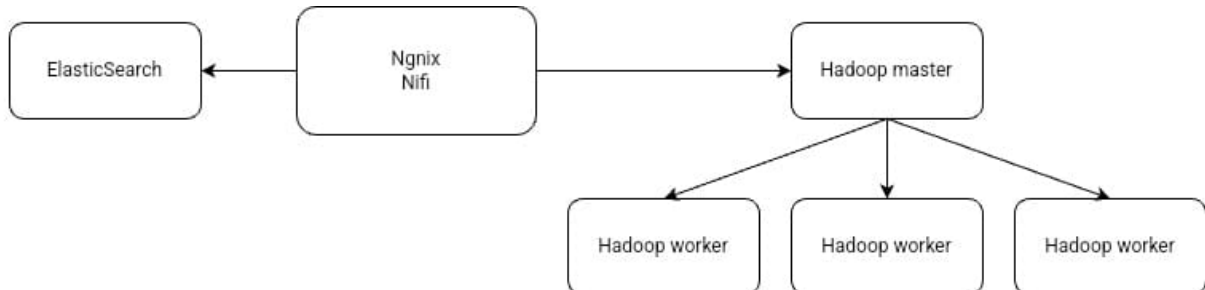


# Captura y Análisis de Logs

## 1. Arquitectura del Sistema

### Diagrama de Arquitectura



### Componentes Principales

1. **Servidor Nginx**: Genera los logs en formato específico.
2. **Apache NiFi**: Utilizado para la ingesta y procesamiento de datos.
3. **Apache Hadoop**: Almacenamiento de datos.
4. **Elasticsearch**: Almacenamiento de datos agrupados por IP.

### Justificación de uso

**Apache NiFi**: Tiene una gran capacidad de manejar flujos de datos en tiempo real. Permite una configuración visual e intuitiva de flujos de datos complejos.

**Apache Hadoop**: Tiene capacidad de almacenamiento distribuido y escalabilidad para grandes volúmenes de datos. Proporciona una solución robusta para el almacenamiento a largo plazo de logs.

**Elasticsearch**: Es muy eficiente al realizar búsquedas y agregaciones, ideal para consultas rápidas sobre logs. Su capacidad para manejar datos estructurados es ideal para tareas en las que se requiere almacenar logs agrupados por IP.

## 2. Implementación del Flujo de Datos a HDFS


### Configuración de NiFi

El flujo de NiFi se compone de tres partes principales: Flow común, Flow para HDFS y Flow para Elasticsearch.

#### Flow común:


##### 1. Lectura de Logs (TailFile):

Configurado para leer continuamente /home/ec2-user/log/nginx.log.  
Asegura la captura en tiempo real de nuevas entradas de log.

	<b>TailFile</b> TailFile 1.28.0 org.apache.nifi - nifi-standard-nar	1
In	0 (0 bytes)	5 min
Read/Write	0 bytes / 318.05 KB	5 min
Out	299 (318.05 KB)	5 min
Tasks/Time	598 / 00:00:00.229	5 min


##### 2. División de Logs (SplitText):

Divide el flujo de logs para procesarlos individualmente.

	<b>SplitText</b> SplitText 1.28.0 org.apache.nifi - nifi-standard-nar	
In	299 (318.05 KB)	5 min
Read/Write	318.05 KB / 0 bytes	5 min
Out	1,497 (316.58 KB)	5 min
Tasks/Time	299 / 00:00:00.166	5 min



##### 3. Extracción de Valores (ExtractText):

Utiliza una expresión regular para extraer campos específicos de cada log.  
Los valores extraídos se guardan como atributos key/value

	<b>ExtractText</b> ExtractText 1.28.0 org.apache.nifi - nifi-standard-nar	
In	1,497 (316.58 KB)	5 min
Read/Write	316.58 KB / 0 bytes	5 min
Out	1,497 (316.58 KB)	5 min
Tasks/Time	1,497 / 00:00:00.586	5 min

#### 4. Conversión a JSON (ConvertRecord):

Transforma los atributos extraídos a formato JSON.


	<b>AttributesToJSON</b> AttributesToJSON 1.28.0 org.apache.nifi - nifi-standard-nar	 1
In	1,492 (315.44 KB)	5 min
Read/Write	0 bytes / 440.74 KB	5 min
Out	2,984 (881.48 KB)	5 min
Tasks/Time	1,492 / 00:00:00.599	5 min

### Flow para HDFS:

#### 5. Agrupación de JSONs (MergeContent):


Agrupar múltiples JSONs en un array.

Añade "[", "]" y "," como demarcadores.

	<b>MergeContent</b> MergeContent 1.28.0 org.apache.nifi - nifi-standard-nar	
In	1,504 (444.01 KB)	5 min
Read/Write	444.01 KB / 445.76 KB	5 min
Out	296 (445.76 KB)	5 min
Tasks/Time	892 / 00:00:00.454	5 min

#### 6. Actualización de Atributo de Nombre de Archivo:

Actualiza el atributo 'filename' para reflejar el contenido agrupado.

	<b>UpdateAttribute</b> UpdateAttribute 1.28.0 org.apache.nifi - nifi-update-attribute-nar	
In	296 (445.76 KB)	5 min
Read/Write	0 bytes / 0 bytes	5 min
Out	296 (445.76 KB)	5 min
Tasks/Time	296 / 00:00:00.125	5 min

#### 7. Almacenamiento en HDFS (PutHDFS):


Configurado para almacenar los documentos JSON en HDFS.

	<b>PutHDFS</b> PutHDFS 1.28.0 org.apache.nifi - nifi-hadoop-nar	
In	0 (0 bytes)	5 min
Read/Write	0 bytes / 0 bytes	5 min
Out	0 (0 bytes)	5 min
Tasks/Time	0 / 00:00:00.000	5 min

## Flow para Elasticsearch:

### 8. Almacenamiento por IP (MergeContent):

Agrupar los JSONs por IP.

	<b>MergeContent</b> MergeContent 1.28.0 org.apache.nifi - nifi-standard-nar	
In	1,504 (444.01 KB)	5 min
Read/Write	444.01 KB / 445.76 KB	5 min
Out	296 (445.76 KB)	5 min
Tasks/Time	892 / 00:00:00.454	5 min

### 9. Transformación de Estructura (JoltTransform):

Reestructura el JSON extrayendo la IP como atributo.


Agrupar lo demás en un array llamado requests.

 **JoltTransformJSON**  
JoltTransformJSON 1.28.0  
org.apache.nifi - nifi-standard-nar

In	1,497 (445.05 KB)	5 min
Read/Write	445.05 KB / 464.06 KB	5 min
Out	1,497 (464.06 KB)	5 min
Tasks/Time	1,497 / 00:00:00.589	5 min

### 10. Actualización de Nombre de Archivo:

Actualizar el atributo filename a IP.json.



## UpdateAttribute


UpdateAttribute 1.28.0  
org.apache.nifi - nifi-update-attribute-nar

In	1,497 (464.06 KB)	5 min
Read/Write	0 bytes / 0 bytes	5 min
Out	1,497 (464.06 KB)	5 min
Tasks/Time	1,497 / 00:00:00.348	5 min

### 11. Inserción en Elasticsearch (PutElasticsearchHttp):

Configurado para realizar un upsert en Elasticsearch.

Utiliza un script para crear nuevos documentos o añadir a los existentes.



▶

PutElasticsearchJson

PutElasticsearchJson 1.28.0

org.apache.nifi - nifi-elasticsearch-restapi-nar

In	4,542 (1.38 MB)	5 min
Read/Write	1.38 MB / 0 bytes	5 min
Out	0 (0 bytes)	5 min
Tasks/Time	59 / 00:00:05.157	5 min

### 3. Implementación del Flujo de Datos a Elasticsearch

#### Transformación de Datos

El flujo para Elasticsearch extiende el flujo común y añade los siguientes pasos:

**1. Agrupación por IP (MergeContent):**

Agrupar los JSONs basándose en la dirección IP.

**2. Transformación de Estructura (JoltTransform):**

Reestructura el JSON extrayendo la IP como atributo.

Agrupar el resto de la información en un array llamado 'requests'.

**3. Actualización de Nombre de Archivo:**

Actualizar el atributo 'filename' a '{IP}.json'.

#### Ingesta en Elasticsearch

**4. Inserción en Elasticsearch (PutElasticsearchHttp):**

Configurado para realizar un upsert en Elasticsearch.

Utiliza un script para crear nuevos documentos o añadir a los existentes.

## 4. Despliegue y Configuración

Se han utilizado diferentes playbooks para automatizar el despliegue:

Elasticsearch: Configura un grupo de seguridad y lanza una instancia ec-2. Luego, instala Java y Elasticsearch en la instancia, configurando los parámetros necesarios. Finalmente, reinicia el servicio Elasticsearch para aplicar los cambios y crea un índice inicial.

Hadoop: Configura grupos de seguridad y lanza instancias ec-2 para el nodo master y los workers. Luego, instala y configura Hadoop y Java en todas las instancias, con las configuraciones específicas necesarias tanto para el master como los workers. Finalmente, inicia los servicios necesarios en cada nodo.

Nginx y NiFi: Crea grupos de seguridad y lanza una instancia ec-2 con Nginx. Luego, instala Docker, Git y genera logs de Nginx usando un contenedor. Instala Java y NiFi, configura sus servicios y establece credenciales de usuario. Finalmente configura un flujo de NiFi, realizando todas las configuraciones y arranques necesarios.

## Problemas Encontrados y Soluciones

### 1. Parsing de logs de Nginx:

Solución: Diseño cuidadoso de la expresión regular en ExtractText.

### 2. Agregación de logs por IP:

Solución: Implementación de lógica de agrupación con RouteOnAttribute y MergeRecord.

### 3. Conexión NiFi-HDFS:

Solución: Referenciación de los archivos de configuración de Hadoop en PutHDFS.

# 5. Resultados

## Browse Directory

/user/ec2-user/user/nifi/us

Go!

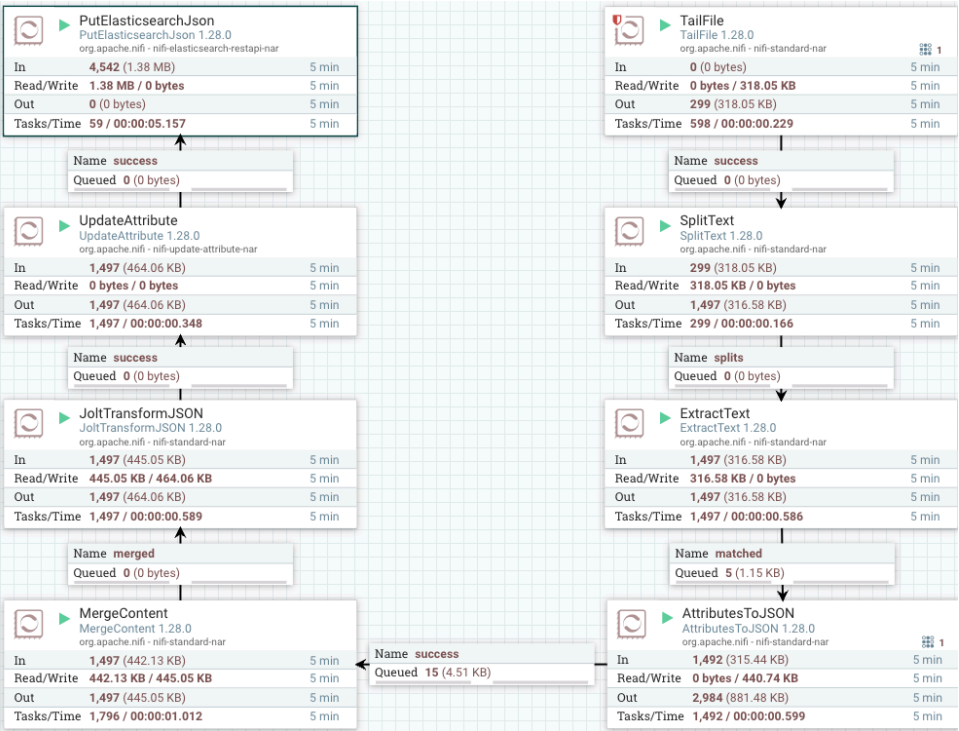
Show

25

entries

Search:

<input type="checkbox"/>	Permission	Owner	Group	Size	Last Modified	Replication	Block Size	Name	
<input type="checkbox"/>	-rw-r--r--	ec2-user	supergroup	1.54 KB	Nov 14 20:57	3	128 MB	0001324c-1786-4bf8-9383-0b81e0461594.json	
<input type="checkbox"/>	-rw-r--r--	ec2-user	supergroup	1.52 KB	Nov 14 22:31	3	128 MB	00017f88-d0c5-4312-9180-09c78bc8307c.json	
<input type="checkbox"/>	-rw-r--r--	ec2-user	supergroup	1.47 KB	Nov 15 14:42	3	128 MB	000c663a-05c9-420c-baf8-d7a40a02207c.json	
<input type="checkbox"/>	-rw-r--r--	ec2-user	supergroup	1.49 KB	Nov 14 21:48	3	128 MB	000d2f2e-d853-419a-bd60-c2d7d46e61fd.json	
<input type="checkbox"/>	-rw-r--r--	ec2-user	supergroup	1.49 KB	Nov 15 23:22	3	128 MB	000dc89d-8311-4834-ac85-718ddc14fe43.json	
<input type="checkbox"/>	-rw-r--r--	ec2-user	supergroup	1.39 KB	Nov 14 20:53	3	128 MB	00121cb7-cc77-46d1-9706-f5237f51f346.json	
<input type="checkbox"/>	-rw-r--r--	ec2-user	supergroup	51.53 KB	Nov 15 17:58	3	128 MB	00127416-083c-45ee-9dbe-bf913f962a79.json	
<input type="checkbox"/>	-rw-r--r--	ec2-user	supergroup	1.42 KB	Nov 14 21:08	3	128 MB	00140b3b-a942-4336-8fd3-7278ce0aab62.json	
<input type="checkbox"/>	-rw-r--r--	ec2-user	supergroup	1.44 KB	Nov 14 22:36	3	128 MB	00171e60-055f-45b9-81e3-dfa5fa8ab2b3.json	
<input type="checkbox"/>	-rw-r--r--	ec2-user	supergroup	1.57 KB	Nov 14 20:40	3	128 MB	00175446-0999-4e46-ba64-c54cb4eae8f.json	
<input type="checkbox"/>	-rw-r--r--	ec2-user	supergroup	1.57 KB	Nov 15 18:48	3	128 MB	00175709-e977-4367-a413-ab1bfe00664c.json	
<input type="checkbox"/>	-rw-r--r--	ec2-user	supergroup	58.23 KB	Nov 15 15:55	3	128 MB	0018332f-4b1d-420c-97e8-b7cee52a58f7.json	



## 6. Conclusión

El sistema implementado proporciona una solución robusta y escalable para la captura, procesamiento y almacenamiento de logs. La arquitectura diseñada permite un análisis eficiente tanto a largo plazo como para consultas rápidas. La implementación de Apache NiFi ha sido fundamental para mejorar la flexibilidad y eficiencia del sistema de Deusto Internet Services S.A. Esta herramienta permite procesar datos en tiempo real y adaptarse fácilmente a nuevos requerimientos, lo cual es crucial en un entorno tan dinámico. La combinación de HDFS y Elasticsearch ha demostrado ser una solución óptima para las necesidades de la empresa. HDFS proporciona un almacenamiento robusto y escalable, mientras que Elasticsearch ofrece capacidades de búsqueda rápida y análisis en tiempo real. Esta arquitectura permite a Deusto Internet Services S.A. mantener un historial completo de datos y realizar consultas ágiles.

## 7. IA

Se ha utilizado IA (perplexity) para:

- Configuración y entender el funcionamiento de los playbooks de ansible
- Expresiones regulares y scripts para transformar datos
- Flujo de procesamiento en NiFi
- Conexión NiFi, Elasticsearch y HDFS