Universidad de San Carlos de Guatemala
Facultad de Ingeniería
Escuela de Estudios de Postgrado
Maestría en Tecnologías de la Información y Comunicación

IMPLEMENTACIÓN DE UN SISTEMA DE MONITOREO EN TIEMPO DE EJECUCIÓN DE APLICACIONES QUE FUNCIONAN EN CANALES ELECTRÓNICOS DE UNA ENTIDAD BANCARIA PARA DETECCIÓN DE FALLAS DE FORMA PREVENTIVA

Ing. Omar Francisco Mendoza Cajtic
Asesorado por el MSc. Ing. Edwin Estuardo Zapeta Gómez

Guatemala, julio de 2022

UNIVERSIDAD DE SAN CARLOS DE GUATEMALA

FACULTAD DE INGENIERÍA

IMPLEMENTACIÓN DE UN SISTEMA DE MONITOREO EN TIEMPO DE EJECUCIÓN DE APLICACIONES QUE FUNCIONAN EN CANALES ELECTRÓNICOS DE UNA ENTIDAD BANCARIA PARA DETECCIÓN DE FALLAS DE FORMA PREVENTIVA

TRABAJO DE GRADUACIÓN

PRESENTADO A LA JUNTA DIRECTIVA DE LA
FACULTAD DE INGENIERÍA
POR

ING. OMAR FRANCISCO MENDOZA CAJTIC
ASESORADO POR EL MSC. ING. EDWIN ESTUARDO ZAPETA GÓMEZ

AL CONFERÍRSELE EL TÍTULO DE

MAESTRO EN TECNOLOGÍAS DE LA INFORMACIÓN Y COMUNICACIÓN

GUATEMALA, JULIO DE 2022

UNIVERSIDAD DE SAN CARLOS DE GUATEMALA

FACULTAD DE INGENIERÍA

NÓMINA DE JUNTA DIRECTIVA

DECANA	Inga. Aurelia Anabela Cordova Estrada
VOCAL I	Ing. José Francisco Gómez Rivera
VOCAL II	Ing. Mario Renato Escobedo Martínez
VOCAL III	Ing. José Milton de León Bran
VOCAL IV	Br. Kevin Vladimir Cruz Lorente
VOCAL V	Br. Fernando José Paz González
SECRETARIO	Ing. Hugo Humberto Rivera Pérez

TRIBUNAL QUE PRACTICÓ EL EXAMEN GENERAL PRIVADO

DECANA	Inga. Aurelia Anabela Cordova Estrada
DIRECTOR	Mtro. Ing. Edgar Darío Álvarez Cotí
EXAMINADOR	Mtro. Ing. Marlon Antonio Pérez Türk
EXAMINADOR	Mtro. Ing. Everest Darwin Medinilla Rodríguez
SECRETARIO	Ing. Hugo Humberto Rivera Pérez

HONORABLE TRIBUNAL EXAMINADOR

En cumplimiento con los preceptos que establece la ley de la Universidad de San Carlos de Guatemala, presento a su consideración mi trabajo de graduación titulado:

IMPLEMENTACIÓN DE UN SISTEMA DE MONITOREO EN TIEMPO DE EJECUCIÓN DE APLICACIONES QUE FUNCIONAN EN CANALES ELECTRÓNICOS DE UNA ENTIDAD BANCARIA PARA DETECCIÓN DE FALLAS DE FORMA PREVENTIVA

Tema que me fuera asignado por la Dirección de la Escuela de Estudios de Postgrado, con fecha 7 de noviembre de 2020.

Ing. Omar Francisco Mendoza Cajtic

Decanato
Facultad de Ingeniería
24189101- 24189102
secretariadecanato@ingenieria.usac.edu.gt

LNG.DECANATO.OI.526.2022

La Decana de la Facultad de Ingeniería de la Universidad de San Carlos de Guatemala, luego de conocer la aprobación por parte del Director de la Escuela de Estudios de Posgrado, al Trabajo de Graduación titulado: **IMPLEMENTACIÓN DE UN SISTEMA DE MONITOREO EN TIEMPO DE EJECUCIÓN DE APLICACIONES QUE FUNCIONAN EN CANALES ELECTRÓNICOS DE UNA ENTIDAD BANCARIA PARA DETECCIÓN DE FALLAS DE FORMA PREVENTIVA,** presentado por: **Omar Francisco Mendoza Cajtic,** que pertenece al programa de Maestría en artes en Tecnologías de la información y la comunicación después de haber culminado las revisiones previas bajo la responsabilidad de las instancias correspondientes, autoriza la impresión del mismo.

IMPRÍMASE:

UNIVERSIDAD DE SAN CARLOS DE GUATEMALA
DECANA
FACULTAD DE INGENIERÍA

Inga. Aurelia Anabela Cordova Estrada
Decana

Guatemala, julio de 2022

AACE/gaoc

Guatemala, julio de 2022

LNG.EEP.OI.526.2022

En mi calidad de Director de la Escuela de Estudios de Postgrado de la Facultad de Ingeniería de la Universidad de San Carlos de Guatemala, luego de conocer el dictamen del asesor, verificar la aprobación del Coordinador de Maestría y la aprobación del Área de Lingüística al trabajo de graduación titulado:

"IMPLEMENTACIÓN DE UN SISTEMA DE MONITOREO EN TIEMPO DE EJECUCIÓN DE APLICACIONES QUE FUNCIONAN EN CANALES ELECTRÓNICOS DE UNA ENTIDAD BANCARIA PARA DETECCIÓN DE FALLAS DE FORMA PREVENTIVA"

presentado por **Omar Francisco Mendoza Cajtic** correspondiente al programa de **Maestría en artes en Tecnologías de la información y la comunicación** ; apruebo y autorizo el mismo.

Atentamente,

"Id y Enseñad a Todos"

UNIVERSIDAD DE SAN CARLOS DE GUATEMALA
FACULTAD DE INGENIERÍA
DIRECCIÓN
ESCUELA DE POSTGRADO

Mtro. Ing. Edgar Darío Álvarez Cotí
Director
Escuela de Estudios de Postgrado
Facultad de Ingeniería

EEPFI-0498-2022

Guatemala, 30 de mayo 2022

Profesional
Omar Francisco Mendoza Cajtic
Carné: 200714554
Maestría en Tecnología de la Información y la Comunicación

Distinguido Profesional Mendoza Cajtic:

De manera atenta hago constar que de acuerdo con la aprobación del coordinador de maestría y docente-revisor doy el aval a su Informe Final y Artículo Científico titulado: "IMPLEMENTACIÓN DE UN SISTEMA DE MONITOREO EN TIEMPO DE EJECUCIÓN DE APLICACIONES QUE FUNCIONAN EN CANALES ELECTRÓNICOS DE UNA ENTIDAD BANCARIA PARA DETECCIÓN DE FALLAS DE FORMA PREVENTIVA" para que pueda continuar con su proceso de graduación.

Con base en la evaluación realizada hago constar la originalidad, calidad, coherencia según lo establecido en el *Normativo de Tesis y Trabajos de Graduación aprobados por la Junta Directiva de la Facultad de Ingeniería Punto Sexto inciso 6.10 del Acta 04-2014 de sesión celebrada el 04 de febrero de 2014*. Cumpliendo tanto en su estructura como en su contenido, por lo cual el trabajo evaluado cuenta con mi aprobación.

Atentamente,

"Id y Enseñad a Todos"

UNIVERSIDAD DE SAN CARLOS DE GUATEMALA · FACULTAD DE INGENIERÍA · DIRECCIÓN · ESCUELA DE POSTGRADO

MSc. Ing. Edgar Darío Álvarez Cotí
Director
Escuela de Estudios de Postgrado
Facultad de Ingeniería

Guatemala, 28 de mayo de 2022

M.A. Edgar Darío Álvarez Cotí
Director
Escuela de Estudios de Postgrado
Presente

M.A. Ingeniero Álvarez Cotí:

Por este medio informo que he revisado y aprobado el **TRABAJO DE GRADUACIÓN** titulado: "IMPLEMENTACIÓN DE UN SISTEMA DE MONITOREO EN TIEMPO DE EJECUCIÓN DE APLICACIONES QUE FUNCIONAN EN CANALES ELECTRÓNICOS DE UNA ENTIDAD BANCARIA PARA DETECCIÓN DE FALLAS DE FORMA PREVENTIVA" del estudiante **Omar Francisco Mendoza Cajtic** quien se identifica con número de carné **200714554** del programa de **Maestría en Tecnologías de la Información y la Comunicación.**

Con base en la evaluación realizada hago constar que he evaluado la calidad, validez, pertinencia y coherencia de los resultados obtenidos en el trabajo presentado y según lo establecido en el *Normativo de Tesis y Trabajos de Graduación aprobado por Junta Directiva de la Facultad de Ingeniería Punto Sexto inciso 6.10 del Acta 04-2014 de sesión celebrada el 04 de febrero de 2014*. Por lo cual el trabajo evaluado cuenta con mi aprobación.

Agradeciendo su atención y deseándole éxitos en sus actividades profesionales me suscribo.

Atentamente,

MARLON ANTONIO PEREZ TURK
INGENIERO EN CIENCIAS Y SISTEMAS
COLEGIADO No. 4492

MA. Ing. Marlon Antonio Pérez Türk
Coordinador
Maestría en Tecnologías de la Información y la Comunicación
Escuela de Estudios de Postgrado

Guatemala, mayo de 2022

En mi calidad como Asesor del Ingeniero **Omar Francisco Mendoza Cajtic** quien se identifica con registro académico **200714554** procedo a dar el aval correspondiente para la aprobación del Trabajo de Graduación titulado: **"IMPLEMENTACIÓN DE UN SISTEMA DE MONITOREO EN TIEMPO DE EJECUCIÓN DE APLICACIONES QUE FUNCIONAN EN CANALES ELECTRÓNICOS DE UNA ENTIDAD BANCARIA PARA DETECCIÓN DE FALLAS DE FORMA PREVENTIVA"** quien se encuentra en el programa de **Maestría En Tecnologías De La Información y la Comunicación** en la Escuela de Estudios de Postgrado de la Facultad de Ingeniería de la Universidad de San Carlos de Guatemala.

Atentamente,

"Id y Enseñad a Todos"

MSc. Ing. Edwin Estuardo Zapeta Gómez
Colegiado No. 12767
Asesor

ACTO QUE DEDICO A:

Dios	Una luz que siempre ilumina mi camino y guía cada uno de mis pasos en esta vida, a él sea toda la gloria, el poder y la honra.
Mis padres	Nicolas Cajtic Ramos y Azucena Mendoza Centeno, por el amor y apoyo incondicional que siempre encuentro en ellos.
Mi esposa	Mariela Castillo, por su comprensión, amor y apoyo en todo momento.
Mi hija	Monserrath Mendoza Castillo, un angelito lleno de inocencia para quien deseo ser un ejemplo.

AGRADECIMIENTOS A:

Universidad de San Carlos de Guatemala	A la gloriosa Tricentenaria, por ser mi casa de estudios y permitir mi formación académica como profesional, de la cual estoy orgullosa de egresar.
Facultad de Ingeniería	Por todo el conocimiento y sabiduría adquiridos en sus aulas durante mis años de estudio.
Mis amigos	Personas que siempre me brindaron soporte en los momentos más difíciles de esta carrera.
Mi asesor	Msc. Ing. Estuardo Zapeta, por compartir su conocimiento y experiencia para la elaboración de este trabajo de investigación.

ÍNDICE GENERAL

ÍNDICE DE ILUSTRACIONES

FIGURAS

TABLAS

LISTA DE SÍMBOLOS

Símbolo	Significado
$	Dólares
%	Porcentaje
Q	Quetzales

GLOSARIO

Almacenamiento de logs	Consiste en el empleo de un medio de almacenamiento permanente (archivos, bases de datos, entre otros) para alojar logs generados y recolectados. Incluye la ejecución de tareas de rotación, archivado, comprensión, reducción, conversión, normalización y chequeo de integridad.
Análisis de logs	Se refiere al análisis del contenido de los logs con el objetivo de interpretarlo y obtener información relevante respecto a la administración de recursos, detección de intrusiones, resolución de problemas, análisis forense o auditorías dentro de la organización. Incluye la correlación de eventos, la visualización y exploración, y la generación de reportes.
Ciclo de vida de un log	Hace referencia a la serie de etapas de existencia del log en la organización desde su generación hasta su definitivo descarte y eliminación.
Eliminación de logs	Hace referencia a la eliminación de las entradas de logs almacenadas correspondientes a un criterio, que generalmente incluye una fecha y una hora determinadas.

Evento	Es una singular ocurrencia dentro de un ambiente, que involucra usualmente un intento de cambio de estado. Un evento incluye una noción de tiempo, la ocurrencia, y una descripción pertinente al evento o al ambiente que pueda ayudar a explicar o entender las causas o efectos del evento.
Fuente generadora de logs	Se refiere a todo dispositivo, sistema o aplicación que esté en la capacidad de registrar la ocurrencia de eventos.
Index	Es una colección de varios documentos o registros de log con una estructura similar que permite realizar consultas sobre sus campos.
Log	Es un registro de los eventos que ocurren dentro de los sistemas y redes de una organización.
Modelo de *Machine Learning*	Es la salida de información que se genera cuando se entrena el algoritmo de *Machine Learning* asociado a dicho modelo, con datos.
Runbook	En un sistema informático o una red, un runbook es una compilación de procedimientos y operaciones de rutina que realiza el administrador o el operador del sistema.
Transporte y recolección de	Incluyen los diversos mecanismos disponibles para mover los logs desde la fuente generadora

logs	hacia una ubicación diferente, generalmente centralizada.
Shard	Es una subdivisión de los *index* de Elasticsearch log compuesto por un pequeño segmento de consultas información asociado a este.

RESUMEN

La transformación digital y el aumento en la demanda de servicios en línea ha creado en las instituciones financieras la necesidad de planear una estrategia de omnicanalidad para mejorar la experiencia de los usuarios e impulsar mejores relaciones con los clientes a través de sus puntos de contacto especialmente en los canales electrónicos.

Esta demanda de servicios en línea ha obligado al sector financiero a reducir el tiempo comprendido desde que un producto o servicio es concebido hasta que está disponible para el usuario final. Esto implica que se deben optimizar los procesos en el desarrollo de la solución y eliminar aquellos que no aportan valor al objetivo de negocio que apoya el proyecto, con este fin se excluyen requisitos no funcionales durante la definición del proyecto para apresurar el inicio de su etapa de desarrollo.

Cuando la solución es implementada, estos requisitos que no se consideraron generan un comportamiento inesperado de las aplicaciones en producción. Estos fallos no son ocasionados por los equipos en los cuales se instala la solución, son provocados por el funcionamiento de la aplicación y esto no es detectado hasta que el fallo ha escalado de tal forma que ha afectado los recursos físicos del equipo donde se encuentra instalado y estos reportan una alerta de mal funcionamiento.

La solución propuesta en este trabajo de investigación tiene como objetivo general la implementación de un sistema de monitoreo en tiempo de ejecución de aplicaciones que funcionan en canales electrónicos de una entidad bancaria

para detección de fallas de forma preventiva utilizando algoritmos de Machine Learning.

Para este estudio de tipo cuantitativo, se desarrolló una investigación sobre generación y centralización de logs, almacenamientos de registros en base de datos no relacional y la definición de un modelo de Machine Learning con la capacidad de analizar un registro en tiempo real y predecir un tipo de falla y su severidad. El diseño de este estudio fue experimental, y para el entrenamiento del modelo se utilizó data histórica cuidadosamente seleccionada relacionada al tipo de alerta que se busca predecir.

Para cumplir el objetivo general se implementó la solución Elastic Stack para la centralización de logs de sistemas operativos y log de aplicaciones que almacenan información de cada ejecución, logrando con esto un análisis de todos los registros generados en log en tiempo real.

Para la detección de fallos de operación de forma preventiva, se definió un modelo de Machine Learning que identifica información relevante en cada uno de los registros de información arbitraria creada por las diferentes fuentes generadoras de log y es capaz de predecir un fallo y asociarle una severidad. En el tiempo que estuvo activo el experimento, el sistema predijo 93 fallas críticas, de las cuales 86 fueron pronosticadas de forma correcta, esto le da al sistema de monitoreo una efectividad del 86.02 % en la predicción de fallas.

El sistema también permite la visualización del estado general del sistema a través de un tablero que se alimenta de los registros generados por las aplicaciones y por el sistema operativo del servidor. La información de este tablero puede visualizarse en distintos periodos de tiempo para evaluar la eficacia del sistema a través del tiempo.

PLANTEAMIENTO DEL PROBLEMA Y FORMULACIÓN DE PREGUNTAS ORIENTADORAS

La transformación digital y el aumento en la demanda de servicios en línea ha creado en las instituciones financieras la necesidad de crear una estrategia de omnicanalidad para mejorar la experiencia de los usuarios e impulsar mejores relaciones con los clientes a través de sus puntos de contacto especialmente en los canales electrónicos, los cuales debido a las restricciones de movilidad impuestas por el Gobierno de Guatemala para disminuir los contagios de COVID-19, han incrementado su demanda de servicio.

Esta demanda de servicios en línea ha obligado al sector financiero a reducir el tiempo comprendido desde que un producto o servicio es concebido hasta que está disponible para el usuario final. Esto implica que se deben optimizar los procesos en el desarrollo de la solución y eliminar aquellos que no aportan valor al objetivo de negocio que apoya el proyecto, con este fin se excluyen requisitos no funcionales durante la definición del proyecto para apresurar el inicio de su etapa de desarrollo.

Cuando la solución es implementada, estos requisitos que no se consideraron generan un comportamiento inesperado de las aplicaciones en producción.

Estos fallos no son ocasionados por los equipos en los cuales se instala la solución, son provocados por el funcionamiento de la aplicación y esto no es detectado hasta que el fallo ha escalado de tal forma que ha afectado los recursos físicos del equipo donde se encuentra instalado y estos reportan una alerta de

mal funcionamiento. Este sistema de detección de fallas de forma reactiva permite que la aplicación siga funcionando aun cuando su comportamiento no es correcto y afecta otros sistemas con los cuales comparte recursos físicos.

La institución financiera objeto de estudio tiene operación en Guatemala y Honduras, con una red de más de 1000 agencias, más 4000 agentes bancarios y canales de atención electrónica como kiosco virtual, banca virtual, aplicación móvil y asistente virtual, y cuenta con su propio Departamento de Tecnología.

Cuando el área de mantenimiento y monitoreo de sistemas detecta una alerta, los administradores de los equipos físicos y virtuales inician una tarea de detección del origen de la falla enfocada en los aspectos técnicos que reportaron según sus protocolos de servicio y severidad de la alerta.

Debido a que las bitácoras únicamente guardan datos sobre los recursos físicos, solamente se puede determinar cuál fue el recurso afectado, por ejemplo, disco duro, memoria RAM, unidad de procesamiento, tráfico de red, pero no es posible determinar cuál fue el origen del problema y la aplicación que origino dicha falla, por lo que cuando esta se repita no será posible identificarla y únicamente se procederá a realizar el protocolo para detección de fallas reactiva documentado por la unidad de mantenimiento y monitoreo.

Si se revisa la documentación del proyecto con especificaciones de la solución no se define una sección de requisitos no funcionales que indique dominio de usuario y de datos que deberá soportar la aplicación para establecer límites óptimos de funcionamiento e instalar la aplicación en un equipo que soporte dichos límites.

Adicional, se determinó que existen fallas que son originadas por deficiencias en el código, y aunque existen herramientas de revisión de código estático como StyleCop *et al.,* (2010) que permiten aumentar su eficiencia, estas no son incluidas en el proceso de desarrollo de software por distintos motivos, principalmente porque muchas de las aplicaciones están ligadas al core bancario basado en Visual Basic 5 un lenguaje que no tienes soporte técnico desde hace muchos años.

Existen otros inconvenientes, por ejemplo, muchas funcionalidades son programadas en procedimientos almacenados en la base de datos, el repositorio central no se encuentra actualizado y no existe un set de pruebas automatizado porque el lenguaje origen no lo permite.

Es importante mencionar, que para las nuevas aplicaciones se tiene una estrategia para eliminar estos problemas porque están desarrolladas utilizando tecnología que si permite la integración de estas herramientas y minimizar las fallas en las aplicaciones originadas por deficiencias de código.

Debido a la transaccionalidad que manejan las operaciones bancarias siempre se deben hacer peticiones a la base de datos por lo que el fallo de una aplicación afecta directamente el rendimiento de esta a nivel general.

Esta degradación en el rendimiento de la base de datos genera que las peticiones no sean atendidas de forma correcta generando una saturación de conexiones que generalmente se resuelve utilizando un protocolo documentado del área de monitoreo que consiste en eliminar todas las peticiones activas, creando inconsistencia en los datos de las peticiones truncadas.

Debido a que las soluciones están instaladas en canales electrónicos también afectan el tráfico en la red interna de la institución, esto genera una degradación del servicio para otras aplicaciones generando retraso o pérdida de solicitudes.

Todos estos inconvenientes generan un incremento en las solicitudes en el centro de ayuda, tanto de primera línea que son las que ingresan los clientes, como las de segunda línea que se ingresan para soporte técnico por los agentes del centro de soporte.

Determinar de forma preventiva todos estos fallos disminuiría el número de quejas o solicitudes en centro de ayuda y determinar el origen de la falla es crucial para realizar una corrección que optimice el funcionamiento de la aplicación.

Cada software siempre tiene datos de registro, esto son esenciales para los desarrolladores y usuarios finales. Con los datos de registro se puede identificar la información de error del sistema, eventos de advertencia y si los procesos del sistema se están cargando con éxito.

Yang, Kristiani, Wang y Liu (2019), desarrollaron un sistema de monitoreo para los datos de registro para realizar un análisis más detallado del funcionamiento de su sistema utilizando el sistema de almacenamiento Ceph y la tecnología Elastic Stack.

Ceph proporciona una gran cantidad de almacenamiento escalable y ELK consta de tres componentes: Elasticsearch, Logstash y Kibana. Elasticsearch es un motor de búsqueda para todo tipo de documentos, Logstash se usa para administrar centralizar, transformar y guardar los datos en Elasticsearch y Kibana

se usa para visualizar datos de Elasticsearch, es una excelente herramienta para el análisis de datos en tiempo real y la toma de decisiones.

A partir de lo explicado anteriormente se plantean las siguientes preguntas.

Pregunta central de investigación:

- ¿Cómo implementar un control de fallos preventivo para las aplicaciones que funcionan en canales electrónicos de una entidad bancaria?

Preguntas auxiliares:

- ¿Cómo se puede utilizar Elastic Stack para monitoreo de funcionamiento de aplicaciones de una entidad bancaria?

- ¿Cómo identificar fallos de operación en una aplicación que funciona en canales electrónicos de una entidad bancaria por medio de algoritmos de Machine Learning?

- ¿Cómo verificar la eficacia del sistema de monitoreo para aplicaciones que funcionan en canales electrónicos de una entidad bancaria?

OBJETIVOS

General

Implementar un sistema de monitoreo en tiempo de ejecución de aplicaciones que funcionan en canales electrónicos de una entidad bancaria para detección de fallas de forma preventiva utilizando algoritmos de Machine Learning.

Específicos

1. Implementar Elastic Stack para la centralización de logs de sistemas operativos y log de aplicaciones que almacenan información de cada ejecución, para el análisis de datos del sistema en tiempo real.

2. Identificar y predecir fallos de operación en una aplicación que funciona en canales electrónicos de una entidad bancaria utilizando algoritmos de Machine Learning.

3. Realizar un tablero que muestre estadísticas iniciales de reportes de fallos que se actualice con los datos del sistema de monitoreo, para verificar la eficacia del sistema de monitoreo para aplicaciones que funcionan en canales electrónicos de una entidad bancaria.

MARCO METODOLÓGICO

- Tipo de estudio

Se realizó un estudio de tipo cuantitativo a fin de responder a la pregunta principal: ¿cómo implementar un control de fallos preventivo para las aplicaciones que funcionan en canales electrónicos de una entidad bancaria?, se desarrolló una investigación sobre generación, centralización y almacenamiento de eventos organizados en archivos de logs en una base de datos documental y la depuración de esta data recolectada con el propósito de entrenar algoritmos de Machine Learning que identifiquen patrones en los registros de log creados en tiempo de ejecución y predecir fallas del sistema, en tiempo real.

- Diseño

El diseño del estudio fue experimental, porque se utilizó la data recolectada y depurada de una cantidad de *logs* generados por el sistema, previo al experimento, como insumo para entrenar un algoritmo de Machine Learning y crear un modelo capaz de identificar patrones basado en fallos anteriores. A este modelo se le suministro como entrada un evento capturado en tiempo de ejecución y su salida fue un pronóstico basado en los datos que lo entrenaron, con la finalidad de detectar fallas de forma preventiva.

- Alcance

El alcance del estudio fue descriptivo, porque se describe un sistema de monitoreo de sistemas para detectar fallos de forma preventiva para las

aplicaciones que funcionan en canales electrónicos de una entidad bancaria y todos los componentes de este. Se describe el proceso para identificar fallos de forma preventiva para minimizar el impacto en rendimiento del sistema, beneficiando directamente a todos los usuarios de los sistemas de la entidad bancaria en la cual se implementará el sistema.

- Variables

A continuación, en la tabla I, se describen las variables.

Tabla I. **Variables**

Variables	Definición	Subvariables	Indicadores
Fallas detectadas de forma preventiva	Alertas de fallo detectadas por el sistema de monitoreo de forma preventiva.	Rendimiento	• Utilización de CPU • Memoria RAM en uso • Memoria RAM Disponible • Capacidad del disco duro • Espacio disponible en Disco. • Velocidad de lectura en disco • Velocidad de escritura en disco • Velocidad de datos enviados • Velocidad de datos recibidos
		Aplicación monitoreada	• Cantidad de aplicaciones monitoreadas.
		Archivos procesados	• Cantidad de archivos procesados • Cantidad de log procesados por aplicación
		Almacenamiento	• Velocidad de respuesta de consultas • Latencia de las consultas • Tasa de solicitudes
		Disponibilidad	• Nivel de servicio
		Alertas	• Número de alerta de riesgo bajo • Número de alertas de riesgo medio • Número de alertas de riesgo alto.

Fuente: elaboración propia.

- Fases del estudio

A continuación, se describen y se detallan las fases de la investigación.

- Revisión documental

Para entender las técnicas utilizadas para la detección de fallos es necesario, entender cuál es el comportamiento y origen de estas, con este propósito se realizó una revisión documental, previo a preparar el diseño e implementación de un sistema de monitoreo de aplicaciones que permita identificar fallas en tiempo real, contemplando la recopilación de información sobre los siguientes temas:

  - Principios de generación de logs
  - Fuentes generadoras de log
  - Conceptos básicos para detección de fallas
  - Correlación de datos
  - Herramientas de software libre para analizar logs
  - Procesamiento y transformación de información
  - Machine Learning para análisis avanzado de log

- Selección de la herramienta

El proceso de selección de la herramienta se ajustó a los requerimientos y presupuesto de la institución bancaria en la cual se implementó, evaluando los siguientes parámetros:

  - Costo total de propiedad
  - Tipo de licencia y soporte ofrecido

- Actualizaciones de la herramienta
- Características principales de la solución
- Documentación y capacitación de los usuarios
- Extensibilidad de la solución

- Diseño del sistema de monitoreo de aplicaciones en tiempo real

En la fase de diseño se incluyeron las siguientes tareas:

- Definición de una política

Se desarrolló una política de retención de los logs en las fuentes y en la central de almacenamiento considerando los siguientes parámetros:

- La existencia y requerimientos exigidos por estándares de cumplimiento adoptados por la institución bancaria donde se va a implementar la solución.

- El nivel de riesgo aceptado por la institución bancaria donde se va a implementar la solución con relación al periodo de tiempo en que se desee analizar la información contenida en los logs.

- Cantidad total de logs generados por las diversas fuentes, generalmente medida en bytes por unidad de tiempo.

- Mecanismos de almacenamiento y archivado de información disponibles en la institución bancaria donde se va a implementar la solución.

- Diseño de la arquitectura

Las características de la institución bancaria donde se implementó la solución se ajustan a un diseño descentralizado. Se implementó la solución ELK mediante un recolector con Logstash que filtra la información transmitida desde las fuentes de logs y almacena los resultados en una instancia de Elasticsearch. Las fuentes se clasifican en dos tipos: fuentes que poseen una instalación local de Logstash (Logstash-forwarder) para la transmisión y fuentes que utilizarán un agente de transporte diferente de Logstash (Logstash-forwarder)

- Construcción del modelo probabilístico

Se construyó un modelo probabilístico, basado en los datos almacenados previo al experimento para aprender el comportamiento normal, formando un patrón y cuando detectan valores que se desvían de este comportamiento, se informa como una anomalía. Se utiliza un sistema probabilístico frente a un sistema de reglas estáticas para reducir el número de falsos positivos.

- Desarrollo de la solución

    - Implementación de la arquitectura

        - Instalación del servidor Logstash.

            - ✓ Configuración de Logstash

        - Instalación del servidor Elasticsearch

            - ✓ Configuración de los índices de Elasticsearch

        - Instalación del servidor de Kibana

        - Configuración de las fuentes generadoras de logs

            - ✓ Configuración de Internet Information Server
            - ✓ Configuración de Microsoft SQL Server
            - ✓ Configuración de Windows Server
            - ✓ Configuración de la aplicación de gestión de tráfico ntopng.

    - Implementación del modelo probabilístico

    - Creación y configuración del panel de control

    - Pruebas integrales de funcionamiento del sistema

- Experimentación

En esta etapa se realizó la detección de anomalías en intervalos controlados de ejecución. El modelo aprendió cuál es el comportamiento normal, formando un patrón. Cuando existen valores que se desvían de este comportamiento, se informa como una anomalía. Esto se realizó construyendo un modelo probabilístico conforme el sistema va aprendiendo. La idea fue utilizar un sistema probabilístico frente a un sistema de reglas estáticas para reducir el número de falsos positivos.

- Análisis y monitoreo de eventos

El análisis y monitoreo de eventos es la etapa de consumo de la información obtenida mediante la gestión de las fuentes generadoras de *logs*. No obstante, existen múltiples herramientas que se pueden integrar con la solución elegida para evaluación, en el presente escenario se utilizaron:

  - Kibana, al formar parte de la solución ELK para la exploración, visualización y creación de cuadros de mando.

- Técnica de recolección de información

En siguientes incisos se describe la técnica de recolección de la información necesaria.

- Recolección de logs

La recolección de logs es el paso previo para la gestión centralizada en el que los registros son importados de diferentes fuentes a través de la infraestructura informática y se recopilan en una única ubicación central. Mediante el uso de ciertos cargadores de ficheros o datos, se envían los registros desde un origen como aplicaciones, contenedores, bases de datos o cualquier otro sistema y los registros se agregan y almacenan en una ubicación central. Para la recolección de logs que alimentaran el sistema de monitoreo en tiempo real, se toman en consideración los siguientes aspectos:

- Formato de logs

Se definió un formato de log a cada aplicación, según su naturaleza basando en los más extendidos:

  - Registros por línea con los datos separados por espacios, para *logs* que generan gran cantidad de registros en un archivo plano con el objetivo de optimizar tu tamaño en disco.

  - JSON y XML, para aplicaciones con intercambio de datos.

- Estándar

Según el entorno o el tipo de log que vaya a generarse, existen varios estándares para facilitar la intercomunicación de elementos.

  - Syslog

Se utilizó para la gestión del sistema y la auditoría de seguridad, mensajes generales de información, análisis y depuración. Permite la separación del *software* que genera mensajes, el sistema que los almacena y el software que los informa y analiza.

  - Common Log Format (CLF) y Extended Log Format (ELF)

Son formatos de archivo de texto estandarizado utilizado por servidores web para generar registros, la diferencia entre ambos es que ELF puede contener más información.

- Definición de la información consignada en logs

Los datos mínimos que se deben guardar para analizar y obtener información válida son los siguientes:

  - Fecha y hora, definir zona horaria
  - Tipo: error o alerta
  - Dirección IP del dispositivo origen
  - Código único del suceso
  - Mensaje con la descripción ampliada del suceso
  - Usuario del sistema o la aplicación
  - Contexto (fichero, página, parte de la aplicación que ha generado el registro o data ingresada en la petición)

- Centralización de datos

Se realizó un proceso de centralización de logs, en el cual los registros son importados de diferentes fuentes a través de la infraestructura ELK-Stack y se recopilan en una única ubicación central, en este caso Elasticsearch. Mediante el uso de ciertos Logstash, se envían los registros desde un origen como aplicaciones, contenedores, bases de datos o cualquier otro sistema y los registros se agregan y almacenan en Elasticsearch.

- Log indexer

Es una técnica de procesamiento de texto para analizar los registros en cada y transformarlos en datos que puedan ser almacenados e indexados para su posterior análisis.

INTRODUCCION

La pandemia que afecta el mundo desde 2020 ha sido catalogada por muchos expertos del área de informática y ciencias de la computación como el principal impulsor de un proceso de transformación digital para las organizaciones del mundo, y quienes se adaptaron a este cambio están experimentando una nueva forma de prestación de servicios por medio de canales digitales. Las empresas que iniciaron de forma acelerada la migración de sus procesos presenciales hacia los canales virtuales están enfrentando algunos problemas derivados de la falta de planificación en la implementación de sus soluciones y el rápido escalamiento en el uso, por parte de los clientes, quienes cada día están más anuentes a usar canales digitales para evitar el contagio de COVID-19.

Entre las organizaciones que actualizaron su modelo de atención al cliente se encuentran las instituciones bancarias, un sector en el cual la interacción entre personas está muy arraigada en la cultura pero que ha tenido que cambiar y ahora se está dando un impulso al uso de los canales electrónicos para proteger la salud de los clientes y trabajadores. Este aumento en la demanda de servicios ha generado que los sistemas sean sometidos a cargas transaccionales muy altas que no estaban proyectadas y, en consecuencia, los sistemas fallan.

Existen fallos de bajo impacto que el área de monitoreo, del departamento de mantenimiento de software, puede controlar y dar una solución rápida, pero existen otros más críticos que requieren mucho tiempo de investigación antes de encontrar una solución. Este estudio implementó una solución a esta problemática, automatizando el monitoreo de aplicaciones que funcionan en

canales electrónicos de una entidad bancaria para detectar los fallos de forma preventiva y mejorar la estabilidad de los canales electrónicos.

El tema de estudio fue la automatización de la detección de fallos a través de la gestión centralizada de logs, para obtener y consolidar datos del funcionamiento de las aplicaciones y sistemas que conforman el entorno para el funcionamiento de los canales electrónicos. Estos datos ahora son importados a través de la infraestructura informática de la institución bancaria hacia un almacén de datos en una ubicación central, con una estructura común y optimizada para su posterior lectura y análisis en condiciones adecuadas de rapidez y eficacia.

Se desarrolló un modelo de Machine Learning el cual busca e identifica patrones entre los registros de eventos ubicados en una base de datos central, que estará siendo actualizada en tiempo real dando a este modelo la capacidad de detectar y predecir fallos en tiempo real de ejecución. Este sistema de monitoreo y notificación de alertas es una herramienta para que el área de mantenimiento de aplicaciones pueda realizar las tareas necesarias que garanticen el correcto funcionamiento del sistema en un momento oportuno.

Para la implementación de este sistema se desarrolló una arquitectura basada en plataformas de código abierto para cubrir las funciones de recuperación de logs, procesamiento de datos, transformación de mensajes, almacenamiento, búsqueda, y análisis de data. El estudio contempló una fase de experimentación para entrenar el modelo predictivo usando Machine Learning y que este sea capaz de dar los resultados esperados en la detección preventiva de fallas.

Para cubrir los aspectos técnicos, operativos y económicos de factibilidad del estudio se realizó en colaboración con una institución bancaria que estuvo dispuesta a ofrecer su apoyo para la implementación exitosa del proyecto.

El trabajo de graduación constara de 6 capítulos, a continuación, una breve reseña de cada uno:

- Antecedentes: aspectos que cambiaron la forma de usar los canales digitales y el impacto derivado de alta demanda de servicios a través de estos.

- Justificación: porque es importante el monitoreo de sistemas y determinar la calidad que se está ofreciendo al cliente.

- Alcances: definición de los alcances técnicos, investigativos y los resultados de este estudio.

- Marco teórico: una investigación documental sobre la gestión centralizada de logs y las herramientas que contribuyeron a obtener resultados satisfactorios para este estudio.

- Presentación de resultados: esquema de los resultados obtenidos durante la fase de experimentación del estudio.

- Discusión de resultados: análisis de los resultados y sus implicaciones para el problema que se planteó.

1. ANTECEDENTES

La transformación digital impacto directamente en la vida de todas las personas, cambiando cómo se hacían las cosas antes. Este cambio no solo afecto a los usuarios de bienes y servicios, también afecto a las instituciones que brindan dichos bienes y servicios. Debido a estos cambios se adaptaron para hacer las cosas más rápidas y por medio de canales electrónicos. Pero, aunque la tecnología y el internet hicieron los procesos más rápidos y accesibles, existen problemas con los cuales las instituciones de distintos sectores deben enfrentar en sus operaciones diarias, como fallos en los sistemas o comportamientos anómalos de las aplicaciones, que impactan directamente en su servicio al usuario final.

Durante la operación del observatorio ALMA; Gil, Reveco y Shen (2016), diariamente se generan una gran cantidad de archivos de registro. Como el software ALMA todavía está en continua evolución, los registros no solo son útiles para diagnosticar fallas detectadas durante operación, pero también son necesarios para representar el análisis de rendimiento a largo plazo y proporciona una vista rápida del comportamiento sistema. Como cualquier otro software, el propósito del log es publicar cualquier tipo de estado e información de diagnóstico. Los logs son esenciales para el análisis *postmortem* de problemas de hardware o software, especialmente para un complejo sistema distribuido como el software de control ALMA.

Los fallos eran reportados como *tickets* y la resolución de estos implica investigaciones se llevan a cabo principalmente a través de la revisión de logs que proporciona información sobre el estado de los equipos y servidores de red

que podrían estar relacionados con el error informado. Dependiendo de la dificultad del problema y el nivel de experiencia del investigador en el área específica, una investigación podría demorar de 10 minutos a un par de días en ser resuelta.

Algunos inconvenientes derivados de esta metodología son la duplicidad de investigaciones que se realizadas para corregir un mismo problema, una alta generación de *tickets* porque se reporta el mismo fallo varias veces, dificultad para analizar toda la información almacenada en los logs debido a su gran tamaño y una falta de estadísticas de sobre la resolución de fallos a través del tiempo.

Para solucionar estos problemas optaron por la combinación de herramientas conocida como ELK-Stack: Elasticsearch como base de datos de back-end, Logstash como pipeline y formateador de registros, y Kibana como herramienta de visualización.

Lograron disminuir las investigaciones duplicadas al identificar el origen de fallos, esto también contribuyo a bajar el número de *tickets* reportados por el sistema. Usando ELK-Stack se proporciona una forma rápida y versátil de analizar toda información generada en *logs* lo que ayudo a bajar el tiempo invertido por los investigadores en encontrar el origen de fallos reportados. Usando los tableros generados por Kibana se implementó un control con estadísticas sobre la resolución de fallos a través del tiempo.

ELK-Stack como sistema de gestión de logs es muy útil en términos de usabilidad debido a que permite fácilmente realizar búsquedas sobre grandes cantidades de información y visualizar los resultados de forma ordenada. Una aplicación de esto puede ser la geo identificación de usuarios, Prakash, Kakkar y

Patel (2017) que acceden un sitio basándose en los *logs* de ingreso. Ellos han utilizado un pequeño conjunto de datos de registro sólo para demostrar la usabilidad de ELK-Stack para obtener información de un sistema y sus usuarios a través del análisis de información almacenada en los *logs*.

Otra implementación de ELK-Stack fue realizada en Brasil, Almeida *et. al.* (2017) con el objetivo de realizar observación meteorológica, utilizando para indexar datos y metadatos provenientes de los *logs* generados por casa uno de los sensores. Además de permitir la aplicación de la curva de calibración del sensor para corregir los datos medidos, el uso de dicha infraestructura también permitió el análisis de datos. Este caso es interesante porque nos muestra cómo se puede lograr una homologación de varios *logs* con formatos e información distinta y luego poder analizarla como un conjunto único de información consolidada.

A medida que se ha incrementado el uso de sistemas de administración de log, la industria del software ha adoptado una serie de criterios, Mitra y Sy (2016), sobre buenas prácticas para un correcto uso de LMS, entre ellos están:

- Definir una política de auditoría: se debe definir una amplia gama de tipos de eventos de seguridad que se pueden grabar en los registros.

- Consolidación de registros: permitir almacenar *logs* como registros de base de datos y archivo de texto plano comprimido para reducir el espacio de almacenamiento y permitir reimportar log de mucho tiempo atrás.

- Monitoreo de eventos: utilizar herramientas de monitoreo de log adicionales a la utilidad Windows Event Log.

- Generación de informes: los informes se deben proporcionar diaria, semanal, mensual y anualmente, y se debe tener la capacidad de generar reportes personalizados según la necesidad del cliente.

- Auditoria: la función de un LMS es convertir datos no estructurados en un conjunto de información útil que permite ser analizada e interpretada a través de búsquedas realizadas por parámetros.

Esta serie de buenas prácticas permiten que un sistema de monitoreo además de detectar fallas y comportamiento anómalo de aplicaciones también sea capaz de brindar una vista para evaluar el desempeño del sistema. La evaluación se realiza para verificar el rendimiento y el objetivo principal de la evaluación es asegurar que los componentes del sistema estén funcionando de forma correcta. Rochim, Aziz y Fauzi (2019)

Los tableros con gráficas obtenidos de la ingesta y procesamiento de log son herramientas útiles que permiten visualizar como se está comportando el sistema y realizar una evaluación del rendimiento con mayor detalle y precisión, sobre todo en tiempo real y con datos de un ambiente de real.

Existen ciertos modelos de tableros que por la información que muestran siempre son implementados en un LMS. Prakash *et al.,* (2017), con algunos cambios según el giro de negocios de la institución. Las principales funciones de estos tableros son la visualización de *logs* recibidos por cada sistema, visualización por severidad, visualización de alertas de funcionamiento y visualización de estadísticas de rendimiento del sistema.

En conclusión, como se muestra a través de distintas implementaciones, ELK-Stack puede funcionar como una excelente herramienta para la gestión de

log, permitiendo consolidar información de distintas fuentes para que pueda ser analizada. Con esta herramienta se puede procesar toda la información generada en los logs de aplicaciones que funcionan en canales electrónicos de una institución bancaria para realizar un análisis de los fallos reportados y determinar su origen. Debido a que la información se almacena en una base de datos, también se pueden generar los tableros que se consideren necesarios para mostrar el funcionamiento del sistema en tiempo real.

2. JUSTIFICACIÓN

La línea de investigación acerca de la cual se elaboró este trabajo es desarrollo de sistemas para complementar la tecnología móvil.

Las medidas de distanciamiento social requeridas para la contención de la pandemia generada por el COVID-19 y las restricciones de movilidad impuestas por el Gobierno de Guatemala han generado en el país un incremento en la demanda de servicios electrónicos.

Empresas que contaban con este tipo de canales como una alternativa para sus usuarios se han visto afectadas por este incremento en el tráfico de solicitudes. Las entidades bancarias son un sector que ahora se enfrentan a este problema debido a que los productos y servicios ofrecidos en sus canales electrónicos era solo una parte de su portafolio y la otra parte era administrada en sus agencias por el modo tradicional de persona a persona.

Para adaptarse a esta nueva normalidad y mantener sus operaciones con regularidad se implementaron productos y servicios a través de sus canales electrónicos con la mayor premura posible que minimizara el impacto financiero que implicaba la denegación de servicios al no poder operar desde sus agencias físicas.

Estos servicios presentan fallos derivados del corto tiempo de desarrollo que impactan directamente en la calidad del servicio al cliente, con la presente investigación se pretende implementar una forma de identificar estos fallos estableciendo un sistema de monitoreo en tiempo de ejecución de aplicaciones

que funcionan en canales electrónicos de una entidad bancaria para detección de fallas de forma preventiva.

Elastic Stack es un grupo de productos de código abierto de Elastic diseñado para ayudar a los usuarios a tomar datos de cualquier tipo de fuente y en cualquier formato y buscar, analizar y visualizar esos datos en tiempo real.

La implementación de Elastic Stack para la centralización de log de sistemas y aplicaciones que funcionan en canales electrónicos de una entidad bancaria, proporciono una herramienta para realizar un análisis detallado del funcionamiento de dichas aplicaciones lo que permite detectar fallos de forma preventiva mejorando el rendimiento de las aplicaciones impactado directamente y de forma positiva en el servicio entregado al cliente final.

3. ALCANCES

3.1. Resultados

Sistema de monitoreo en tiempo de ejecución, con al menos una aplicación configurada, con las siguientes funcionalidades:

- Generación e ingesta de logs hacia la base de datos Elasticsearch.
- Almacenamiento persistente y organizado de la información de los logs.
- Algoritmos de análisis predictivo para detección de fallos de forma preventiva.
- Tableros de visualización del estado del sistema en tiempo real.

3.2. Técnicos

- Diseño e implementación del esquema de registro para los datos que deben almacenarse en los logs de cada aplicación que forme parte del sistema de monitoreo.
- Diseño e implementación del modelo de datos en el cual se almacenará la información en la base de datos Elasticsearch.
- Diseño e implementación de los procesos de análisis sobre la data almacenada para la predicción de fallos en tiempo real utilizando técnicas

de procesamiento de logs (filtro de logs, análisis estadístico de eventos, correlación de logs y minería de datos)

- Definición los eventos normales o esperados en un entorno particular y controlado, de modo que las alertas se realicen solo en patrones y eventos asociados a una falla critica.

- Diseño e implementación la arquitectura para la instalación de los componentes de ELK-Stack.

- Diseño e implementación de los tableros en Kibana para visualizar el estado del sistema en tiempo real.

3.3. Investigativos

- Identificación de la relación entre número de incidentes reportados a la mesa de ayuda y el número de fallos identificados de forma predictiva por el sistema de monitoreo en tiempo real.

- Descripción del proceso y la metodología para análisis predictivo de fallos utilizando la data de log almacenada.

- Registro de eventos del sistema de monitoreo para aplicaciones instaladas en canales electrónicos de una institución bancaria.

- Demostración de la efectividad para la detección preventiva de fallos utilizando un sistema de monitoreo en tiempo real.

4. MARCO TEÓRICO

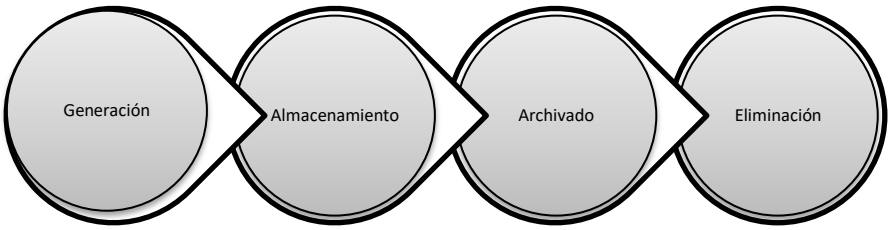
4.1. Gestión de logs

En entornos que trabajan con tecnología como una herramienta en su giro de negocios existen muchas aplicaciones que funcionan como fuentes de generación de logs, usualmente cada log posee un formato distinto y se almacena en distintos sitios, según la aplicación que lo genere. Para obtener información de valor para el negocio de esta gran cantidad de datos sin estructura, es necesario recopilar, filtrar y estructurar cada uno de los mensajes de las diversas fuentes y almacenarlos de forma centralizada, esto permite acceder fácilmente a ellos para su revisión o archivado para fines de auditoría. Para realizar la implementación de un sistema de gestión de log es necesario conocer los principios básicos descritos a continuación. (West-Brown, *et al.,* 1998)

4.1.1. Fases de un archivo de registros

Un log es un registro de eventos generados por un sistema y posee una serie de fases descritas en la figura 4. El registro se inicia en la ejecución de una aplicación, y se almacena durante el tiempo que esta esté operando, en el mismo sitio de almacenamiento. Cuando la ejecución termina, los logs son archivados en sitios distintos y se mantienen ahí por un periodo de tiempo, previo a ser eliminados. (Chuvakin y Schmidt 2012)

Figura 1. **Fases de un archivo de registros**



Fuente: elaboración propia.

4.1.1.1. Generación de logs

Si se administra un entorno con varios sistemas digitales, la mayoría de ellos pueden ser configurados para crear archivos de registros de eventos. (Kent y Souppaya, 2006) Estos archivos pueden ser divididos según su naturaleza:

- Seguridad

Se usan principalmente para detectar y responder a ataques, infecciones de códigos maliciosos, suplantación de identidad y otros incidentes de seguridad.

- Operaciones

Se crean para guardar información de eventos en la ejecución de tareas y procesos del sistema.

- Depuración de aplicaciones

Los programadores utilizan este tipo específico de registro en el entorno de desarrollo. No se implementan en producción debido a impactos en el rendimiento, pero se pueden activar por medio de banderas para determinadas aplicaciones si fuera necesario.

Según Chuvakin y Schmidt (2012) existen tres componentes que se deben tomar en cuenta, durante el proceso de generación de registros de eventos:

- Transporte

Establece la forma en la cual se realizará el movimiento de archivos entre distintas ubicaciones. Por ejemplos, usando protocolos UDP, TCP, SOAP sobre HTTP y SNMP.

- Sintaxis

Establece la estructura interna del archivo de registros. Cómo se forma el registro, cómo guardarlo y cómo lo visualiza el usuario para su análisis. Existen algunas estructuras usadas dentro de la industria del *software* que por su eficacia se han consolidado como estándar, tal es el caso de Syslog, Apache o el utilizado por CISCO.

- Formato del log

En este se establece cual es tipo de información que se almacenara. Se debe consignar data que genere un valor agregado para cada aplicación y por esto no hay un estándar.

También se debe considerar que existen muchos tipos de eventos que pueden ser almacenados y estos pueden ser categorizados Chuvakin y Schmidt (2012), aunque no es un estándar, como lo muestra la tabla II:

Tabla II. **Operaciones registradas en los logs**

Tipo de evento	Descripción del tipo de evento
Cambios administrativos en el sistema	Guarda cambios en el sistema, componentes, cuentas de usuario y actualizaciones.
Autenticación y autorización	Documente las decisiones e intentos de autenticación y autorización para usuarios.
Administración de permisos	Eventos relacionados a la política de roles y autorizaciones por usuario.
Gestión de amenazas.	Eventos de vulnerabilidad de la seguridad de sistemas e información.
Gestión recursos	Registro de eventos relacionados al rendimiento y estado de los recursos físicos de los sistemas.
Disponibilidad y continuidad del negocio	Archivo del estado de un sistema en un momento determinado.
Errores y fallas	Comportamiento no esperado del sistema o aplicación
Tramas de depuración	Mensajes asociados a peticiones que se desean monitorear.

Fuente: elaboración propia.

4.1.2. Conservación de archivos de registro

Dado que el análisis de eventos requiere la disponibilidad de todos los datos recolectados, usualmente los archivos se mueven desde la fuente que los genera a almacenes de datos con gran capacidad de espacio en disco. Este espacio en disco suele consumir recursos económicos, por tanto, se debe establecer una política asociada a la seguridad de la información Chuvakin y Schmidt (2012), del tiempo que deben permanecer en custodia y luego ser eliminados, en tabla III se muestran algunos aspectos importantes por considerar:

Tabla III. **Conservación de archivos de registro**

Requerimiento	Descripción del requerimiento
Estándares	Existen estándares internaciones como PCI con directrices según el tipo de institución.
Administración del riesgo	Según la data que se recolecte y el tipo de datos, el área de riesgos debe definir cuándo puede eliminarse un archivo.
Espacio	Se debe hacer una proyección del espacio que ocuparan los archivos a futuro, y si se cuenta con ese espacio.
Tipo de almacenamiento	Se debe considerar si la organización tiene la capacidad de costear las unidades de almacenamiento físico y si estas satisfacen las necesidades del sistema de gestión de archivos de logs. Puede tener en cuenta una opción de almacenamiento en la nube, con costos de mantenimiento menor a una instalación propia.

Fuente: elaboración propia.

4.1.3. Exploración de data en los archivos de registro

El análisis de eventos requiere que la data esté concentrada en un solo lugar para una exploración unificada, que pueda convertir data fragmentada en información de valor para la organización. Para este proceso se usan técnicas de análisis de información Kent y Souppaya (2006), descrita en la tabla III, acerca de toda la data recolectada de los archivos generados en sus fuentes primarias.

Tabla IV. **Técnicas de análisis de información**

Técnica	Descripción
Selección por tipo	Se establece un criterio bajo el cual se agrupan ciertos tipos de archivos de eventos.
Estadística	Se aplica estadística descriptiva sobre toda la data con el objetivo de encontrar parámetros para analizar su comportamiento.
Correlación de *logs*	Se establece un criterio para relacionar eventos, de distintas fuentes y tipos, pero que están relacionados.
Minería de datos	Se crea un modelo para búsqueda de patrones de eventos en la data recolectada.

Fuente: elaboración propia.

La detección de fallas es solo un paso de la gestión de archivos de eventos, para generar valor se debe enlazar a un sistema de monitoreo y alertas, que muestre información útil del estado del sistema. Para esto se utilizará el componente Kibana, del ELK-Stack.

4.1.4. Fuentes

Cada uno de los sistemas que conforman un ecosistema basado en tecnología son considerados fuentes de generación de archivos de eventos Chuvakin y Schmidt (2012), para clasificar estas fuentes existen 2 criterios: tipo de mecanismo y tipo de sistema. Esta tarea está a cargo del componente Logstash, de ELK Stack.

4.1.4.1. Tipo de sistema

Según el tipo de sistema, una fuente de generación de log puede ser de 3 tipos Kent y Souppaya (2006), descritos en la tabla V:

Tabla V. **Clasificación de fuente por tipo de sistema**

Fuente generadora	Información generada
Dispositivos de seguridad y de red	Inicio y finalización de sesión.
	Datos de conexión a un servidor de la red
	Data de transferencia de información.
	Reinicio de sistema.
	Cambios de configuración.
Sistemas operativos	Autenticación.
	Inicio, apagado y reinicio del sistema.
	Inicio, apagado y reinicio de servicios.
	Falla de servicios
	Estatus de sistema
Aplicaciones	bitácora
	Bitácora de usuarios con alto nivel de acceso.
	Actividades críticas.
	Reconfiguraciones.

Fuente: elaboración propia.

4.1.4.2. Tipo de mecanismo

Chuvakin y Schmidt (2012) En la clasificación por mecanismo existen 2 categorías:

- *Push-based*- Envío de archivos al servidor central

En esta categoría están todas las fuentes que utilizan Eventos Windows, SNMP o Syslog.

- *Pull-based*-Recolección de archivos por el servidor central

Esta categoría reúne a todas las aplicaciones desarrolladas para ir directamente donde se genera el log, y recolectar la data para enviarla a una base de datos centralizada por medio de agentes.

4.2. ELK Stack

ELK Stack, es un grupo de productos de código abierto de Elastic, diseñado para tomar datos de cualquier tipo de fuente y formato y buscar, analizar y visualizar esos datos en tiempo real (Sachdeva, 2017). Implementar ELK Stack para la centralización de log de sistemas y aplicaciones daría una vista del funcionamiento de las aplicaciones en tiempo real, que se incluyan en este sistema.

Gil, *et. al.* (2016) Implementaciones previas de ELK Stack han demostrado que este *software* de administración de log permite identificar fallos de aplicaciones a través del procesamiento de log, además de ayudar en la

investigación de la resolución de estos por medio del análisis de grandes cantidades de información de distintas fuentes de origen.

4.2.1. Componentes del ELK Stack

Chhajed (2015) expresa que la plataforma ELK es una solución completa de gestión centralizada de *logs*, construida sobre una combinación de tres componentes de código abierto: Elasticsearch, Logstash y Kibana. El componente central de ELK Stack es Elasticsearch, un motor de análisis y búsqueda de código abierto distribuido. Está basado en Apache Lucene y está diseñado para escalabilidad horizontal. Logstash es un canal de recopilación, enriquecimiento y transporte de datos.

La capacidad de integrar conectores con una infraestructura común le da a Logstash la capacidad de procesar múltiples tipos de registros, eventos y fuentes de datos no estructurados para su distribución en una variedad de resultados, incluido Elasticsearch. ELK se completa con Kibana, que es una plataforma de visualización de datos que permite la interacción con los datos a través de gráficos. Kibana puede dar una visión de los datos con paneles que aprovechan una variedad de visualizaciones disponibles. A continuación, se hace una descripción más detallada de cada componente:

4.2.1.1. Logstash

Turnbull (2019) Es el motor central de flujo de datos de ELK Stack para recopilar, enriquecer y unificar todos sus datos sin importar el formato o el esquema. El procesamiento en tiempo real es especialmente poderoso cuando se combina con Elasticsearch, Kibana y Beats.

Logstash es esencialmente un marco integrado para la recopilación, centralización, análisis, almacenamiento y búsqueda. Es un *software* de código abierto que puede unificar dinámicamente datos de fuentes dispares y normalizar los datos según el destino, en este caso Elasticsearch. Permite transformar cualquier tipo de evento con una amplia matriz de complementos de entrada, filtro y salida, con muchos códecs nativos que simplifican aún más el proceso de ingestión de data. Existe una amplia gama de filtros que se pueden aplicados a los registros recopilados para transformar los eventos.

Logstash tiene una arquitectura extensible y existen varios complementos para los desarrolladores. Ofrece información casi en tiempo real inmediatamente en el índice o el tiempo de salida. Enviar estos registros de Logstash a Elasticsearch permite realizar una amplia gama de mapeos, agregaciones y búsquedas.

4.2.1.2. Kibana

Como lo describe Azarmi (2017) Es un componente del ELK-Stack para visualización y análisis diseñado para interactuar con la base de datos Elasticsearch. Realiza análisis de datos avanzados y visualización de información en varios tipos de gráficos, como tablas o mapas. Se puede utilizar para buscar, ver e interactuar data estructurada o no estructurada, almacenada en Elasticsearch.

Interpretar grandes volúmenes de datos es bastante intuitivo con Kibana mediante la sencilla interfaz para desarrollar y administrar tableros con información, que pueden mostrar cambios en tiempo real gracias a la alta velocidad de respuesta de Elasticsearch.

Según Prakash *et al.,* (2017) existen algunos modelos de tableros que por la información que muestran sueles ser implementados para un LMS, con algunos cambios según el giro de negocios de la institución que lo utiliza.

- *Syslog evaluation*

Los *logs* generados por cada agente son enviados para ser ingestados y procesador por *Elasticsearch*. Este tablero debe mostrar cada uno de los logs recibidos exitosamente y la información relevante en ellos.

- *Log severity evaluation*

Según la información de cada log, este debe ser agrupado por una criticidad. Este tablero debe mostrar los grupos de log según su criticidad.

- *Home dashboard evaluation*

Muestra información general sobre aplicaciones monitoreadas. El panel de inicio consta de series de tiempo de evaluación, unidades de mayor impacto, indicador de gravedad, visualización de gravedad.

- *Application evaluation*

La evaluación de la aplicación explica los resultados y apariencia de la aplicación que se ha creado.

4.2.1.3. Elasticsearch

Dixit, Rogozin'ski, Kuc, y Chhajed (2017) es un motor de análisis, búsqueda y almacenamiento distribuido en tiempo real. Se puede usar para muchos propósitos, pero un contexto en el que sobresale es la indexación de flujos de datos semiestructurados, como registros almacenados en un log.

Proporciona un motor de búsqueda multihilo, con capacidad de búsqueda por texto completo y un API que utiliza documentos JSON. Se puede utilizar para búsquedas de texto completo, búsquedas estructuradas, analítica, o una combinación de los tres.

Una de sus características clave es la capacidad de buscar rápidamente indexando el texto que se buscará. Puede realizar búsquedas de texto completo, manejar sinónimos y calificar documentos por relevancia. Además, también puede generar análisis y agregación a partir de los mismos datos, en tiempo real.

5. PRESENTACION DE RESULTADOS

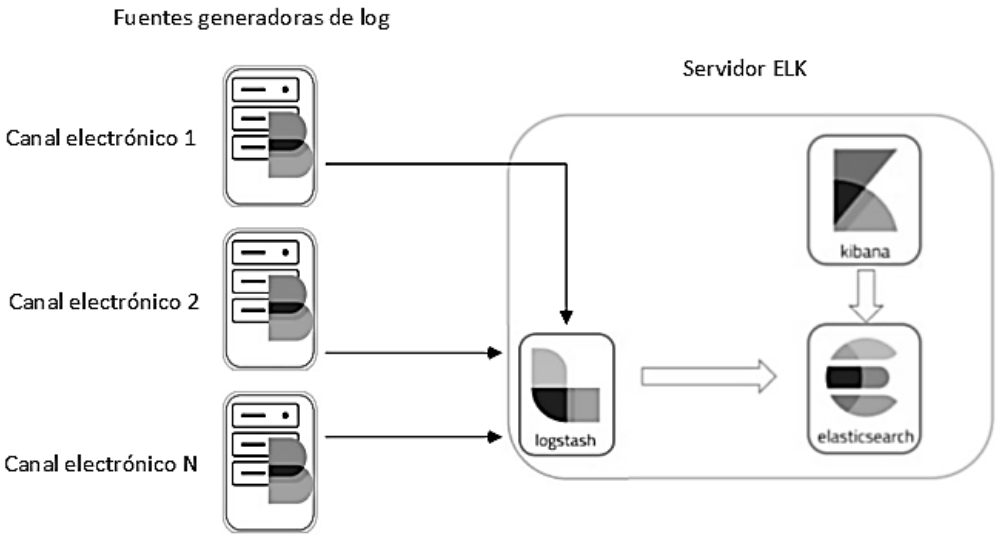
La pregunta principal de este trabajo de investigación fue ¿Cómo implementar un control de fallos preventivo para las aplicaciones que funcionan en canales electrónicos de una entidad bancaria?, en respuesta a esta interrogante se diseñó e implementó un sistema de monitoreo de aplicaciones en tiempo de ejecución con la capacidad de detectar fallas de forma preventiva.

5.1. Elastic Stack para monitoreo de funcionamiento de aplicaciones

El monitoreo de funcionamiento de aplicaciones se realizó implementando Elastic Stack para la centralización de logs de sistemas operativos y log de aplicaciones que almacenan información de cada ejecución, para el análisis de datos del sistema en tiempo real.

Como se muestra en la figura 2, cada una de las fuentes generadoras del log, identificadas como Canal electrónico N, siendo N el número correspondiente al orden en cual se agregó cada canal al sistema de monitoreo, crean archivos con registros de eventos tanto de aplicaciones como de sistema operativo, estos eventos son capturados y filtrados por un pipeline en Logstash, y por cada evento se inserta un registro indexado una base de datos de Elasticsearch. Estos registros, con diferentes orígenes y formatos, pueden ser analizados y visualizados utilizando Kibana, tanto en tiempo real como en datos históricos.

Figura 2. **Arquitectura implementada**



Fuente: elaboración propia, utilizando MS Visio.

Los componentes correspondientes al servidor ELK se instalaron en un servidor con sistema operativo Windows 10 Pro for Workstations con Java SE Development Kit 18.0.1.1, necesario para su correcta ejecución. En la tabla VI se describe la versión de cada uno de los componentes.

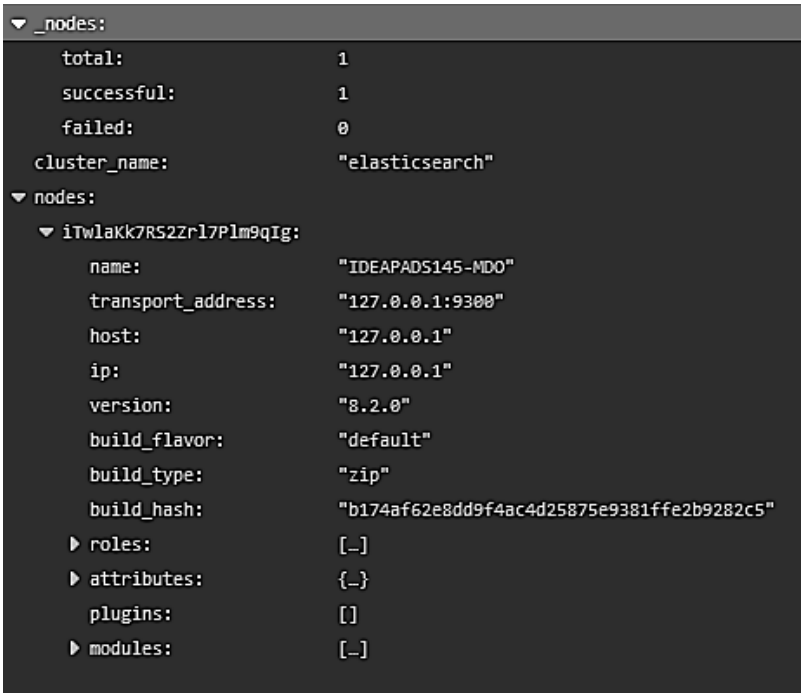
Tabla VI. **Versión de los componentes ELK**

Componente	Versión
Elasticsearch	8.2.0
Kibana	8.2.0
Logstash	8.2.0
Java SE Development Kit	18.0.1.1
SO Windows 10 Pro for Workstations	21H2 Build 19044.1682

Fuente: elaboración propia.

El esquema de la base de datos se definió con un clúster y contenido en este clúster, se configuro un nodo. El nombre del clúster es “*elasticsearch*” y el identificador del nodo es “*IDEAPADS145-MDO*”, las características del nodo activo se describen en la figura 3.

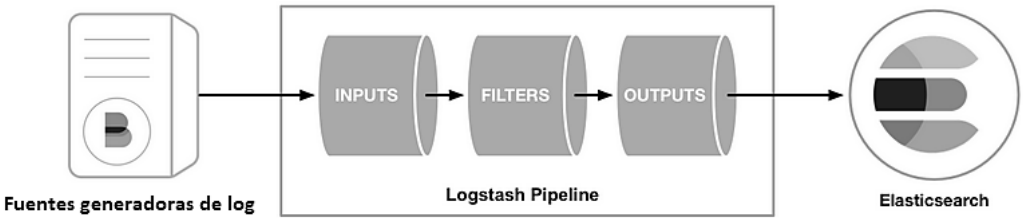
Figura 3. **Configuración del clúster**



Fuente: elaboración propia, utilizando MS Snipping Tool.

La ingesta de datos se realiza por medio de un pipeline en Logstash, utilizando la configuración descrita en la figura 4, mediante el cual se capturan los eventos generados por las fuentes generadoras de log y se examinan utilizando los filtros *dissect,* que analiza registros de acuerdo con delimitadores y *grok*, que funciona de acuerdo con la coincidencia de expresiones regulares para entradas que no cuentan con la misma cantidad de columnas en todos los registros.

Figura 4. **Configuración del pipeline**



Fuente: elaboración propia, utilizando MS Visio.

Estos registros, son indexados y almacenados en la base datos Elasticsearch, creando un solo depósito de información que puede ser fácilmente consultada bajo distintos criterios, como ilustra en la figura 5.

Figura 5. **Criterios de búsqueda**

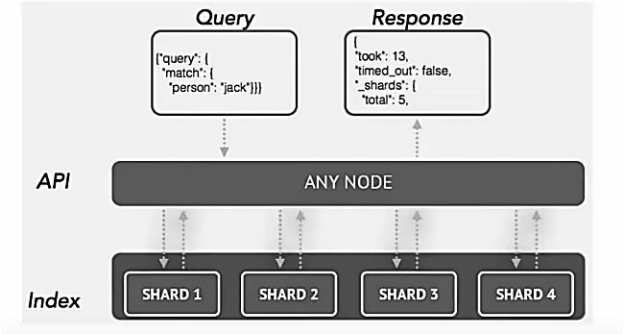
Fuente: elaboración propia, utilizando MS Snipping Tool.

En la figura 6 se puede visualizar la forma en que Elasticsearch ejecuta una consulta sobre la información, utilizando una agrupación lógica de registros

denominada *index* la cual a su vez se enlace con uno más *shard* los que pueden tener asociados una o más replicas, este es el mecanismo que usa Elasticsearch utiliza para distribuir datos alrededor del clúster.

Para este estudio se utilizó la configuración predeterminada de 5 *shard* por *index*. Esta división por capas permite que las consultas realizadas a la base de datos puedan realizarse con pocos criterios de búsqueda y obtener resultados precisos y rápidos. El API de Elasticsearch es accesible desde métodos REST y la información se retorna en formato JSON.

Figura 6. **Ejecución de una consulta**



Fuente: elaboración propia, utilizando MS Visio.

La visualización de eventos en Kibana se muestra en el tablero de la figura 7, con el pipeline recibiendo datos en *streaming* o tiempo real por distintas fuentes generadoras de log. El intervalo de consulta de estos eventos también puede configurarse para visualizar registros creados en un rango de tiempo específico.

Figura 7. **Visualización de eventos**

Fuente: elaboración propia, utilizando MS Snipping Tool.

5.2. Machine Learning para identificar fallos de operación

Utilizando el módulo de Machine Learning de Elasticsearch se ejecutó un modelo predictivo que se alimenta de los logs asociados a las transacciones. El algoritmo predictivo identifica información relevante a partir de texto arbitrario creado por las fuentes generadoras de logs y almacenado en registros indexados en la base de datos de Elasticsearch, para crear un modelo predictivo.

En el algoritmo utilizado para crear el modelo de pronóstico de falla, se definió la variable dependiente “Fallas detectadas de forma preventiva” a la cual se le asocio un indicador categórico (Crítico, Severo, Medio y Normal) y las variables independientes con sus respectivos indicadores numéricos están descritas en la tabla VII. La variable dependiente debe ser definida para un tipo especifico de falla y los valores seleccionados para las variables independientes deben estar relacionadas al mismo tipo de falla.

Tabla VII. **Variables independientes**

Variable	Indicador
Rendimiento	• Utilización de CPU • Memoria RAM en uso • Memoria RAM Disponible • Capacidad del disco duro • Espacio disponible en Disco. • Velocidad de lectura en disco • Velocidad de escritura en disco • Velocidad de datos enviados Velocidad de datos recibidos
Almacenamiento	• Velocidad de respuesta de consultas • Latencia de las consultas • Tasa de solicitudes
Disponibilidad	• Nivel de servicio
Alerta	• Severidad

Fuente: elaboración propia.

5.3. Entrenamiento del modelo

Machine Learning permite entrenar un modelo con un conjunto de datos antes de ser implementado, utilizando diferentes técnicas de aprendizaje. Después del entrenamiento, al proporcionar al modelo una entrada, se recibirá un pronóstico basado en los datos que entrenaron al modelo.

La técnica utilizada para entrenar el modelo de Machine Learning descrito en el párrafo anterior fue Aprendizaje Supervisado con un sistema de clasificación *multicase* que implementa el algoritmo de aprendizaje estándar del sector conocido como regresión logística multinomial. Este aplica la clasificación, regresión y detección de valores atípicos encontrados en los datos para predecir fallos en el comportamiento de las aplicaciones. Usa transformaciones de índice continuas para convertir un índice de logs de aplicaciones en una vista de actividades centrada en el usuario y crea un modelo de detección de fallos

mediante clasificación de severidad. Después usa el procesador de ingesta de eventos para aplicar el modelo a los datos entrantes.

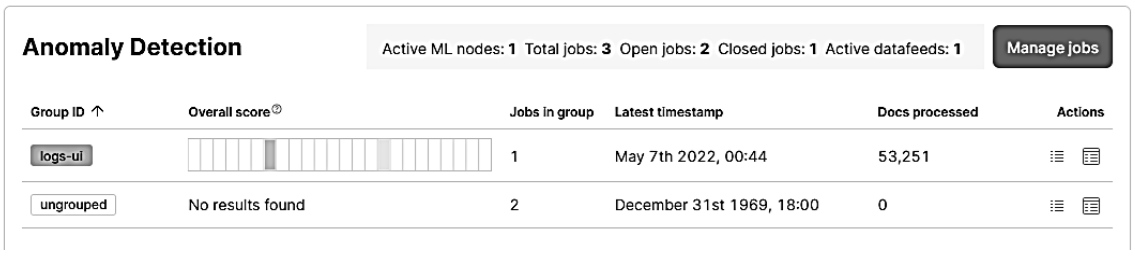
La data seleccionada para entrenar al modelo corresponde a un lote de 127006 eventos para la aplicación recibe transacciones de otros bancos por ACH. Esta aplicación realiza el proceso por lotes de archivos con transacciones que se envían por medio del sistema de ACH nacional de otros bancos hacia el banco en cual se instala el sistema de monitoreo. Cada evento corresponde a un archivo procesado, y en el conjunto de registros hay archivos procesados correctamente y archivos con error.

5.4. Implementación del modelo

La implementación de este modelo le permite al sistema de monitoreo la capacidad de pronosticar, en tiempo de ejecución, si falla un lote transacciones y enviar una alerta oportuna con la severidad adecuada.

En la figura 8, se puede observar que el modelo proceso 53,251 documentos o eventos durante el experimento.

Figura 8. **Implementación del modelo de predicción**

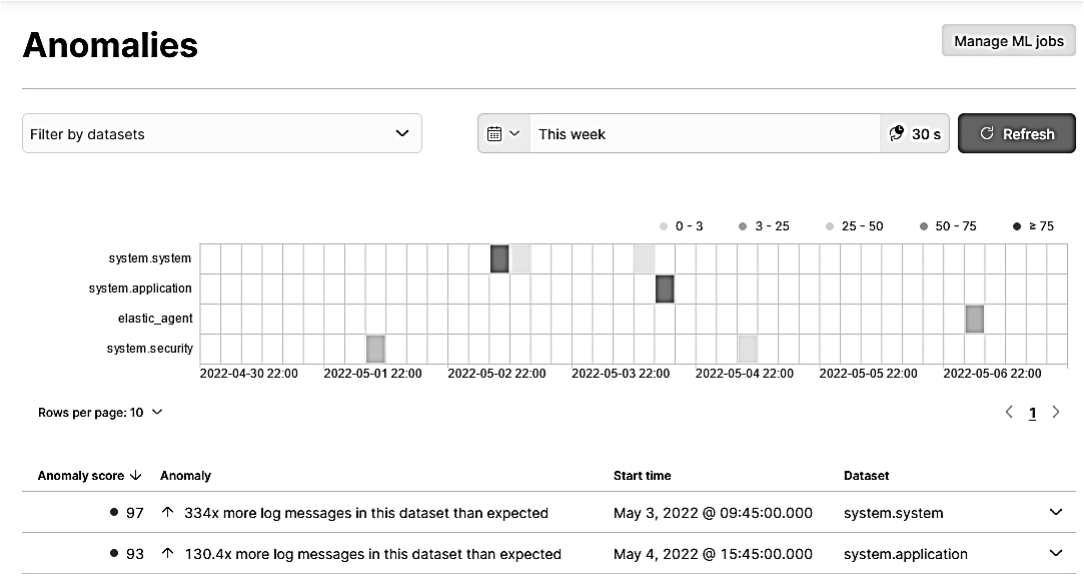


Fuente: elaboración propia, utilizando MS Snipping Tool.

5.5. Análisis de alertas detectadas por el sistema de monitoreo

La figura 9 ilustra los valores de datos reales a lo largo del tiempo para una falla en especifica las cuales son clasificadas por un modelo Machine Learning entrenado para reconocer comportamientos anómalos. Se puede examinar un evento deslizando el selector de tiempo y cambiando su duración si desea consultar información histórica y no en tiempo real.

Figura 9. **Tablero de detección de anomalías**



Fuente: elaboración propia, utilizando MS Snipping Tool.

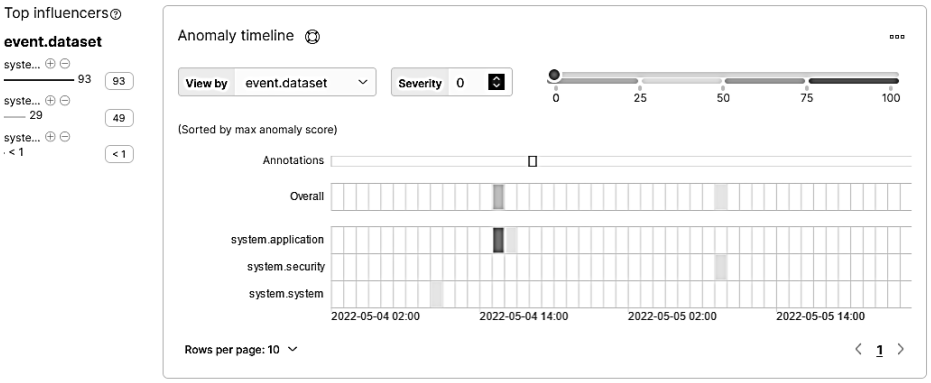
Se calcula una puntuación de anomalía para cada intervalo de tiempo, con un valor de 0 a 100. Los eventos anómalos se resaltan en colores que indican su gravedad. Si una anomalía se representa con un símbolo de cruz en lugar de un punto, tiene un impacto medio o alto. Este análisis adicional puede detectar anomalías incluso cuando se encuentran dentro de los límites del comportamiento esperado.

Si crea un pronóstico, los valores de datos pronosticados se agregan al gráfico. Un área sombreada alrededor de estos valores representa el nivel de confianza; a medida que pronostica más hacia el futuro, el nivel de confianza generalmente disminuye.

Si la gráfica del modelo está habilitada, puede mostrar opcionalmente los límites del modelo, que están representados por un área sombreada en el gráfico. A medida que el trabajo analiza más datos, aprende a predecir más de cerca los patrones de comportamiento esperados.

Del total de eventos procesados, el modelo pronosticó 93 alertas con severidad alta (ver figura 10) de la cuales, al realizar la verificación manual, se encontró que 80 si corresponden a fallos durante el proceso de la aplicación y los 13 restantes corresponden a falsos positivos reportados por el modelo. Esto le da al modelo una eficacia del 86.02 % en la predicción de eventos de fallas criticas para el proceso de archivos con transacciones enviadas de otros bancos.

Figura 10. **Categorización de alertas**

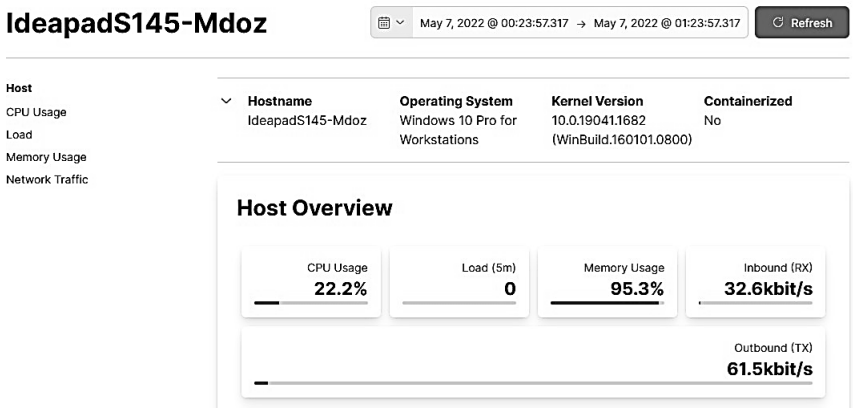


Fuente: elaboración propia, utilizando MS Snipping Tool.

5.6. Análisis de registros creados por el servidor

El sistema de monitoreo también colecta información relacionada al funcionamiento físico del servidor como se muestra en la figura 11. Esta información es útil para determinar si una falla corresponde al entorno físico en el cual se ejecuta. El sistema de monitoreo permite parametrizar alertas relacionadas a la infraestructura y asociar acciones correctivas inmediatas, como liberación de memoria RAM para optimizar el rendimiento del servidor.

Figura 11. **Tablero de visualización del estado del servidor**



Fuente: elaboración propia, utilizando MS Snipping Tool.

6. DISCUSIÓN DE RESULTADOS

Debido a la escalabilidad horizontal en la cual está basada esta arquitectura, se pueden agregar a demanda, más nodos al clúster “elasticsearch” y configurar otro clúster para garantizar la alta disponibilidad de la información en el momento que se considere necesario teniendo en cuenta la inversión en infraestructura necesaria. El alcance del sistema de monitoreo propuesto se cubre con la implementación de un clúster contiendo en un solo nodo.

Según la experiencia de la implementación de ELK en el Instituto Nacional de Investigación Espacial de Brasil Almeida, *et al*., (2018), la centralización y homologación de logs constituyen una importante herramienta en la búsqueda de soluciones a incidentes reportados. Debido a que la información se encuentra centralizada y ordenada, se determinó que es más fácil buscar información relevante para la resolución de un incidente detectado en mesa de ayuda. Esto reduce el tiempo de investigación evitando problemas como la falta de permisos a directorios donde se crean los logs, dependencia del área de desarrollo sobre ubicación del log y su estructura.

Esto concuerda con la implementación realizada en el Observatorio ALMA Gil, *et al.,* (2016), que demostró que el análisis de registros para el diagnóstico y la visualización de problemas es factible utilizando ELK, que les permitió consultar más de seis meses de información correspondiente a 6 TB de datos. Esta infraestructura fue diseñada para ser confiable, redundante y rápida, y constituye una poderosa herramienta para el soporte de aplicaciones de software. La implementación realizada de ELK también brinda al usuario la

capacidad de realizar búsquedas ordenadas a través de registros indexados de todas las fuentes generadoras de logs que se agreguen al sistema de monitoreo.

Un sistema produce tantos eventos porque en un servidor existen varias aplicaciones instaladas que producen una gran cantidad de registros que no están normalizados, lo que significa que tienen diferentes formatos dependiendo del dispositivo o aplicación de origen. Una de las claves del éxito para la correcta implementación de un sistema de monitoreo es la aplicación de estándares de la industria de LMS (Log Management System) descritos por Mitra y Sy, (2016) tales como definición de una política de auditoría, consolidación de logs, monitoreo de eventos, generación de reportes o alertas y auditoria de sistemas. La inclusión de estas prácticas fue un factor importante en la fase de diseño del sistema de monitoreo de aplicaciones en tiempo real propuesto en este trabajo.

Respecto a los algoritmos de Machine Learning, utilizados para crear modelos de predicción de fallos, este estudio determinó la importancia de realizar una correcta definición de la variable dependiente y las variables independientes, y los indicadores asociados a estas porque estas constituyen el criterio para la selección del set de datos con el cual se entrenará el modelo. Según la experiencia durante el entrenamiento del modelo utilizado en el experimento, la herramienta de Machine Learning de Elastic Stack toma la definición de la variable dependiente y las variables independientes, y utilizando un algoritmo de clasificación *multicase*, permite clasificar un evento según su severidad, siendo la severidad critica asociada a incidentes de alto impacto.

Este algoritmo, a través de procesos matemáticos sobre la data, también es capaz de seleccionar cuales son las variables independientes que tienen una alta incidencia sobre la variable dependiente y elige las significativas dependiendo el grado de confianza que se parametrice. Entre más alto el grado

de confianza, más columnas utilizara, pero esto hace más lento el cálculo en tiempo real por eso es importante definir correctamente el nivel de confianza de la predicción.

La categorización asignada a cada variable dependiente puede cambiar, según la naturaleza de la falla que se desea predecir. Por ejemplo, si se desea predecir el uso de memoria RAM de un servidor o el tráfico de entrada en un punto de red es conveniente asociar un indicador numérico, a esta variable que pueda ser pronosticado entrenando un modelo que implemente un algoritmo de aprendizaje estándar del sector conocido como regresión lineal.

La relevancia teórica de este estudio se basa en demostrar que es posible utilizar una solución de código abierto, como Elastic Stack para el monitoreo de eventos y predicción de fallos en tiempo de ejecución. Adicional, la centralización e indexación de eventos de log permiten realizar investigaciones de incidentes complejos analizando una gran cantidad de información de forma ordenada y rápida.

El impacto en la reducción de tiempos de resolución de fallos tiene incidencia directa en la forma en la cual los usuarios perciben la eficacia de un sistema informático incrementando su confianza en el mismo, por esta razón este trabajo busca contribuir en el desarrollo y soporte de sistemas para complementar la tecnología móvil que se encuentra en auge.

CONCLUSIONES

1. Se implementó la solución *Elastic Stack* para la centralización de los registros de eventos generados por sistemas operativos y las aplicaciones que forman parte del ambiente de canales electrónicos de una entidad bancaria. Los registros son capturados desde las distintas fuentes, analizados y filtrados a través de un pipeline en Logstash y almacenados con una estructura uniforme en Elasticsearch para ser consultados en tiempo real.

2. Se identificaron y predijeron fallos de operación en canales electrónicos de una entidad bancaria utilizando el módulo de Machine Learning de Elasticsearch. Se entrenó un algoritmo de aprendizaje supervisado conocido como regresión logística multinomial para crear un modelo con un sistema de clasificación *multicase* con la capacidad de identificar información relevante en un registro de log y realizar un pronóstico sobre la criticidad del evento. A este modelo se le definió una variable dependiente “Fallas detectadas de forma preventiva” a la cual se le asocio un indicador categórico: critico, severo, medio y normal.

3. Se implementó un tablero para la visualización del estatus del sistema que muestra indicadores en tiempo de ejecución o en un rango de fechas para evaluar el rendimiento del sistema de monitoreo para aplicaciones que funcionan en canales electrónicos de una entidad bancaria, a través del tiempo.

4. Se implementó un sistema de monitoreo que captura en tiempo de ejecución los registros de log generados por el sistema operativo y por las aplicaciones que forman parte del ambiente de canales electrónicos de una entidad bancaria y, utilizando un modelo de Machine Learning previamente entrenado para el experimento, tiene la capacidad de predecir fallos críticos con un 80.06 % de efectividad.

RECOMENDACIONES

1. Virtualizar un servidor de base de datos en el cual se instale el componente Elasticsearch y un servidor de aplicaciones para configurar los componentes de Logstash y Kibana. La infraestructura desplegada para esta implementación está asociada al alcance de la solución. Por esta razón únicamente se utilizó un servidor físico y dentro de él instalaron todos los componentes de Elastic Stack.

2. Diseñar una estrategia para escalar el sistema que permita incorporar más aplicaciones, creando un segundo servidor de base de datos para configurarlo como una réplica y garantizar la alta disponibilidad del sistema de monitoreo.

3. Seleccionar cuidadosamente un conjunto de datos históricos representativos para entrenar cada modelo asociado a una nueva falla que se desea predecir. Para la predicción de fallos es necesario identificar de forma individual cada una de las fallas que se desean integrar al sistema de monitoreo.

4. Realizar un proceso de depuración sobre toda la data histórica seleccionada para que pueda ser cargada a Elasticsearch y esté disponible para incluirse en el aprendizaje del modelo.

5. Crear una estrategia para la integración de fallas a predecir y priorizar las más críticas y recurrentes para que el sistema sea eficaz.

6. Modificar el archivo de configuración de Logstash y adecuarlo a cada fuente generadora de log utilizando los plugin disponibles, configurando los filtros según la estructura de log creado por cada fuente.

7. Desarrollar remediación automática para las alertas críticas detectadas por el sistema.

REFERENCIAS

1. Almeida, E.; Koga, I.; Santana, M.; Guimaraes, P.; Sugawara, L., y Eklin, T. (3 de agosto de 2017). Exploratory study of the *ELK Stack* for meteorological observation system data analysis. *Journal of Computational Interdisciplinary Sciences* 8(3), 131-142. Recuperado de https://www.researchgate.net/publication/323975183_Exploratory_study_of_the_ELK_stack_for_meteorological_observation_system_data_analysis.

2. Azarmi, B. (2017). *Learning Kibana 5.0*. Birmingham, UK: Packt Publishing.

3. Chhajed, S. (2015). *Learning ELK Stack*. Birmingham, UK: Packt Publishing.

4. Chuvakin, A. y Schmidt, K. (2012). *Logging and Log Management: The Authoritative Guide to Understanding the Concepts Surrounding Logging and Log Management*. Estados Unidos: Syngress.

5. Dixit, B.; Rogozin'ski, M., Kuc, R. y Chhajed, S. (2017). *Elasticsearch*: A Complete Guide. Packt Publishing. Birmingham, UK: Packt Publishing.

6. Gil, J.; Reveco, J. y Shen, T. (26 de julio de 2016). Operational *logs* analysis at ALMA observatory based on *ELK Stack*. *Proceedings of the SPIE,* 9913(23), 1-9. Recuperado de https://www.spiedigitallibrary.org/conference-proceedings-of-spie/9913/991323/Operational-*logs*-analysis-at-ALMA-observatory-based-on-ELK-stack/10.1117/12.2232258.full.

7. Kent, K. y Souppaya, M. (3 de septiembre de 2006). Guide to Computer Security Log Management. *NIST*, 80 (92), 1-72. Recuperado de https://nvlpubs.nist.gov/nistpubs/Legacy/SP/nistspecialpublication800-92.pdf.

8. Mitra, M. y Sy, D. (17 de novienmbre de 2016). The Rise of *Elastic Stack*. [Mensaje en un blog]. Recuperado de https://www.researchgate.net/publication/309732494_The_Rise_of_Elastic_Stack.

9. Novak, J.; Krajnc, A. y Zontar, R. (15 de julio de 2010). Taxonomy of Static Code Analysis Tools. *The 33rd International Convention MIPRO,* 1(1), 418-422. Recuperado de https://www.semanticscholar.org/paper/Taxonomy-of-static-code-analysis-tools-Novak-Krajnc/13a690909731c81a43d736e54d02994430fdb2ac.

10. Prakash, T.; Kakkar, M. y Patel, K. (2 de octubre de 2017). Geo-Identification of Web Users through *Logs* using. *6th International Conference - Cloud System and Big Data Engineering,* 606-610. [Mensaje en un blog]. Recuperado de https://www.researchgate.net/publication/305675550_Geo-identification_of_web_users_through_*logs*_using_ELK_stack/citation/download.

11. Rochim, A.; Aziz, M. y Fauzi, A. (12 de octubre de 2019). Design Log Management System of Computer Network Devices. *ICECOS*, 338-342. Recuperado de https://www.researchgate.net/publication/339095357_Design_Log_Management_System_of_Computer_Network_Devices_Infrastructures_Based_on_ELK_Stack.

12. Sachdeva, G. (2017). *Practical ELK Stack: Build Actionable Insights and Business Metrics Using the Combined Power of Elasticsearch, Logstash, and Kibana*. New Delhi, Delhi, India: Apress.

13. Turnbull, J. (2019). *The Logstash Book, Log management made easy*. Maharashtra, India: Shroff Publishers.

14. West-Brown, M.; Stikvoort, D.; Kossakowski, K.; Killcrece, G.; Ruefle, R., y Zajicek, M. (1998). *Handbook for Computer Security Incident Response Teams (CSIRTs)*. Pittsburgh, PA, USA: Carnegie Mellon. Recuperado de https://resources.sei.cmu.edu/asset_files/Handbook/2003_002_001_14102.pdf

15. Yang, C.; Kristiani, E.; Wang, Y. y Liu, M. (2019). *The Implementation of NetFlow Log System Using Ceph and ELK Stack*. New York, USA: Springer Link.