ISSN: 0123-3033 – ISSN-e: 2027-8284

INGENIERIA Y COMPETITIVIDAD
REVISTA CIENTÍFICA Y TECNOLÓGICA

Vol. 24 No. 1-2022 - DOI: 10.25100/iyc.24i1.11086

SYSTEMS ENGINEERING

Study of monitoring platforms to select the technology stack as a basis for a software testing specialized analytics system

INGENIERÍA DE SISTEMAS

Estudio de plataformas de monitoreo para seleccionar la pila tecnológica base de un sistema de analíticas especializado para pruebas de software

Carlos E. Gordillo[1§], Beatriz Florian-Gaviria[1], Eider M. Aristizábal[2]

[1]Universidad del Valle, Facultad de Ingeniería, Escuela de Ingeniería de Sistemas y Computación, Cali, Colombia

[2]Green SQA, Tecnología e Investigación Aplicada, Cali, Colombia

[§]*carlos.gordillo@correounivalle.edu.co, beatriz.florian@correounivalle.edu.co, earisti@greensqa.com*

Recibido: 19 de marzo de 2021 – **Aceptado**: 15 de octubre de 2021

Abstract

To rapidly improve the quality of web applications, it is necessary to implement automated testing strategies that allow you to comprehensively evaluate functionality and performance attributes, which are directly affected by the behavior of the hardware of the servers where they are hosted. Just as IT teams monitor the infrastructure that supports the applications, automations should be supported by monitoring platforms with specialized software testing analytics. This article presents the evaluation of four technology stacks, which will make it possible to select the most appropriate one to be included in the development of a platform specialized in software quality analytics. The four stacks in this study are: PALG, TICK, ELK, AM. This article will explain the characteristics and differences of stacks compared to various components such as: database, information filtering, collection of operating system metrics, alert system, data ingestion and visualization. Then simulations and tests carried out on each of them are described to evaluate their capabilities. Finally, conclusions are issued that will allow to select the technology stack that will be used in the development of the system specialized analytics for software testing.

Keywords: *data ingestion, monitoring platforms, software quality, technology stack.*

Como citar:
Gordillo CE, Florian-Gaviria B, Aristizábal EM. Estudio de plataformas de monitoreo para seleccionar la pila tecnológica base de un sistema de analíticas especializado para pruebas de software. INGENIERÍA Y COMPETITIVIDAD. 2022;24(1):e201811086. https://doi.org/10.25100/iyc.v24i1.11086

Resumen

Para mejorar la calidad de las aplicaciones web, es necesario implementar estrategias de pruebas automatizadas que permitan evaluar integralmente los atributos de funcionalidad y desempeño, los cuales se ven directamente afectados por el comportamiento del hardware de los servidores donde están alojadas. Así como los equipos de IT monitorean la infraestructura que soporta las aplicaciones, las automatizaciones deberían apoyarse en plataformas de monitoreo con analíticas especializadas de pruebas de software. Este artículo presenta la evaluación de cuatro pilas tecnológicas, lo cual permitirá seleccionar la más adecuada para incluirla en el desarrollo de una plataforma especializada en analíticas de calidad de software. Las cuatro pilas de este estudio son: PALG, TICK, ELK, AM. Este artículo explicará las características y diferencias de las pilas frente a diversos componentes como: base de datos, filtrado de información, recolección de métricas del sistema operativo, sistema de alertas, ingesta y visualización de datos. Luego se describen simulaciones y pruebas realizadas sobre cada una de ellas para evaluar sus capacidades. Finalmente se emiten unas conclusiones que permitirán seleccionar la pila tecnológica que será utilizada en el desarrollo de la plataforma de monitoreo o sistema de analíticas especializadas para soportar la realización de pruebas de software a las aplicaciones.

Palabras clave: *calidad de software, ingesta de datos, pila tecnológica, plataformas de monitoreo.*

1. Introducción

Una causa típica de la baja satisfacción de usuario frente al uso de las aplicaciones es dada por los elevados tiempos de respuesta y problemas de usabilidad [(1)]. Las empresas cuyo núcleo de negocio está soportado en aplicaciones web, corren el riesgo de disminuir su reputación o incluso de perder clientes al tener baja satisfacción de usuario en sus aplicaciones [(2)]. Las falencias de desempeño podrían ocurrir por diversas razones, siendo las más comunes: Algoritmos ineficientes que ejecutan cálculos redundantes en la CPU, fuga o mal manejo de memoria, también por un bajo poder de cómputo a nivel de hardware [(2)]. El mercado actual exige velocidad de desarrollo e innovación, por esto, es necesario realizar pruebas automatizadas en donde se realicen simulaciones de un gran número de usuarios y situaciones de carga y estrés en el software; paralelamente es necesario realizar monitoreo en línea de los servidores para detectar los momentos o situaciones en donde se presenta lentitud en las aplicaciones.

Con robots de pruebas de software ejecutados masivamente, se pueden hacer mediciones continuas del rendimiento y estado de correctitud de los procesos de negocio. Esta estrategia de pruebas que puede ser invasiva o no invasiva genera muchos datos, los cuales deben ser organizados y visualizados de manera correcta. Las herramientas de monitoreo pueden potenciar el almacenamiento y la visualización de las analíticas para pruebas de software, tanto funcionales como de rendimiento, y esto se puede visualizar e interpretar al mismo tiempo que las pruebas de software se estén ejecutando.

El objetivo de este artículo es evaluar las principales pilas tecnológicas de monitoreo para seleccionar aquella que será utilizada en el desarrollo de un sistema de analíticas o plataforma de monitoreo para pruebas funcionales y de rendimiento, esta plataforma de monitoreo será de gran aporte para la realización de las pruebas de software a las aplicaciones, ya que permitirá agilizar el proceso de análisis de resultados de las pruebas, al poder visualizar de forma inmediata y en línea el resultado de las pruebas, obteniendo información en tiempo real, a través de gráficos y tablas organizadas y comprensibles del comportamiento de los robots o agentes de pruebas y del comportamiento del hardware de los servidores. Las pilas por evaluar son:

- PALG: Prometheus, Alert Manager, Grafana.

- TICK: Telegraf, InfluxDB, Chronograf, Kapacitor.
- ELK: Elasticsearch, Logstash, Kibana.

AM: Microsoft Azure Monitor.

2. Metodología

Para realizar la selección de la pila tecnológica el primer paso fue realizar una investigación del comportamiento de las pilas en componentes como son: bases de datos, filtrado de información, administración, recopilación de métricas de rendimiento del sistema operativo, sistema de alertas y visualización de datos. Una vez realizada la investigación, se procedió a comparar estos componentes en cada una de las pilas para saber cuál era la más adecuada para el soporte de la plataforma de monitoreo.

Después de esto, se realizó una simulación en cada una de las pilas, en la cual se evaluó el comportamiento de estas como soporte de la plataforma, esta consistía en simular que la plataforma de monitoreo servía como herramienta para analizar los resultados de pruebas de rendimiento, para esto se ejecutaron robots que realizaban pruebas a una aplicación alojada en una máquina virtual, los resultados de estas pruebas eran enviados a la plataforma de monitoreo por los robots, al igual que el comportamiento del hardware de la máquina donde estaba alojada la aplicación durante el proceso de prueba. Posteriormente, ingresando a la plataforma se podían visualizar los resultados de las pruebas de rendimiento.

Finalmente, se realizó una prueba de ingesta de datos, esta consistía en enviar gran cantidad de registros a las bases de datos de las pilas y evaluar cuáles de ellas registraban todos los datos en el menor tiempo y correctamente, para así poder determinar cuáles eran las pilas más eficientes a la hora de almacenar las métricas que luego serían usadas como analíticas especializadas.

3. Resultados

3.1. Bases de datos

En esta sección se describen las características de cada pila en el componente de base de datos, el cual es fundamental para saber cómo se almacenan los datos generados por las pruebas o simulaciones.

3.1.1. Pila PALG (Prometheus, Alert Manager, Grafana)

La base de datos de la pila PALG es Prometheus, esta almacena fundamentalmente todos los datos como series de tiempo*:* secuencias de valores con marca de tiempo que pertenecen a la misma métrica y al mismo conjunto de dimensiones etiquetadas. Además de las series de tiempo almacenadas, Prometheus puede generar series de tiempo derivadas temporales como resultado de las consultas [(3)].

Cada serie de tiempo se identifica de forma única por su nombre de métrica y pares de clave-valor opcionales llamados etiquetas [(3)].

Las muestras forman los datos reales de la serie temporal. Cada muestra consta de:

- Un valor float64.
- Una marca de tiempo de precisión de milisegundos.

3.1.2. Pila TICK (Telegraf, InfluxDB, Chronograf, Kapacitor)

La base de datos de la pila TICK es InfluxDB, la cual tiene pares clave-valor como etiquetas, InfluxDB tiene un segundo nivel de etiquetas llamadas campos, que tienen un uso más limitado. InfluxDB admite marcas de tiempo con una resolución de hasta nanosegundos y tipos de datos float64, int64, bool y string.

InfluxDB es similar a una base de datos SQL, pero está especialmente diseñado para datos de series de tiempo. Las bases de datos relacionales pueden manejar datos de series de tiempo, pero no están optimizadas para cargas de trabajo de series de tiempo comunes. InfluxDB está diseñado para almacenar grandes volúmenes de datos de series de tiempo y realizar rápidamente análisis en tiempo real de esos datos [(4)].

InfluxDB es una base de datos de series de tiempo, este fue diseñado para tener una alta calidad de escritura de consultas y almacenamiento de datos [(5)].

3.1.3. Pila ELK (Elasticsearch, Logstash, Kibana)

La base de datos de la pila ELK es Elasticsearch, algunas de sus características son:

- Es una base de datos NoSQL orientada a documentos JSON, por lo cual, no necesita que se definan esquemas a la hora de insertar los datos [(6)].
- Permite indexar grandes volúmenes de datos, para poder consultarlos posteriormente. Elasticsearch se basa en los documentos JSON para poder realizar esta indexación. El documento JSON es un conjunto de pares clave/valor. Las claves son cadenas de texto y los valores pueden ser cadenas, números, fechas o listas [(6)].
- El proceso de añadir información a Elasticsearch se llama “indexación”, ya que cuando se insertan datos es una base de datos distribuida que escala de manera dinámica de forma horizontal, por lo que a mayor demanda se puede ir creciendo en nodos, llegando a poder almacenar petabytes de información [(6)].

Elasticsearch es principalmente un motor de búsqueda y analíticas, este soporta consultas extendidas con múltiples agregaciones, filtros y funciones de indexación [(7)].

3.1.4. Pila AM (Azure Monitor)

En la pila AM, los datos son almacenados en Azure Monitor y son alojados en la nube de Azure, algunas características de Azure Monitor son:

- Almacena y analiza toda la telemetría operativa en un almacén de datos centralizado, totalmente administrado y escalable, optimizado para ofrecer una mejor relación costo-rendimiento [(8)].
- Contiene una plataforma de análisis y un extenso lenguaje de consulta para analizar y derivar conclusiones e interactuar con ellas a partir de enormes volúmenes de datos operativos en segundos [(8)].
- Aísla las anomalías y detecta problemas rápidamente utilizando algoritmos de aprendizaje automático y análisis inteligentes [(8)].

Es una herramienta que está fuertemente orientada a las herramientas de desarrollo propias de Microsoft, y destaca como puede presentar analíticas sólidas obtenidas de grandes volúmenes de eventos, logs, métricas, transacciones y eventos de seguridad [(9)].

3.2. Filtrado de información

En esta sección se describen las características de cada pila en el componente de filtrado de datos, este componente sirve para agregar campos o etiquetas a los datos que permiten filtrar o diferenciar la información dependiendo de cada cliente, proyecto, entre otras variables.

3.2.1. Pila PALG (Prometheus, Alert Manager, Grafana)

Para filtrar la información en la pila PALG se utilizan las etiquetas, estas habilitan el modelo de

datos dimensional de Prometheus: Cualquier combinación dada de etiquetas para el mismo nombre de métrica identifica una instanciación dimensional particular de esa métrica [3].

3.2.2. Pila TICK (Telegraf, InfluxDb, Chronograf, Kapacitor)

Para poder filtrar la información en InfluxDB se debe hacer uso de las etiquetas (tags), las cuales se adicionan a los campos o mediciones y permite poder diferenciarlos según se desee, una de las formas de adicionar etiquetas a las mediciones es agregarlas por medio del recopilador de métricas del sistema llamado Telegraf, en la configuración de este, se adicionan etiquetas globales a las métricas del sistema que recopila. Posteriormente se pueden filtrar estas mediciones en InfluxDB por medio de la función group by <tag>.

3.2.3. Pila ELK (Elasticsearch, Logstash, Kibana)

El filtrado de datos en la pila ELK se realiza por medio de Logstash, este recibe los datos desde los recolectores como Metricbeat, luego los filtra o transforma y finalmente envía los datos filtrados a la base de datos de Elasticsearch. Logstash viene listo para usar con muchas agregaciones y mutaciones junto con capacidades de búsqueda dinámica, mapeo geográfico y coincidencia de patrones.

3.2.4. Pila AM (Azure Monitor)

Para el filtrado de los datos en Azure Monitor es necesario crear métricas personalizadas con campos específicos que permitan filtrar estas métricas, la creación de métricas personalizadas se logra por medio del SDK de Application Insights.

3.3. Administración

En esta sección se describen las características de cada pila en el componente de administración, este componente permite administrar la pila tecnológica y algunas pilas contienen funciones adicionales con beneficios adicionales que serán mencionados en cada pila.

3.3.1. Pila PALG (Prometheus, Alert Manager, Grafana)

Prometheus proporciona un conjunto de APIs de administración para facilitar la automatización y las integraciones [10].

3.3.2. Pila TICK (Telegraf, InfluxDb, Chronograf, Kapacitor)

En la pila TICK el administrador de la base de datos InfluxDB es el Chronograf, el cual permite tanto la administración de la base de datos como la visualización de los datos por medio de gráficos. Las opciones más comunes de administración que ofrece el Chronograf son:

- Ver todos los hosts y sus estados en la infraestructura [11].
- Crear y eliminar bases de datos y políticas de retención [11].
- Crear, eliminar y asignar permisos a los usuarios [11].
- Las consultas continuas (Continuous Query) de InfluxDB son consultas que se ejecutan de forma automática y periódica en datos en tiempo real y almacenan los resultados de las consultas en una medición específica [11].

3.3.3. Pila ELK (Elasticsearch, Logstash, Kibana)

El administrador de la pila ELK es Kibana, el cual hace que los datos sean procesables al proporcionar funciones clave como son:

- Una interfaz de usuario para administrar la configuración de seguridad, asignar roles de usuario, tomar instantáneas, acumular datos, etc.

- Detecta las anomalías que se esconden en los datos de Elasticsearch y explora las propiedades que influyen significativamente en ellas con características de Machine Learning no supervisado [12].
- Un centro de operaciones centralizado para las soluciones de Elastic. Desde el análisis de registros hasta el descubrimiento de documentos.

3.3.4. Pila AM (Azure Monitor)

Los roles integrados en Azure Monitor están diseñados para ayudar a limitar el acceso a recursos de una suscripción, sin impedir que los responsables de la infraestructura de supervisión obtengan y configuren los datos que necesitan. Azure Monitor proporciona dos roles de fábrica: un lector de supervisión y un colaborador de supervisión [13].

Las personas asignadas al rol lector de supervisión pueden ver todos los datos de supervisión en una suscripción, pero no pueden modificar cualquier recurso o editar cualquier configuración relacionada con la supervisión de recursos [13].

Las personas asignadas al rol colaborador de supervisión pueden ver todos los datos de supervisión en una suscripción y crear o modificar la configuración de supervisión, pero no pueden modificar los demás recursos [13].

3.4. Recolección de métricas del sistema

En esta sección se describen las características de cada pila en el componente de recolección de métricas del sistema operativo, el cual es fundamental para el monitoreo de los servidores mientras se realizan pruebas en ellos. Un agente de recolección es un programa de software especializado que capta información significativa sobre entidades monitoreadas como pueden ser distintos tipos de terminales, bases de datos o dispositivos de red, etc., la encapsula en entradas de datos cuantitativos y reporta estas entradas a un sistema de monitoreo en intervalos regulares. Estas entradas son transformadas en métricas que pueden ser almacena-das en series temporales, es decir, secuencias de datos, observaciones o valores ordenados cronológicamente. La recolección de datos puede ser un proceso continuo o también puede darse de forma periódica en intervalos regulares de tiempo, dependiendo la naturaleza de las mediciones y del costo de los recursos involucrados en dicha recolección de datos [14].

3.4.1. Pila PALG (Prometheus, Alert Manager, Grafana)

Prometheus permite la integración con múltiples exportadores de métricas, por lo cual puede extraer las métricas recopiladas por los exportadores y almacenarlas en la base de datos, el exportador más adecuado para recopilar métricas de rendimiento del sistema y por lo tanto para el monitoreo de los servidores es el Node Exporter.

3.4.2. Pila TICK (Telegraf, InfluxDb, Chronograf, Kapacitor)

Telegraf es un agente de servidor basado en complementos para recopilar e informar métricas y es la primera pieza de la pila TICK . Telegraf tiene complementos para obtener una variedad de métricas directamente del sistema en el que se está ejecutando, extraer métricas de APIs de terceros o incluso escuchar métricas a través de estadísticas y servicios al consumidor de Kafka [15].

3.4.3. Pila ELK (Elasticsearch, Logstash, Kibana)

Metricbeat es un colector de métricas ligero que se instala en los servidores para recopilar periódicamente métricas del sistema operativo y de los servicios que se ejecutan en el

servidor. Metricbeat toma las métricas y estadísticas que recopila y las envía a la salida que se especifique, como Elasticsearch o Logstash [16]. Metricbeat ayuda a monitorear los servidores mediante la recopilación de métricas del sistema y los servicios que se ejecutan en el servidor [16].

3.4.4. Pila AM (Azure Monitor)

Application Insights es una característica de Azure Monitor, que es un servicio de Application Performance Management (APM). Se utiliza para supervisar las aplicaciones en directo. Ayuda a detectar automáticamente anomalías en el rendimiento e incluye eficaces herramientas de análisis que ayudan a diagnosticar problemas y a saber lo que hacen realmente los usuarios con la aplicación. Está diseñado para ayudar a mejorar continuamente el rendimiento y la facilidad de uso [17]. Application Insights está dirigido al equipo de desarrollo y ayuda a conocer el rendimiento de una aplicación y cómo se utiliza [17].

3.5. Sistema de alertas

En esta sección se describen las características de cada pila en el componente de sistema de alertas, útil para generar alertas o notificaciones cuando los valores de monitoreo estén por fuera de los límites.

3.5.1. Pila PALG (Prometheus, Alert Manager, Grafana)

El sistema de alerta de la pila PALG es el AlertManager, el cual maneja alertas enviadas por aplicaciones cliente como el servidor de Prometheus. Se encarga de duplicarlas, agruparlas y enrutarlas a la integración correcta del receptor, como correo electrónico, PagerDuty u OpsGenie. También se encarga del silenciamiento e inhibición de alertas [18].

3.5.2. Pila TICK (Telegraf, InfluxDb, Chronograf, Kapacitor)

El sistema de alerta de la pila TICK es Kapacitor, el cual es un marco de procesamiento de datos de código abierto que facilita la creación de alertas, la ejecución de trabajos ETL y la detección de anomalías. Kapacitor es la pieza final de la TICK [19].

3.5.3. Pila ELK (Elasticsearch, Logstash, Kibana)

La pila ELK en sí viene con capacidades de alerta limitadas. Se deben configurar complementos de terceros para recibir alertas y notificaciones, y eso tampoco es lo suficientemente fuerte, ya que necesita otros complementos para enviar notificaciones.

3.5.4. Pila AM (Azure Monitor)

Las alertas de Azure Monitor informan de forma proactiva de los estados críticos e intentan aplicar acciones correctivas. Las reglas de alertas basadas en métricas proporcionan avisos casi en tiempo real que se generan en función de unos valores numéricos, mientras que las reglas basadas en registros permiten aplicar una lógica compleja entre datos de diversos orígenes [20].

3.6. Visualización de datos

En esta sección se describen las características de cada pila en el componente de visualización de datos, este componente es de los más importantes ya que permite crear y diseñar tableros para la visualización de los datos almacenados en tiempo real.

3.6.1. Pila PALG (Prometheus, Alert Manager, Grafana)

El sistema de visualización de la pila PALG es Grafana, este permite consultar, visualizar, alertar y explorar las métricas sin importar dónde estén almacenadas. Es un lenguaje sencillo,

proporciona herramientas para convertir los datos de una base de datos de series de tiempo (TSDB) en gráficos y visualizaciones profesionales [21].

Las características principales son:

- Permite explorar métricas y registros.
- Permite definir visualmente las reglas de alerta para las métricas más importantes [21].
- Permite anotar gráficos con eventos enriquecidos de diferentes fuentes de datos [21].
- Se pueden hacer uso de cientos de paneles y complementos en la biblioteca oficial [21].

3.6.2. Pila TICK (Telegraf, InfluxDb, Chronograf, Kapacitor)

El sistema de visualización de la pila TICK es el Chronograf, algunas de sus características son:

- Se pueden supervisar los datos de las aplicaciones con los paneles de control creados automáticamente por Chronograf.
- Se pueden crear paneles propios personalizados con varios tipos de gráficos y variables de plantilla.
- Se pueden investigar los datos con el explorador de datos y las plantillas de consulta de Chronograf.

3.6.3. Pila ELK (Elasticsearch, Logstash, Kibana)

El sistema de visualización de la pila ELK es Kibana el cual ofrece muchas opciones para mostrar los datos. Se debe utilizar Lens, una interfaz para crear rápidamente gráficos, tablas, métricas y más. Si hay una mejor visualización para los datos, **Lens la** sugiere, lo que permite un cambio rápido entre los tipos de visualización. Una vez que las visualizaciones estén de la manera que se desea, se debe usar Dashboard para recopilarlas en un solo lugar [12].

3.6.4. Pila AM (Azure Monitor)

Azure Monitor cuenta con sus propias características para visualizar los datos de supervisión y utiliza otros servicios de Azure para publicarlos ante diferentes destinatarios. Los paneles de Azure permiten combinar distintos tipos de datos, como métricas y registros, en un único panel de Azure Portal. Si se desea, también compartir el panel con otros usuarios de Azure. Los elementos que componen Azure Monitor pueden agregarse a un panel de Azure, así como a la salida de cualquier consulta de registro o gráfico de métricas [20].

3.7. Evaluación crítica de componentes de las pilas para soportar Green Heart

En esta sección, se muestra la Tabla 1 que compara los diferentes componentes en cada una de las pilas, resume las características principales de los componentes y muestran los aspectos positivos y negativos según la crítica de los autores, especialmente para soportar la plataforma de monitoreo. Los aspectos positivos se mostrarán seguidos del icono ✓. Mientras que los aspectos negativos irán seguidos de este icono ✕. A continuación de la tabla se encuentran las explicaciones para estas valoraciones positivas y negativas.

Después de realizada la investigación de los componentes de cada pila, se puede deducir que las pilas PALG y TICK tienen bases de datos de series de tiempo orientadas a métricas, lo cual podría representar una optimización para almacenar y visualizar los datos de Green Heart. Por otro lado, se observa que la base de datos de la pila ELK está orientada más a registro y documentos Json, lo cual podría provocar menor velocidad comparada con las de series de tiempo, además utiliza un escalamiento horizontal, lo cual podría ocasionar mayor consumo de memoria y por ende mayores costos. También vale la pena destacar que el Chronograf de la pila TICK tiene una funcionalidad que puede ser beneficiosa como es el continuos query.

Tabla 1. *Evaluación de componentes de las pilas tecnológicas para soportar la plataforma de monitoreo Green Heart*

Pila	Base de datos	Administración (características)	Métricas del sistema	Sistema de alertas	Visualización de datos
PALG	TSDB, base de datos de series de tiempo. ✓	conjunto de APIs de administración. ✗	Node Exporter. ✓	Alert Manager ✓	Grafana. ✓
TICK	TSDB, base de datos de series de tiempo. ✓	Chronograf .Continuous Query. Para optimizar la base de datos ✓	Telegraf. ✓	Kapacitor. ✓	Chronograf. ✓
ELK	Base de datos NoSQL **orientada a documentos JSON.** ✗	Características de Machine Learning para apoyar la administración de la base de datos ✓	Metricbeat ✓	No tiene sistema de alertas. ✗	Kibana ✓
AM	Incluye registros y métricas. ✓	Roles integrados en Azure Monitor ✓	Application Insights ✗	Alertas de Azure Monitor. ✓	Azure Monitor ✓

Fuente: propia

En cuanto a la pila ELK cabe destacar que no posee sistema de alertas propio. En cuanto a los recolectores de métricas del sistema operativo parecen ser muy completos en todas las pilas, aunque en la pila AM parece estar más enfocado al monitoreo de las aplicaciones y no al monitoreo de los servidores.

3.8. Simulación de monitoreo de servidores

Hasta este momento se realizó una investigación sobre los componentes y características de cada pila, y se pudieron observar los aspectos positivos y negativos de estas, pero esta información no es suficiente criterio para seleccionar una de ellas, por esto fue necesario realizar una simulación con cada una de las pilas, en donde se pudo evaluar en detalle el comportamiento de sus componentes y su capacidad de ser usadas para soportar la plataforma de monitoreo.

La prueba de concepto consistía en simular que la plataforma tenía varios clientes y estos clientes tenían servidores los cuales serían monitoreados por medio de esta. para esto se instaló una aplicación web en una máquina virtual, la cual podía ser accedida a través de una dirección IP pública. Posteriormente, se crearon robots por medio de una herramienta de automatización de pruebas, estos robots accedían a la aplicación web y simulaban usuarios ingresando a la aplicación y verificando la usabilidad de esta. Luego, se ejecutaban estos robots masivamente, simulando múltiples usuarios ingresando a la aplicación generando gran transaccionalidad sobre esta, al mismo tiempo estos robots enviaban información a la plataforma.

Por otro lado, se instalaba un recolector de métricas de rendimiento del hardware en la máquina servidor donde estaba instalada la aplicación y este enviaba toda la información del rendimiento de la máquina a la plataforma. Finalmente, en la plataforma se diseñaron tres niveles de visualización del comportamiento de las pruebas, distribuidos de la siguiente manera:

En el primer nivel orientado a nivel de cliente, se mostraba la ubicación de los servidores en el mapa mundial, las plataformas (sistema operativo) de los servidores y los proyectos asociados al cliente.

El segundo nivel orientado a nivel de proyecto, se mostraba la información de los nodos o servidores que tiene cada proyecto, el listado de iteraciones de pruebas realizadas y la supervisión de las alertas activas.

El tercer nivel orientado hacia el nodo mostraba la información específica y detallada de cada nodo o servidor, enfocado en monitoreo de rendimiento, por lo cual mostraba información de uso de CPU, memoria RAM y disco.

3.8.1. Simulación en Pila PALG (Prometheus, Alert Manager, Grafana)

Para la pila PALG (Figura 1)se realizó la simulación de que se instalaba el componente **Node Exporter** en cada uno de los servidores, este se encargaba de recopilar métricas del comportamiento del hardware de la máquina y exponer esas métricas en una dirección IP pública, en un formato compatible con **Prometheus**. Después de esto, estas métricas expuestas eran extraídas por el **servidor HTTP de Prometheus** instalado en el servidor de la plataforma, después estas métricas eran almacenadas en la **base de datos TSDB de Prometheus**.

Luego, se creaban unas reglas de alerta que disparaban alertas cuando se sobrepasaban los límites del comportamiento saludable del hardware del servidor, estas alertas eran administradas por el componente **Alert Manager**, el cual se encargaba de distribuirlas por correo electrónico o enviarlas a **Grafana** para su visualización en tiempo real. Paralelamente, se añadían unas etiquetas como ClientName, ProjectName, IterationId, entre otras, que permitían filtrar las métricas por cliente, proyecto, iteración, entre otras, y finalmente se diseñaban los tableros gráficos en la herramienta **Grafana**, que tenía como fuente de datos la base de datos de **Prometheus**.

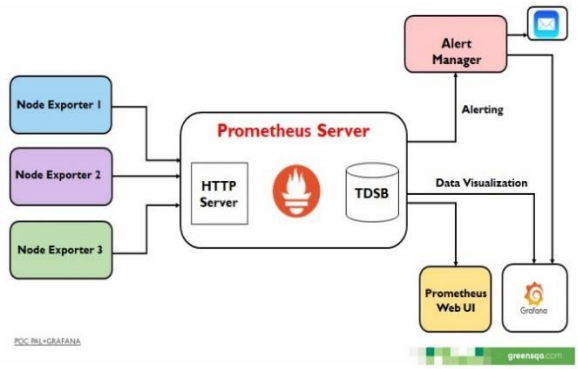

Figura 1. *Arquitectura pila PALG. Fuente: propia*

3.8.2. Simulación en Pila TICK (Telegraf, InfluxDb, Chronograf, Kapacitor)

Para la pila TICK (Figura 2)se simuló que se instalaba **Telegraf** en los servidores de los clientes, este recopilaba las métricas del comportamiento de hardware de la máquina y además añadía unas etiquetas globales como ClientName, ProjectName, ClientId, entre otras, que permitía filtrar las métricas por cada uno de estos ítems en la base datos, después de esto **Telegraf** enviaba las métricas recopiladas a la base de datos de **InfluxDB** instalada en el servidor de la plataforma, luego por medio del componente **Chronograf** se administraba la base de datos y se creaban unas reglas de alerta por medio de **Kapacitor**, las cuales disparaban alertas cuando se sobrepasaban los límites del comportamiento saludable del hardware del servidor, y finalmente se creaban los tableros gráficos en **Grafana,** ya que este permitía tener a **InfluxDB** como fuente de origen datos.

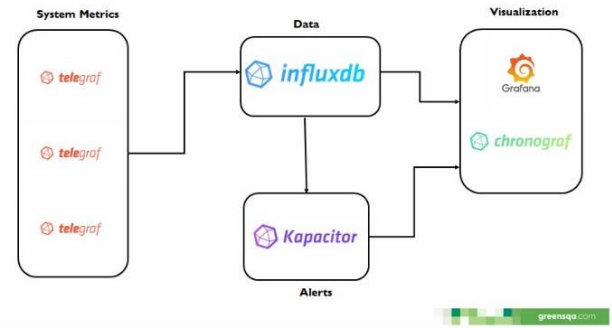

Figura 2. *Arquitectura pila TICK con Grafana. Fuente: propia*

3.8.3. Simulación en Pila ELK (Elasticsearch, Logstash, Kibana)

En cuanto a la pila ELK (Figura 3), se simuló que se instalaba **MetricBeat** en los servidores, este se encargaba de recopilar las métricas del comportamiento del hardware de la máquina y las enviaba a **Logstash**, en **Logstash** se le adicionaban a estas métricas campos como ClientName, ProjectName, IterationId, entre otros, los cuales permitían filtrar la información en la base de datos, posteriormente **Logstash** enviaba las métricas filtradas a la base de datos de **Elastic**, y finalmente se diseñaban unos tableros gráficos en **Kibana** el cual accedía a la base de datos de **Elastic** como fuente de origen de datos.

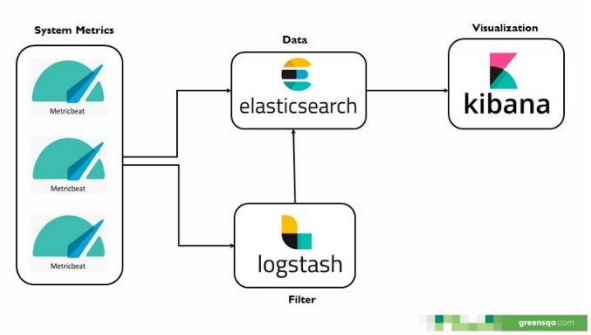

***Figura 3**. Arquitectura pila ELK. Fuente: propia*

3.8.4. Simulación en Pila AM (Azure Monitor)

En la pila AM (Figura 4) no se pudo realizar la simulación debido a dos problemas:

Application Insights recopila métricas del comportamiento del hardware de los servidores, pero estas métricas eran pocas y no mostraban información detallada de los componentes del hardware en comparación a los colectores de las otras pilas. Además, no permitía adicionar etiquetas o campos a estas métricas para permitir filtrar la información.

A pesar de que por medio de la función GetMetric del SDK de **Application Insights**, se pueden crear campos nuevos como ClientName, ProjectName, IterationId, para poder filtrar la información y de este modo enviar métricas personalizadas a **Azure Monitor**, es necesario utilizar un colector de métricas del sistema externo a la pila tecnológica o desarrollar un colector propio y adicionarle estos filtros.

Debido a esto, no se realizó la simulación y se concluyó que la pila AM era más adecuada para monitorear aplicaciones como páginas web, servicios web, máquinas virtuales y demás recursos alojados directamente en la nube de Azure, pero no era la más adecuada para soportar la plataforma de monitoreo.

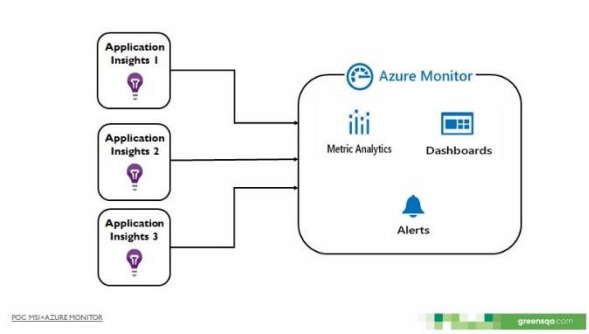

***Figura 4.** Arquitectura pila AM. Fuente: propia*

3.9. Prueba de ingesta de datos

Por último, para tener mayores criterios de selección de la pila, se optó por realizar una prueba de ingesta de datos, la prueba consistía en enviar métricas personalizadas desde una aplicación desarrollada en el lenguaje de programación C# hacia las cuatro diferentes pilas, específicamente las métricas se enviaban directamente a las bases de datos de estas, las cuales son: **Prometheus**, **InfluxDB, Elastic** y **Azure Monitor**.

Para realizar la prueba se crearon 100 métricas personalizadas en la aplicación y estas fueron enviadas a las pilas con las siguientes configuraciones:

1. 10 iteraciones con 100 métricas para 1,000 registros.

2. 100 iteraciones con 100 métricas para 10,000 registros.
3. 1000 iteraciones con 100 métricas para 100,000 registros.
4. 10.000 iteraciones con 100 métricas para 1,000,000 de registros.

El servidor donde estaban alojadas las bases de datos de las pilas tecnológicas fue una máquina virtual en el servicio de Azure, sus características fueron un sistema operativo Linux, 2 CPU virtuales y 8 Gibibytes de memoria RAM.

La aplicación o agente de pruebas desde donde se realizó el envío de las métricas se ejecutó en otra máquina virtual de Azure, esta tenía una velocidad de internet aproximada de 1,575 Megabits/sec.

Después de realizar las pruebas en cada una de las pilas, los tiempos máximos que tardaron las bases de datos en realizar el total de registros fueron los mostrados en la Tabla 2 y la Figura 5:

Tabla 2. *Prueba de ingesta de datos.*

Pilas	1,000 regs (s)	10,000 regs (s)	100,000 regs (s)	1,000,000 regs (s)
Prometheus	0.60	5.13	50.23	504.40
InfluxDB	0.53	4.34	42.32	432.10
Elastic	4.95	9.02	51.90	468.96
Azure Monitor	0.56	5.10	50.50	503.86

Fuente: propia

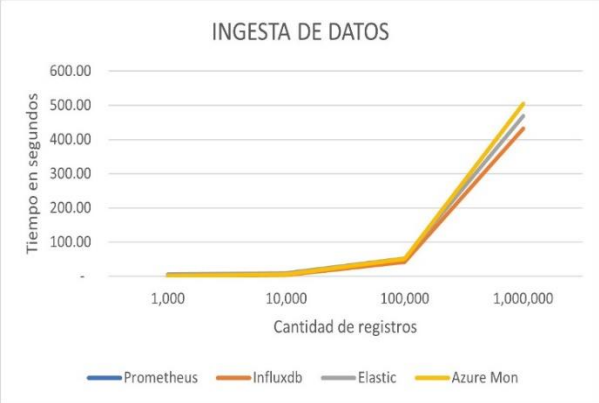

Figura 5*. Prueba Ingesta de datos. Fuente: propia*

Se puede observar de la Figura 5 que los tiempos de Prometheus y Azure Monitor son un poco más elevados (ambos presentaron tiempos similares, por esta razón en la gráfica se aprecia únicamente la línea amarilla), en la prueba, el envío de datos en estas pilas presentaron mejor velocidad de envío en un principio, sin embargo, se encontró que no se alcanzaban a almacenar correctamente las métricas en las bases de datos, es decir, que había una considerable perdida de información al realizarse el envío a esa velocidad. Por lo tanto, fue necesario ralentizar el envío sucesivo de estas y asignarle un retardo de 50 milisegundos entre un envío y otro, para así evitar pérdida de las métricas y lograr un correcto almacenamiento en las bases de datos, este fue el intervalo de tiempo mínimo que evitaba que se perdieran registros. Además, en Azure Monitor solo se pueden visualizar las métricas promediadas en intervalos de un minuto, lo que representa una gran desventaja si se requiere visualizar en detalle el comportamiento de estas.

Por otro lado, InfluxDB y Elastic guardaron todas las métricas correcta y continuamente en todas las pruebas realizadas. Se puede observar que InfluxDB registró tiempos de ingesta más rápidos que Elastic, siendo InfluxDB la que obtuvo mejores resultados en esta prueba.

4. Discusión

Después de realizar la evaluación de las pilas tecnológicas y las pruebas se realizaron comparativas de los resultados obtenidos que se describen a continuación:

La pila AM es ideal para monitorear y supervisar todo tipo de recursos de Microsoft Azure como páginas web, servicios web, máquinas virtuales, entre otros, pero no es la ideal para la plataforma de monitoreo.

En cuanto a la pila ELK, se pudo evaluar que Elastic es una base de datos orientada a

documentos JSON y está basada en Apache Lucene, por lo cual sería ideal si se desean almacenar documentos, más no es la óptima para almacenar datos numéricos de series de tiempo, ya que podría ser más lenta y tener mayor consumo de RAM y disco comparada con las bases de datos orientadas a series de tiempo, pudiendo así elevar los costos de operación y mantenimiento de los servidores.

Por otro lado, las bases de datos de Prometheus e InfluxDB, son bases de datos de series de tiempo TSDB, lo cual es óptimo para almacenar datos de métricas para el monitoreo de los servidores, teniendo mayor velocidad y menos consumo de memoria, siendo mejor InfluxDB, debido a que permite almacenar datos en intervalos de tiempo menores y con más variedad de tipos, como por ejemplo cadenas de texto, además de que su capacidad de ingesta de datos es mejor. Además, InfluxDB permite registrar métricas de forma periódica y no periódica, mientras que Prometheus es ideal para registrar métricas de forma periódica, como por ejemplo las métricas del comportamiento del hardware que se envían periodicamente en un intervalo de tiempo determinado. Mientras que InfluxDB no solo permite ese mismo tipo de ingestión, sino que registra las métricas del comportamiento de los robots utilizados en las pruebas, los cuales no tienen un intervalo de tiempo de envío definido, sino que van enviando los datos de forma continua a medida que se van ejecutando.

InfluxDB tiene una interfaz gráfica de administración muy completa, como es Chronograf, por medio de la cual se pueden crear o eliminar bases de datos y políticas de retención de datos, crear continuous query (para así reducir el volumen de datos innecesarios y por lo tanto reducir espacio de almacenamiento y mejorar el rendimiento de la base de datos) y otras funcionalidades de forma más práctica. Por lo tanto, teniendo en cuenta estas diferencias, se puede concluir que la pila TICK presenta una mejor interfaz para la administración de base de datos.

En cuanto a la recopilación de métricas de rendimiento del sistema se pudo observar que los recopiladores más completos son el Node Exporter de la pila PAL y el Telegraf de la pila TICK.

Por el lado del sistema de alertas se pudo concluir que el sistema mejor capacitado es el Alert Manager de la pila PAL, puesto que este provee un administrador de alertas donde se pueden manejar agrupamientos, silencios e inhibiciones personalizados de estas. Teniendo en cuenta la visualización de datos se concluye como la mejor opción la herramienta Grafana de la pila PALG, como la mejor para visualización y diseño de tableros, debido a que posiblemente, según la documentación observada puede brindar mejores opciones de visualización, mayor cantidad de plugins y fuentes de datos, entre otras ventajas.

5. Conclusiones

Para seleccionar la pila más adecuada para soportar la plataforma de monitoreo Green Heart en una primera instancia se descartó Azure Monitor debido a que esta está diseñada principalmente para supervisión del comportamiento de las aplicaciones y recursos de Azure, pero no es óptima para soportar la plataforma, por las razones mencionadas anteriormente.

Además, también se descartó la pila ELK, puesto que Elastic está más enfocada al almacenamiento de documentos o registros, pudiendo así provocar un consumo de memoria es más alto, elevando costos de infraestructura.

Por lo tanto, se eligieron mejores bases de datos para esté contexto Prometheus e InfluxDB, ya que al ser bases de datos de series de tiempo están optimizadas para almacenar grandes volúmenes

de información en forma de métricas y obtener visualización de estas en tiempo real.

Posteriormente se optó por elegir InfluxDB sobre Prometheus, ya que está tiene un sistema de administración más completo como es el Chronograf, además de que su capacidad de ingesta de datos es mucho más eficiente, al realizar el registro de métricas en tiempo continuo sin pérdida de información y de forma no periódica.

El recolector de métricas del sistema seleccionado fue Telegraf, ya que hace parte de la pila TICK y no presenta grandes diferencias con los otros sistemas.

El sistema de alertas seleccionado fue Kapacitor al hacer parte de la pila TICK y no tener diferencias relevantes con los otros sistemas.

Finalmente se seleccionó como sistema de visualización de datos Grafana, ya que según la documentación observada puede tener mejores capacidades que los otros sistemas y además permite soportar como fuente de datos a InfluxDB e incorporarse a la pila TICK.

6. Declaración de financiación

Los autores declaran que no recibieron financiación de ninguna entidad público/privada para la realización de este artículo.

7. Referencias

(1) Finstad K. The usability metric for user experience. Interact Comput. 2010;22(5):323–7. https://doi.org/10.1016/j.intcom.2010.04.004.

(2) ¿Por qué necesitas realizar pruebas de rendimiento a tu aplicación? Software testing bureau [Internet]. 2019 [cited 2021 Aug 1]. Available from: https://www.softwaretestingbureau.com/cuanta-carga-puede-soportar-tu-aplicacion/.

(3) Prometheus Authors. Data model. Prometheus.io [Internet]. 2016 [cited 2021 Jan 14]. Available from: https://prometheus.io/docs/concepts/data_model/.

(4) Compare InfluxDB con bases de datos SQL. influxdata.com [Internet]. 2015 [cited 2021 Mar 2]. Available from: https://docs.influxdata.com/influxdb/v1.8/concepts/crosswalk/.

(5) John AT, Chika Amadi B, Chima Umeokpala F. Detection of Anomalies in a Time Series Data using InfluxDB and Python. arXiv:2012.08439 [Preprint]. 2020 [cited 2021 Mar 2]. Available from: https://arxiv.org/abs/2012.08439.

(6) Cuervo V. ¿Qué es Elasticsearch?. arquitectoit.com [Internet]. 2019 [cited 2021 Mar 2]. Available from: http://www.arquitectoit.com/elasticsearch/que-es-elasticsearch/.

(7) Martinviita M. Time series database in Industrial IoT and its testing tool [Master's thesis]. University of Oulu, Degree Programme in Computer Science and Engineering; 2018. Available from: http://urn.fi/URN:NBN:fi:oulu-201811093007.

(8) Azure Monitor. Microsoft [Internet]. 2014 [cited 2021 Mar 2]. Available from: https://azure.microsoft.com/es-es/services/monitor/.

(9) Martín Olías P. Prueba de concepto Azure Monitor [Tesis de pregrado]. Universitat oberta de Catalunya; 2020. Available from: http://openaccess.uoc.edu/webapps/o2/bitstream/10609/108786/6/mmendozafTFG0120memoria.pdf.

(10) Prometheus Authors. Management API. Prometheus.io [Internet]. 2019 [cited

2021 Mar 2]. Available from: https://prometheus.io/docs/prometheus/latest/management_api/

(11) Documentación de Chronograf 1.8. influxdata.com [Internet]. 2017 [cited 2021 Mar 2]. Available from: https://docs.influxdata.com/chronograf/v1.8/.

(12) Kibana: Explora, visualiza y descubre datos. elastic.co [Internet]. 2020 [cited 2021 Mar 2]. Available from: https://www.elastic.co/es/kibana.

(13) Roles, permisos y seguridad en Azure Monitor. Documentos de Microsoft [Internet]. 2021 [cited 2021 Mar 2]. Available from: https://docs.microsoft.com/es-es/azure/azure-monitor/roles-permissions-security.

(14) Perera N, Otarán F. Propuesta de una solución de monitoreo para sistemas del CeSPI [Tesis de pregrado]. Universidad Nacional de La Plata; 2017. Available from: http://sedici.unlp.edu.ar/handle/10915/61525

(15) Documentación de Telegraf 1.17. influxdata.com [Internet]. 2016 [cited 2021 Mar 2]. Available from: https://docs.influxdata.com/telegraf/v1.17/.

(16) Elastic Stack and Product Documentation. elastic.co [Internet]. 2015 [cited 2021 Jan 14]. Available from: https://www.elastic.co/guide/index.html.

(17) ¿Qué es Azure Application Insights? - Monitor de Azure. Documentos de Microsoft [Internet]. 2021 [cited 2021 Mar 2]. Available from: https://docs.microsoft.com/en-us/azure/azure-monitor/app/app-insights-overview.

(18) Prometheus Authors. Alertmanager. Prometheus.io [Internet]. 2016 [cited 2021 Mar 2]. Available from: https://prometheus.io/docs/alerting/latest/alertmanager/.

(19) Kapacitor 1.5 Documentation. influxdata.com [Internet]. 2016 [cited 2021 Mar 2]. Available from: https://docs.influxdata.com/kapacitor/v1.5/.

(20) Azure Monitor documentation - Azure Monitor. Microsoft Docs [Internet]. 2021 [cited 2021 Jan 14]. Available from: https://docs.microsoft.com/en-us/azure/azure-monitor/.

(21) Grafana documentation. Grafana.com [Internet]. 2019 [cited 2021 Jan 14]. Available from: https://grafana.com/docs/grafana/latest/.