) y 23 (4 de 43), con puntuación máxima de 0,9731, y en los nueve restantes la puntuación máxima osciló entre 0,007 y 0,027. En la segunda, un barrido exhaustivo en seis fases con una herramienta estándar de sondeo, hacia un blanco que el cortafuegos protege, dejó 1.692 flujos puntuados en la copia del cortafuegos y una única alerta, sobre el puerto 1433 con 0,7281, una exhaustividad del orden del 0,06 %. Las copias del mismo barrido registradas por los demás exportadores no se puntuaron, por la exclusión descrita en la sección D.12.2. Las dos pruebas son la única verdad de terreno de exhaustividad con ataques controlados y coinciden con el experimento natural: la puntuación depende del puerto y no de la forma del sondeo.

El censo de precisión adjudicó la totalidad de las 562 alertas del clasificador en una ventana de veinticuatro horas delimitada por marcas de tiempo absolutas, con condiciones fijadas de antemano y sin mostrar la puntuación. Se declara benigna la alerta cuyo origen pertenece al espacio propio, la que se dirige a un servicio que la organización presta y la que responde a una conexión iniciada desde la red propia; el resto, sondeos no solicitados a servicios no prestados, se declara amenaza. La primera adjudicación declaró benignas por respuesta solicitada 147 alertas y arrojó una precisión del 61,7 %. El criterio contenía el mismo defecto de sentido de la sección 5.2: contaba como conexión iniciada desde la red propia cualquier flujo del equipo propio hacia el origen, incluidas sus respuestas. Restringido a flujos efectivamente iniciados y reevaluadas solo las fichas que el cambio podía afectar, 128 pasaron a amenaza, las respuestas solicitadas quedaron en 19 (3,4 %) y la precisión en el 84,5 % (475 de 562), con una carga de 1,18. El umbral de evidencia de sondeo se fijó de antemano igual al del detector, diez pares propios; con cincuenta la precisión sería del 78,1 %, y si las quince

fichas dudosas restantes fueran amenazas, del 87,2 %. Ninguna ficha quedó indeterminada, y de las 562 alertas 558 también las emite la línea base; las cuatro que solo emite el clasificador resultaron benignas.

Contada por par de origen y destino en lugar de por flujo, ponderando cada ficha por la inversa del número de veces que su par aparece en el censo, la precisión es del 88,7 %, sobre 438 condiciones con un máximo de diez repeticiones: las condiciones repetidas tienden a lo benigno y la cifra por flujo es la conservadora. Se reportan ambas.

El criterio de servicio prestado exigió una decisión propia. Declararlo desde la configuración no es posible, porque las políticas del cortafuegos admiten todos los servicios sobre rangos completos y la topología es enrutada, con una sola publicación explícita. Se descartaron dos criterios observacionales, la popularidad del puerto y un mínimo de cuatro paquetes de respuesta, que capturaba a los sondeadores que completan el saludo TCP, y se adoptó el de conversación: un par dirección-puerto es un servicio cuando sostiene al menos 20 conversaciones con un mismo origen o devuelve al menos 15 paquetes. Los casos límite, como servicios de enrutamiento o de administración, se declararon a mano con un motivo escrito. El inventario resultante se conserva en una tabla del almacenamiento analítico, y cada alerta adjudicada conserva su motivo.

La transferibilidad de la línea base se examinó regla por regla. R3 exige una duración inferior al percentil 1, que es cero en los 25 días observados, de modo que ningún flujo la satisface. R4 exige cero paquetes de vuelta, y antes del emparejamiento se disparaba en el 0,72 % de los flujos porque el cortafuegos copiaba el contador; después, en el 77,85 %. Su poder discriminante resultó invertido, pues en la muestra adjudicada no la cumple ninguna de 84 amenazas y sí 9 de 57 flujos benignos. Ambas reglas tienen sentido sobre los registros bidireccionales de CICIDS2017 y no se trasladan, y la línea base corregida se reduce a R1 o R2. Las 84 amenazas se dirigen, además, a puertos de R1, porque la condición de adjudicación que declara amenaza el sondeo a un servicio no prestado y la regla R1 dependen ambas del puerto. Con esa verdad de terreno la carga de la línea base queda determinada por el criterio de adjudicación y no constituye una medición independiente.

La latencia por protocolo se midió de nuevo sobre seis horas de tráfico del cortafuegos, con su reloj sincronizado por NTP —desviación de 11 a 21 ms— y el temporizador activo ya reducido. Las medianas fueron de 201,5 s en TCP y 334,2 s en UDP hacia el rango propio, y de 206,9 s y 363,7 s hacia el resto, prácticamente iguales medidas desde el inicio o desde el fin del flujo, porque la mayoría de los flujos son de un solo paquete. La latencia se descompone en el temporizador de sesión del cortafuegos tras el último paquete, de uno a diez segundos en TCP y de 180 s en UDP; la espera al cierre del archivo del colector, de 150 s en promedio; el sondeo del procesador, de hasta 30 s, y el procesamiento. La suma predice del orden de 170 a 200 s en TCP y de 350 s en UDP, y confirma la explicación por composición de la medición anterior. La palanca ajustable es la rotación del colector, pues acortar el temporizador UDP del cortafuegos cortaría sesiones legítimas de larga inactividad, como las de telefonía IP.

Las características derivadas de la duración se calculan en segundos, mientras CICIDS2017 las expresa en microsegundos. Recalcular las puntuaciones en la escala del corpus desplazó la puntuación como máximo en 0,0072 y no movió ningún flujo a través del umbral, coherente con el 4,90 % de la ganancia que esas seis características reúnen en el modelo desplegable. La incoherencia se documenta y el código se mantiene, porque corregirla obligaría a rehacer las mediciones sin un efecto demostrable; la corrección corresponde al próximo reentrenamiento.

E.9 Arquitectura de Dos Capas: Entradas, Roles y Salidas

La Figura E.1 resume el recorrido de un flujo desde su exportación hasta las alertas y su evaluación, y distingue qué entra en cada capa de detección y qué produce cada una.

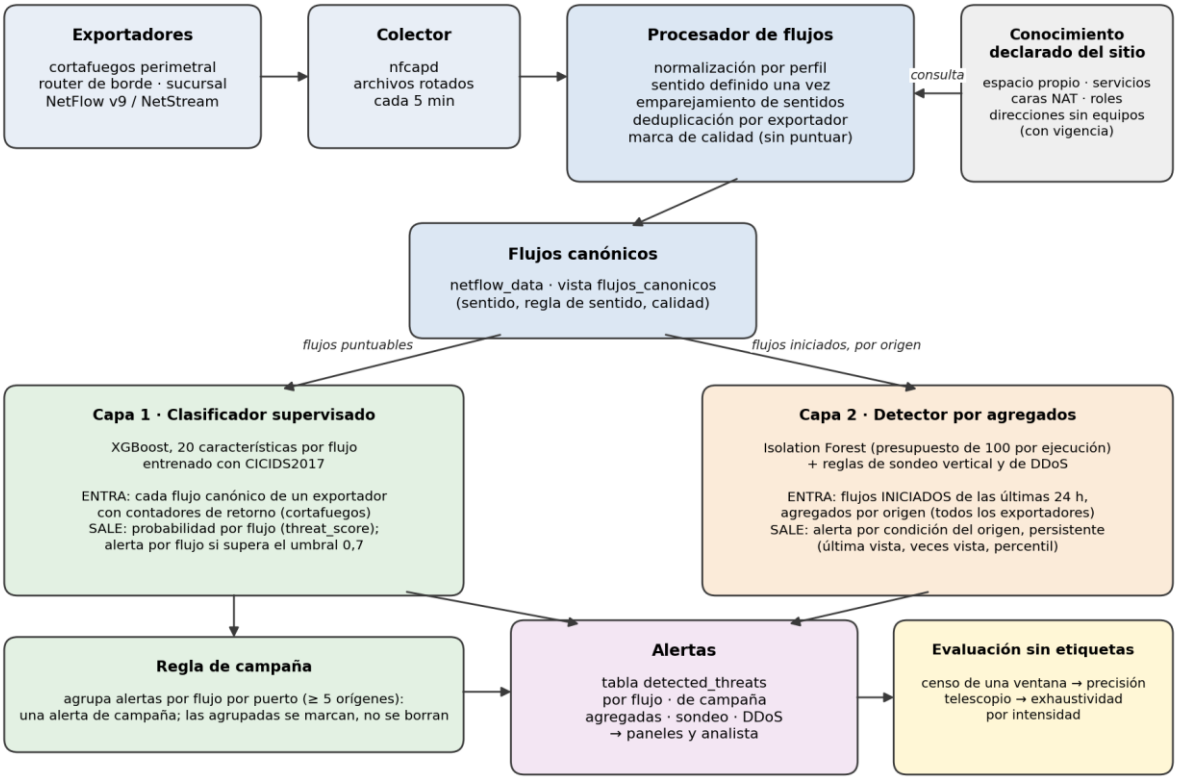


Figura E.1. Arquitectura de dos capas del prototipo: el clasificador supervisado juzga flujos y el detector por agregados juzga orígenes; ambos escriben en la misma tabla de alertas, y la evaluación se realiza sin etiquetas.

El procesador entrega a ambas capas el mismo flujo canónico, con su sentido ya definido, y las diferencias empiezan en la selección. El clasificador recibe cada flujo puntuable, es decir, de un exportador que reporta contadores de retorno, que en el prototipo es el cortafuegos; los registros de los demás exportadores se marcan como no puntuados en lugar de recibir un cero que no es dato. Su salida es una probabilidad por flujo y, sobre el umbral de 0,7, una alerta por flujo, que la regla de campaña agrupa por puerto cuando cinco o más orígenes coinciden. El detector recibe, en cambio, los flujos iniciados de las últimas veinticuatro horas de todos los exportadores, agregados por origen, y su salida es una alerta por condición del origen —anomalía, sondeo vertical o denegación de servicio— que persiste mientras la condición se observa. El conocimiento declarado del sitio interviene en el procesador, que lo consulta para definir el sentido, y en la evaluación, que lo usa para adjudicar y para delimitar el telescopio. Las dos capas no se reemplazan: el clasificador aporta precisión dentro de lo que aprendió y el detector cubre lo que solo se ve en el agregado, y la medición sobre el telescopio cuantifica ese reparto.

E.10 Conocimiento Declarado del Sitio, Sentido y Telescopio: Procedimientos

El sentido se define en el procesador con una precedencia fija. Si uno solo de los dos extremos usa un puerto de servicio —inferior a 1024 o declarado como servicio publicado—, inicia la conversación el otro extremo. Si la asimetría no lo resuelve y existe el flujo de sentido opuesto del mismo exportador dentro de una ventana de diez segundos, inicia el que empezó primero, con una tolerancia de un segundo; si ambos empiezan en el mismo segundo, el sentido queda indeterminado. Los consumidores aplican una política declarada para los indeterminados: el tráfico de un origen externo hacia uno propio cuenta como iniciado, para no ocultar sondeos, y el de uno propio hacia uno externo como respuesta, para no atribuir sondeos a equipos propios. Los protocolos que usan el mismo puerto en ambos extremos, como la sincronización horaria o el intercambio de claves de

túneles, quedan fuera de la regla de asimetría y dependen del orden o del empate. La medición en paralelo de la regla anterior del detector, sobre una ventana de veinticuatro horas, mostró que discrepaba en el 17 % de los flujos que la nueva definición resuelve: en el 98 % de los casos la regla anterior llamaba respuesta a una conexión iniciada, y el 91 % eran flujos hacia un puerto de servicio cuyo sentido opuesto aparecía antes, efecto de la segmentación de sesiones largas; el 68 % de esos flujos venía de orígenes externos. La regla anterior se conserva solo para los flujos sin sentido definido.

El conocimiento del sitio se declara en tablas con vigencia, que se cierran con una fecha en lugar de borrarse. La declaración de roles asigna a cada rango uno de ocho papeles —sin equipos, cara de traducción de salida, infraestructura de medición, usuarios de VPN, monitoreo, red privada, laboratorio y servidor—, y una función devuelve el rol del rango más específico que contiene una dirección en un instante dado, lo que permite clasificar a posteriori cualquier flujo o alerta sin modificar el código de los detectores. Las caras de traducción de salida son siete, seis del cortafuegos y una del exportador secundario. La declaración de servicios tiene dos versiones: la primera, con la que se hizo el censo, se conserva intacta, y la segunda declara solo los servicios confirmados en la configuración y retira la conjetura sobre el segmento sin equipos. La guarda de pre-registro del procedimiento de declaraciones impidió recargarlas después de emitidos los veredictos, y la única corrección aplicada a mano —agregar las caras de traducción que faltaban— se documentó tras verificar que ninguna ficha del censo dependía de ellas.

El telescopio está formado por las cinco direcciones libres de un segmento público cuyo último equipo se retiró hace más de tres meses. La política del cortafuegos acepta el tráfico entrante hacia ese segmento, de modo que todo lo que llega se exporta; si la política pasara a denegarlo, el telescopio quedaría ciego, porque lo denegado no se exporta, y su verificación debe comprobar tanto que nadie responda como que siga llegando tráfico. En toda la retención salieron de esas direcciones 28.396 flujos, de los cuales 28.388 son respuestas TCP desde los puertos de voz 5060 y 2000, de uno a tres paquetes de unos 52 bytes, que el módulo de voz del cortafuegos en modo proxy genera en nombre de cualquier destino permitido; los ocho restantes son pares entre direcciones vecinas del propio telescopio, consultas DNS y paquetes IPv6 con origen falsificado. La excepción del intermediario quedó declarada en una vista de verificación que debe permanecer en cero. Del tráfico entrante, el origen más persistente enviaba de forma continua al puerto de un servicio de registro de eventos, desde un único origen con ocho puertos fuente y sin respuesta alguna, y el detector lo alertó cada día: el patrón corresponde a agentes de un sitio remoto que siguen apuntando a un colector retirado, tráfico no solicitado y de intención benigna que la verdad de terreno no distingue del hostil.

La operación sobre tráfico real exigió además dos correcciones de mantenimiento del almacenamiento analítico, documentadas por su efecto sobre cualquier medición. Los índices por bloques de las marcas de tiempo no se resumían desde el último mantenimiento automático, de modo que una consulta de una hora leía más de trescientos mil bloques; resumidos los rangos y activado su resumen automático, la misma consulta leyó 5.705 bloques y bajó de 40 s a 1,5 s. Y la verificación de frescura calculaba un máximo sobre la tabla completa en cada ejecución, carga que crecía con los datos, y pasó a resolverse por la llave primaria. Una tabla de telemetría que solo recibe inserciones necesita, en suma, un mantenimiento que el predeterminado no asegura a medida que crece.

Anexo F: Lista de Acrónimos

Este anexo reúne los acrónimos y las siglas empleados a lo largo del informe, con su forma desarrollada y su equivalente en castellano cuando corresponde. Se ordenan alfabéticamente. Se omiten los nombres de productos, de fabricantes y de bibliotecas de programación, que se introducen en el texto en su primera aparición.

Tabla F.1. Acrónimos y siglas empleados en el informe

Acrónimo	Significado
ACL	Access Control List — lista de control de acceso
API	Application Programming Interface — interfaz de programación de aplicaciones
AUC-ROC	Area Under the Receiver Operating Characteristic Curve — área bajo la curva característica operativa del receptor
BGP	Border Gateway Protocol — protocolo de pasarela de frontera
CICIDS2017	Canadian Institute for Cybersecurity Intrusion Detection System dataset 2017
CPU	Central Processing Unit — unidad central de procesamiento
CSF	Cybersecurity Framework — marco de ciberseguridad del NIST
DDoS	Distributed Denial of Service — denegación de servicio distribuida
DMZ	Demilitarized Zone — zona desmilitarizada de la red
DoS	Denial of Service — denegación de servicio
DPI	Deep Packet Inspection — inspección profunda de paquetes
F1-Score	Media armónica entre precisión y cobertura
FNF	Flexible NetFlow — implementación configurable de NetFlow de Cisco
FTP	File Transfer Protocol — protocolo de transferencia de archivos
gNMI	gRPC Network Management Interface — interfaz de gestión de red sobre gRPC
HTTP / HTTPS	HyperText Transfer Protocol (Secure) — protocolo de transferencia de hipertexto (seguro)
IANA	Internet Assigned Numbers Authority — autoridad de asignación de números de Internet
IAT	Inter-Arrival Time — tiempo entre llegadas de paquetes
ICMP	Internet Control Message Protocol — protocolo de mensajes de control de Internet
IDS / IPS	Intrusion Detection / Prevention System — sistema de detección / prevención de intrusiones
IETF	Internet Engineering Task Force — grupo de trabajo de ingeniería de Internet
IP	Internet Protocol — protocolo de Internet
IPFIX	IP Flow Information Export — exportación de información de flujos IP (RFC 7011)
JSON / JSONB	JavaScript Object Notation (Binary) — notación de objetos de JavaScript (binaria)
MPLS	Multiprotocol Label Switching — conmutación de etiquetas multiprotocolo
MTTD	Mean Time To Detect — tiempo medio de detección
ML	Machine Learning — aprendizaje automático
MSSQL	Microsoft SQL Server

NaN	Not a Number — valor numérico indefinido
NETCONF	Network Configuration Protocol — protocolo de configuración de red
NIST	National Institute of Standards and Technology — instituto nacional de estándares y tecnología (EE. UU.)
NTP	Network Time Protocol — protocolo de sincronización horaria
P2P	Peer to Peer — entre pares
PCAP	Packet Capture — captura completa de paquetes
PSH / URG / SYN / FIN / RST / ACK	Banderas de control del encabezado TCP
QoS	Quality of Service — calidad de servicio
RDP	Remote Desktop Protocol — protocolo de escritorio remoto
RFC	Request for Comments — documento normativo de la IETF
RESTCONF	Protocolo de configuración de red sobre HTTP con modelos YANG
SMB	Server Message Block — protocolo de compartición de archivos
SMOTE	Synthetic Minority Over-sampling Technique — técnica de sobremuestreo sintético de la clase minoritaria
SNMP	Simple Network Management Protocol — protocolo simple de gestión de red
SOAR	Security Orchestration, Automation and Response — orquestación, automatización y respuesta de seguridad
SOC	Security Operations Center — centro de operaciones de seguridad
SQL	Structured Query Language — lenguaje de consulta estructurado
SSH	Secure Shell — intérprete de órdenes seguro
SVM	Support Vector Machine — máquina de vectores de soporte
TCP	Transmission Control Protocol — protocolo de control de transmisión
TLS	Transport Layer Security — seguridad de la capa de transporte
ToS	Type of Service — tipo de servicio (campo del encabezado IP)
UDP	User Datagram Protocol — protocolo de datagramas de usuario
UEBA	User and Entity Behavior Analytics — análisis de comportamiento de usuarios y entidades
UTC	Coordinated Universal Time — tiempo universal coordinado
VLAN	Virtual Local Area Network — red de área local virtual
VRF	Virtual Routing and Forwarding — enrutamiento y reenvío virtual
XGBoost	Extreme Gradient Boosting — potenciación extrema del gradiente
XSS	Cross-Site Scripting — ejecución de guiones entre sitios
YANG	Yet Another Next Generation — lenguaje de modelado de datos de red