TP - Monitoring des logs d'une application Spring Boot avec Filebeat, Logstash, Elasticsearch et Kibana

Réalisé par **Mr. Iyanou Eraste AKANDE**, Ingénieur des données Telecom à Synaptique Maghreb, Ingénieur Certifié Elasticsearch.

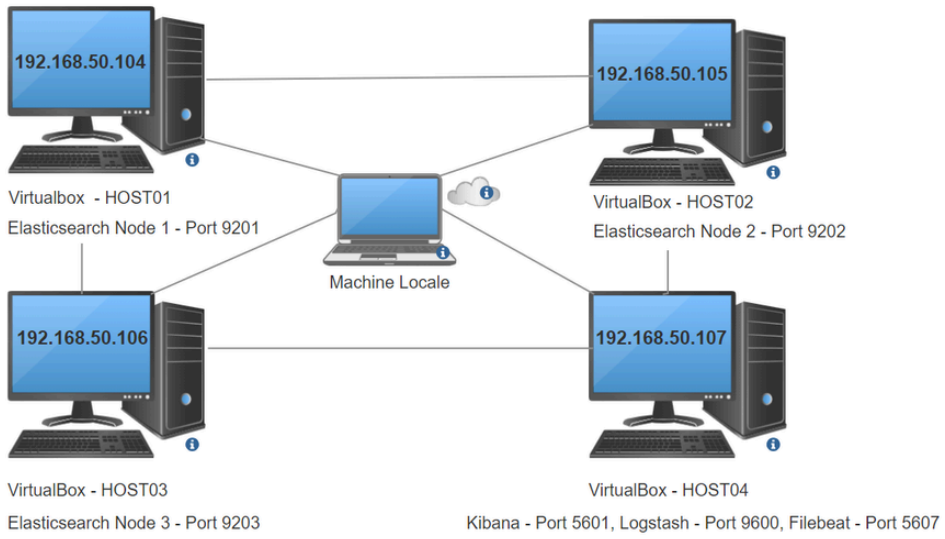
eraste.akande@gmail.com

Etape 1: Introduction à la pile ELK

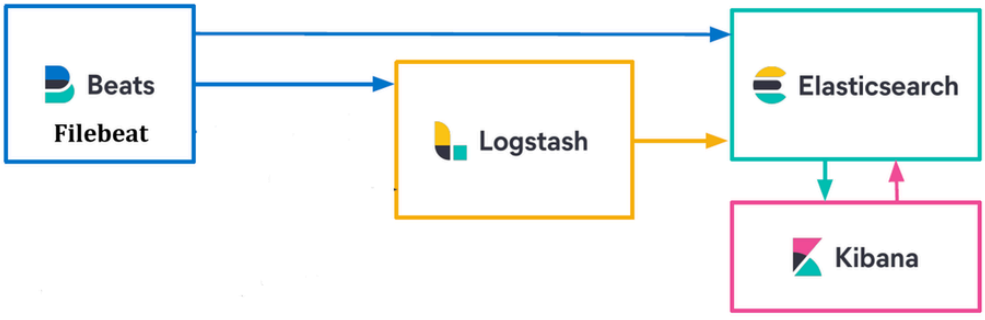
Présentation des composantes de la pile ELK, le rôle de chaque composante ainsi que les différents cas d'utilisation → (PPT Introduction to Elastic Stack).

Etape 2: Architecture

Les machines virtuelles



Les applications



Etape 3: Installation et configuration des machines virtuelles

- Téléchargez et installez la version de Virtual Box compatible à votre système d'exploitation sur le site https://www.virtualbox.org/wiki/Downloads.
- Dans certains cas pour les utilisateurs Windows, vous devriez installer au préalable Visual Studio C++ Redistributable à partir du site https://learn.microsoft.com/en-us/cpp/windows/latest-supported-vc-redist?view=msvc-170.
- Téléchargez et installez la version de Vagrant compatible à votre système d'exploitation à partir du site https://developer.hashicorp.com/vagrant/install?product_intent=vagrant.
- Téléchargez le fichier nommé Vagrantfile depuis ce répertoire GitHub https://github.com/iyanou/elastic-training-lab.
- Placez le fichier Vagrantfile dans un dossier et exécutez ce qui suit depuis cet emplacement

```
cd $vagrant_file_repository
vagrant up
```

- Se connecter sur chaque machine et vérifier qu'on arrive a faire un ping vers les autres adresses IP

```
ping -c 10 HOST01
ping -c 10 HOST02
ping -c 10 HOST03
ping -c 10 HOST04
```

Etape 4: Installation et configuration des applications

1. **Les noeuds Elasticsearch**

HOST01

- Se connecter au HOST01 depuis votre machine en utilisant Vagrant

```
cd $vagrant_file_repository
vagrant ssh HOST01
```

- Création du dossier d'installation

```
sudo mkdir -p /opt/training
```

- Téléchargement de Elasticsearch 8.13.0

```
cd /opt/training
sudo wget https://artifacts.elastic.co/downloads/elasticsearch/elasticsearch-8.13.0-linux-x86_64.tar.gz
```

- Décompression de Elasticsearch 8.13.0

```
sudo tar -xzf elasticsearch-8.13.0-linux-x86_64.tar.gz
sudo rm elasticsearch-8.13.0-linux-x86_64.tar.gz
```

- Modifier le fichier de configuration `elasticsearch.yml` avec le contenu du fichier `es_node01.yml` situé dans le répertoire GitHub https://github.com/iyanou/elastic-training-lab.

```
cd /opt/training/elasticsearch-8.13.0
sudo rm config/elasticsearch.yml
```

```
sudo nano config/elasticsearch.yml
```

- Mettre a jour le paramètre `vm.max_map_count`

```
sudo grep vm.max_map_count /etc/sysctl.conf

grep 'vm.max_map_count=262144' /etc/sysctl.conf || sudo echo 'vm.max_map_count=262144'
| sudo tee -a /etc/sysctl.conf

sudo sysctl -p
```

- Donner les permissions nécessaires a l'utilisateur sur le dossier elasticsearch

```
sudo chown -R vagrant:vagrant /opt/training/elasticsearch-8.13.0
```

- Générer les certificats de connexion elasticsearch en utilisant les fichiers contenus dans le répertoire GitHub https://github.com/iyanou/elastic-training-lab/certificates.

```
sudo mkdir -p config/certificates/ca
sudo mkdir -p config/certificates/node01
sudo nano config/certificates/node01/node01.key
sudo nano config/certificates/node01/node01.crt
sudo nano config/certificates/ca/ca.crt
```

Voici comment tous les certificats ont été généré en utilisant le fichier `instances.yml` sur le répertoire GitHub https://github.com/iyanou/elastic-training-lab/.

```
cd /opt/training/elasticsearch-8.13.0
sudo mkdir config/certificates
sudo nano config/certificates/instances.yml
sudo bin/elasticsearch-certutil ca --silent --pem -out config/certificates/ca.zip
sudo apt install unzip
sudo unzip -o config/certificates/ca.zip -d config/certificates

sudo bin/elasticsearch-certutil cert --silent --pem -out config/certificates/bundle.zip
--in config/certificates/instances.yml --ca-cert config/certificates/ca/ca.crt --ca-key
config/certificates/ca/ca.key

sudo unzip -o config/certificates/bundle.zip -d config/certificates
sudo rm config/certificates/bundle.zip
sudo rm config/certificates/ca.zip
```

- Autoriser les ports sur le pare-feu

```
sudo ufw enable
sudo ufw allow 9201/tcp
sudo ufw allow 9301/tcp
sudo ufw status
```

- Démarrer le nœud Elasticsearch comme un daemon et vérifier les logs

```
./bin/elasticsearch -d -p pid
```

Pour arrêter Elasticsearch, exécutez `pkill -F pid`.

HOST02

- Se connecter au HOST02 depuis votre machine en utilisant Vagrant

```
cd $vagrant_file_repository
vagrant ssh HOST02
```

- Création du dossier d'installation

```
sudo mkdir -p /opt/training
```

- Téléchargement de Elasticsearch 8.13.0

```
cd /opt/training
sudo wget https://artifacts.elastic.co/downloads/elasticsearch/elasticsearch-8.13.0-linux-x86_64.tar.gz
```

- Décompression de Elasticsearch 8.13.0

```
sudo tar -xzf elasticsearch-8.13.0-linux-x86_64.tar.gz
sudo rm elasticsearch-8.13.0-linux-x86_64.tar.gz
```

- Modifier le fichier de configuration `elasticsearch.yml` avec le contenu du fichier `es_node02.yml` situé dans le répertoire GitHub https://github.com/iyanou/elastic-training-lab.

```
cd /opt/training/elasticsearch-8.13.0
sudo rm config/elasticsearch.yml
sudo nano config/elasticsearch.yml
```

- Mettre a jour le paramètre `vm.max_map_count`

```
sudo grep vm.max_map_count /etc/sysctl.conf

grep 'vm.max_map_count=262144' /etc/sysctl.conf || sudo echo 'vm.max_map_count=262144'
| sudo tee -a /etc/sysctl.conf

sudo sysctl -p
```

- Donner les permissions nécessaires a l'utilisateur sur le dossier elasticsearch

```
sudo chown -R vagrant:vagrant /opt/training/elasticsearch-8.13.0
```

- Générer les certificats de connexion elasticsearch en utilisant les fichiers contenus dans le répertoire GitHub https://github.com/iyanou/elastic-training-lab/certificates .

```
sudo mkdir -p config/certificates/ca
sudo mkdir -p config/certificates/node02
sudo nano config/certificates/node02/node02.key
sudo nano config/certificates/node02/node02.crt
sudo nano config/certificates/ca/ca.crt
```

- Autoriser les ports sur le pare-feu

```
sudo ufw enable
sudo ufw allow 9202/tcp
sudo ufw allow 9302/tcp
sudo ufw status
```

- Démarrer le nœud Elasticsearch comme un daemon et vérifier les logs

```
./bin/elasticsearch -d -p pid
```

HOST03

- Se connecter au HOST03 depuis votre machine en utilisant Vagrant

```
cd $vagrant_file_repository
vagrant ssh HOST03
```

- Création du dossier d'installation

```
sudo mkdir -p /opt/training
```

- Téléchargement de Elasticsearch 8.13.0

```
cd /opt/training
sudo wget https://artifacts.elastic.co/downloads/elasticsearch/elasticsearch-8.13.0-linux-x86_64.tar.gz
```

- Décompression de Elasticsearch 8.13.0

```
sudo tar -xzf elasticsearch-8.13.0-linux-x86_64.tar.gz
sudo rm elasticsearch-8.13.0-linux-x86_64.tar.gz
```

- Modifier le fichier de configuration `elasticsearch.yml` avec le contenu du fichier `es_node03.yml` situé dans le répertoire GitHub https://github.com/iyanou/elastic-training-lab.

```
cd /opt/training/elasticsearch-8.13.0
sudo rm config/elasticsearch.yml
sudo nano config/elasticsearch.yml
```

- Mettre a jour le paramètre `vm.max_map_count`

```
sudo grep vm.max_map_count /etc/sysctl.conf

grep 'vm.max_map_count=262144' /etc/sysctl.conf || sudo echo 'vm.max_map_count=262144'
| sudo tee -a /etc/sysctl.conf

sudo sysctl -p
```

- Donner les permissions nécessaires a l'utilisateur sur le dossier elasticsearch

```
sudo chown -R vagrant:vagrant /opt/training/elasticsearch-8.13.0
```

- Générer les certificats de connexion elasticsearch en utilisant les fichiers contenus dans le répertoire GitHub https://github.com/iyanou/elastic-training-lab/certificates .

```
sudo mkdir -p config/certificates/ca
sudo mkdir -p config/certificates/node03
sudo nano config/certificates/node03/node03.key
sudo nano config/certificates/node03/node03.crt
sudo nano config/certificates/ca/ca.crt
```

- Autoriser les ports sur le pare-feu

```
sudo ufw enable
sudo ufw allow 9203/tcp
sudo ufw allow 9303/tcp
sudo ufw status
```

- Démarrer le nœud Elasticsearch comme un daemon et vérifier les logs

```
./bin/elasticsearch -d -p pid
```

Connexion au Cluster Elasticsearch

- Générer le mot de passe de l'utilisateur `elastic` et `kibana_system` en utilisant l'un des hosts et sauvegarder ces mots de passe

```
cd /opt/training/elasticsearch-8.13.0
bin/elasticsearch-reset-password -u elastic
bin/elasticsearch-reset-password -u kibana_system
```

- Sur votre machine local, aller sur votre navigateur et accéder au lien https://192.168.50.104:9201/_cluster/health?pretty pour vérifier que le cluster est en bon état (statut “green“).
- Sur votre machine local, aller sur votre navigateur et accéder au lien https://192.168.50.104:9201/_cat/nodes?pretty pour vérifier que les 3 nœuds ont rejoint le cluster.

Au lieu d’aller sur le navigateur, on peut faire les mêmes vérifications depuis l’un des hosts avec les commandes suivantes :

```
curl --cacert /opt/training/elasticsearch-8.13.0/config/certificates/ca/ca.crt
-u elastic:$elastic_password -XGET 'https://192.168.50.104:9201/_cluster/health?pretty'

curl --cacert /opt/training/elasticsearch-8.13.0/config/certificates/ca/ca.crt
-u elastic:$elastic_password -XGET 'https://192.168.50.104:9201/_cat/nodes?pretty'
```

2. **Le serveur Kibana**

La configuration de Kibana se fera sur le HOST04

- Se connecter au HOST04 depuis votre machine en utilisant Vagrant

```
cd $vagrant_file_repository
vagrant ssh HOST04
```

- Création du dossier d’installation

```
sudo mkdir -p /opt/training
```

- Téléchargement de Kibana 8.13.0

```
cd /opt/training
sudo wget https://artifacts.elastic.co/downloads/kibana/kibana-8.13.0-amd64.deb
```

- Décompression de Kibana 8.13.0

```
sudo dpkg -i kibana-8.13.0-amd64.deb
sudo rm kibana-8.13.0-amd64.deb
```

- Modifier le fichier de configuration `kibana.yml` avec le contenu du fichier `kibana.yml` situé dans le répertoire GitHub https://github.com/iyanou/elastic-training-lab. Ne pas oublier de mettre a jour le mot de passe du user `kibana_system.`

```
sudo rm /etc/kibana/kibana.yml
sudo nano /etc/kibana/kibana.yml
```

- Déposer le certificat d’autorité de Elasticsearch sur le HOST04. Copier le certificat depuis le répertoire GitHub https://github.com/iyanou/elastic-training-lab/certificates.

```
sudo mkdir -p /opt/training/ca
sudo nano /opt/training/ca/ca.crt
```

- Autoriser les ports sur le pare-feu

```
sudo ufw enable
sudo ufw allow 5601/tcp
sudo ufw status
```

- Démarrer Kibana

```
sudo /bin/systemctl daemon-reload
sudo systemctl enable kibana.service
sudo systemctl start kibana.service
sudo systemctl status kibana.service
```

> Pour arrêter le service Kibana, exécutez `sudo systemctl stop kibana.service`.

- Afficher les logs du service Kibana

```
sudo journalctl --unit=kibana.service -n 100 --no-pager
```

- Sur votre machine aller sur l'interface de Kibana http://192.168.50.107:5601 et connecter vous avec le user `elastic` et son mot de passe généré précédemment.

3. **L'ETL Logstash**

> La configuration de Logstash se fera sur le HOST04

- Se connecter au HOST04 depuis votre machine en utilisant Vagrant

```
cd $vagrant_file_repository
vagrant ssh HOST04
```

- Téléchargement de Logstash 8.13.0

```
cd /opt/training
sudo wget https://artifacts.elastic.co/downloads/logstash/logstash-8.13.0-amd64.deb
```

- Décompression de Logstash 8.13.0

```
sudo dpkg -i logstash-8.13.0-amd64.deb
sudo rm logstash-8.13.0-amd64.deb
```

- Modifier le fichier de configuration `logstash.yml` avec le contenu du fichier `logstash.yml` situé dans le répertoire GitHub https://github.com/iyanou/elastic-training-lab.

```
sudo rm /etc/logstash/logstash.yml
sudo nano /etc/logstash/logstash.yml
```

- Autoriser les ports sur le pare-feu

```
sudo ufw enable
sudo ufw allow 9600/tcp
sudo ufw allow 5044/tcp
sudo ufw status
```

- Donner les permissions nécessaires à Logstash

```
sudo chmod 755 /usr/share/logstash/data
sudo chown -R logstash:logstash /usr/share/logstash/data
```

- Démarrer Logstash

```
sudo /bin/systemctl daemon-reload
sudo systemctl enable logstash.service
sudo systemctl start logstash.service
sudo systemctl status logstash.service
```

Pour arrêter le service Logstash, exécuter `sudo systemctl stop logstash.service`.

- Afficher les logs du service Logstash

```
sudo journalctl --unit=logstash.service -n 100 --no-pager
```

- Aller sur l'interface de Logstash sur votre navigateur avec le lien http://192.168.50.107:9600/?pretty pour se rassurer que Logstash est démarré.

4. **L'agent Filebeat**

La configuration de Filebeat se fera sur le HOST04

- Se connecter au HOST04 depuis votre machine en utilisant Vagrant

```
cd $vagrant_file_repository
vagrant ssh HOST04
```

- Téléchargement de Filebeat 8.13.0

```
cd /opt/training
sudo wget https://artifacts.elastic.co/downloads/beats/filebeat/filebeat-8.13.0-amd64.deb
```

- Décompression de Filebeat 8.13.0

```
sudo dpkg -i filebeat-8.13.0-amd64.deb
sudo rm filebeat-8.13.0-amd64.deb
```

- Autoriser les ports sur le pare-feu

```
sudo ufw enable
sudo ufw allow 5067/tcp
sudo ufw status
```

- Démarrer Filebeat

```
sudo /bin/systemctl daemon-reload
sudo systemctl enable filebeat.service
sudo systemctl start filebeat.service
sudo systemctl status filebeat.service
```

Pour arrêter le service Filebeat, exécuter `sudo systemctl stop logstash.service`.

- Afficher les logs du service Filebeat

```
sudo journalctl --unit=filebeat.service -n 100 --no-pager
```

Etape 5: Introduction à Filebeat

Explication du fonctionnement de Filebeat et présentation des différentes options et paramètres de configuration → (PPT Starting with Filebeat).

Etape 6: Introduction à Logstash

Explication du fonctionnement de Logstash et présentation des différentes options et paramètres de configuration des pipelines Logstash → (PPT Starting with Logstash).

Etape 7: Introduction à Elasticsearch (Mapping et Analysers)

Explication de quelques concepts importants liés à Elasticsearch et à son fonctionnement → (PPT Starting with Elasticsearch).

Etape 8: Collecte des logs Spring Boot : De Filebeat vers Logstash et transfert vers Elasticsearch

Ce LAB se déroulera sur le HOST04.

- Se connecter au HOST04 depuis votre machine en utilisant Vagrant

```
cd $vagrant_file_repository
vagrant ssh HOST04
```

- Créer le fichier `spring-boot-app-logs.json` en utilisant le fichier `spring-boot-app-logs.json` situé dans le répertoire GitHub https://github.com/iyanou/elastic-training-lab.

```
sudo mkdir -p /opt/training/app/logs
sudo nano /opt/training/app/logs/spring-boot-app-logs.json
```

- Créer le pipeline Logstash pour le traitement des données venant de Filebeat. Utiliser le fichier `filebeat-pipeline.conf` situé dans le répertoire GitHub https://github.com/iyanou/elastic-training-lab.

```
 sudo nano /etc/logstash/conf.d/filebeat-pipeline.conf
```

- Redémarrer le service Logstash

```
sudo systemctl restart logstash.service
```

- Mettre a jour le index template de filebeat en utilisant le fichier filebeat-index-template.txt situé dans le répertoire GitHub https://github.com/iyanou/elastic-training-lab.
- Configuration de `filebeat.yml` en utilisant le fichier `filebeat.yml` situé dans le répertoire GitHub https://github.com/iyanou/elastic-training-lab.

```
sudo rm /etc/filebeat/filebeat.yml
```

```
sudo nano /etc/filebeat/filebeat.yml
```

- Supprimer le registre de Filebeat

```
sudo rm -r /var/lib/filebeat/registry
sudo rm /var/lib/filebeat/meta.json
udo rm /var/lib/filebeat/filebeat.lock
```

- Redémarrer le service Filebeat

```
sudo systemctl restart filebeat.service
```

- Afficher les logs du service Filebeat

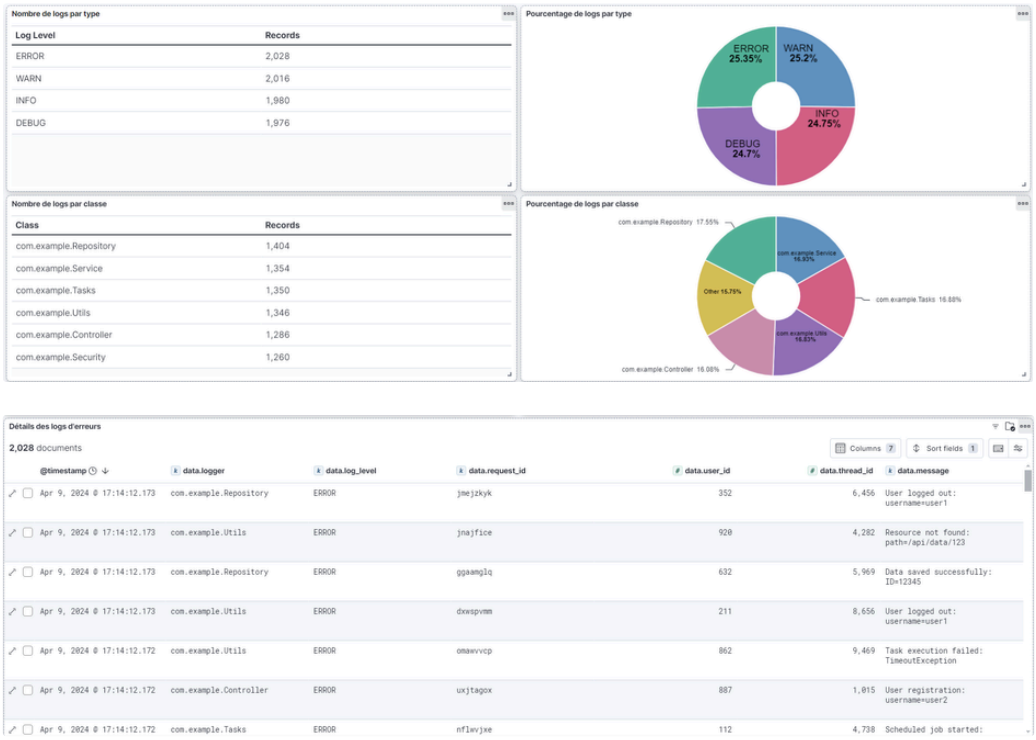
```
sudo journalctl --unit=filebeat.service -n 100 --no-pager
```

- Afficher les logs du service Logstash

```
sudo journalctl --unit=logstash.service -n 100 --no-pager
```

- Aller sur Kibana, choisir le data view nommé "**filebeat**" et aller dans la rubrique "**Discover**" pour vérifier que les logs ont été envoyés sur le cluster Elasticsearch.
- Tester sur Discover que l'analyser `filebeat_analyser` fonctionne comme voulu

Etape 9: Visualisation des Logs et création de Dashboards avec Kibana



Tester que le système est automatique

Sur le HOST04,

- Créer un nouveau fichier `spring-boot-app-logs-test.json` en utilisant le fichier `spring-boot-app-logs-test.json` situé dans le répertoire GitHub https://github.com/iyanou/elastic-training-lab.

```
sudo nano /opt/training/app/logs/spring-boot-app-logs-test.json
```

- Visualiser que les logs ont mis le Dashboard à jour.

Introduction to Elastic Stack

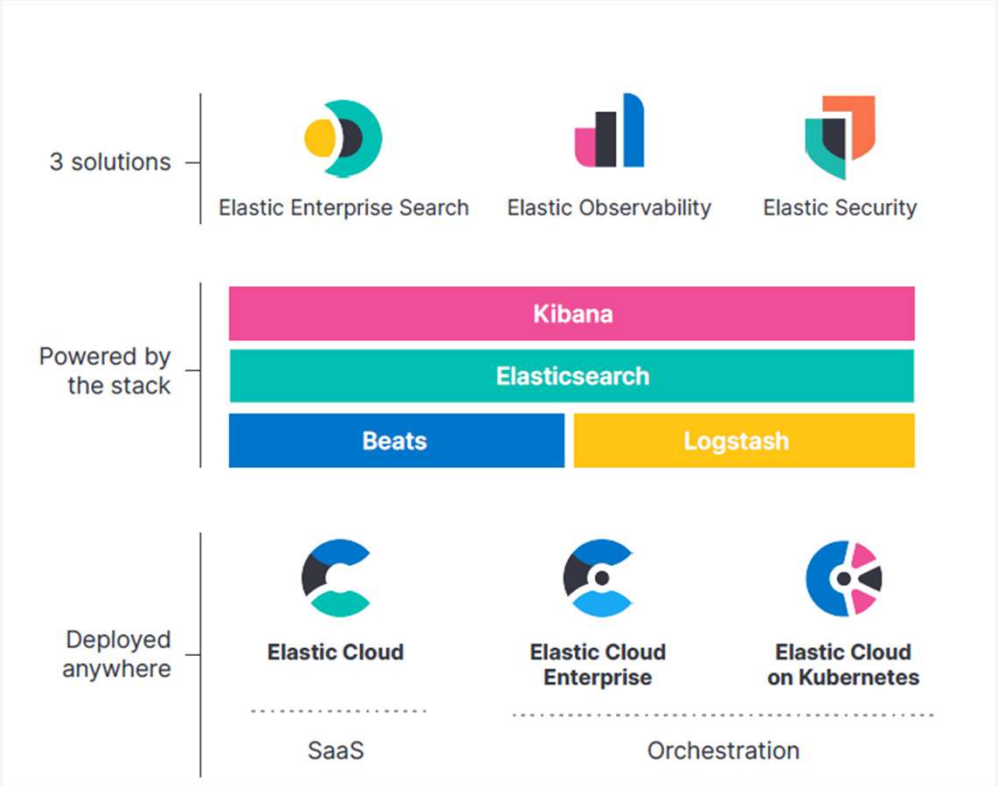
Mr. Iyanou Eraste AKANDE
Elastic Certified Engineer
Data Engineer at Synaptique Maghreb

The Beginning

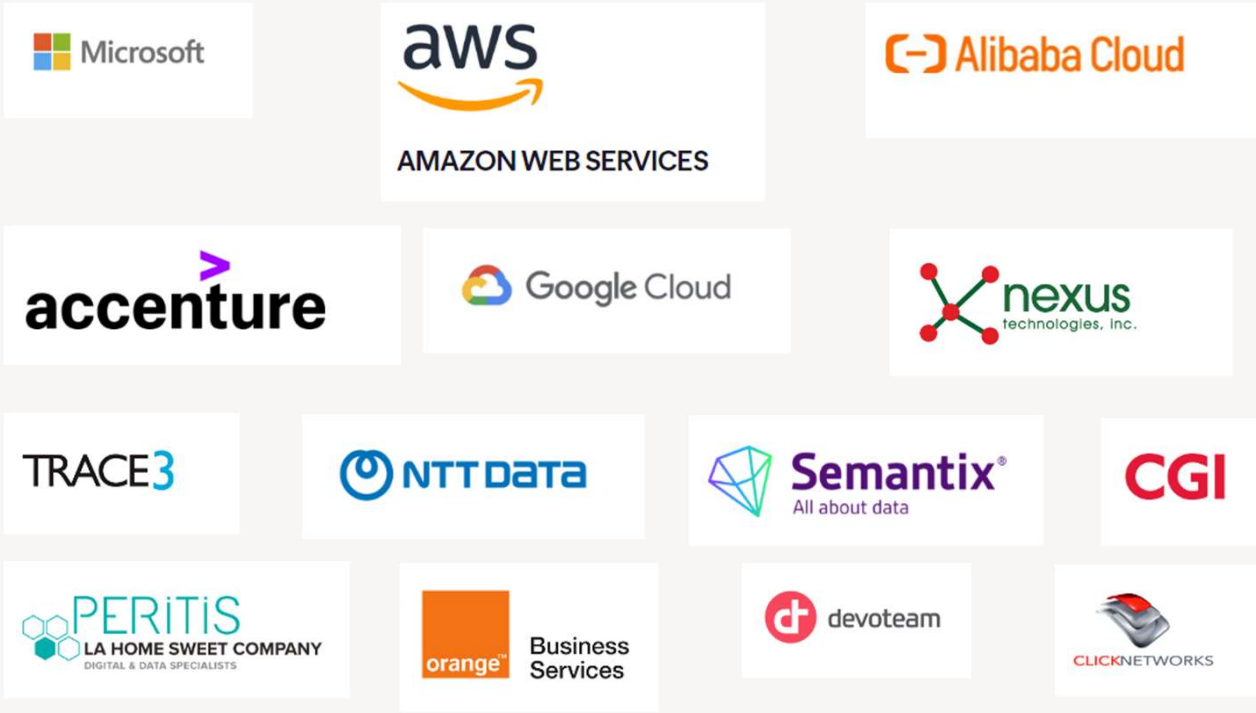
Lucene	Compass	Elasticsearch
1999 Doug Cutting Search Engine in Java	2004 Shay Banon Search Product in Java, scalability included, built on Lucene	2010 Shay Banon Search product built on Compass concepts, distributed and integrated with other language

Elastic Stack Components

- Elasticsearch : NoSQL Database
- Kibana : Analytics and Visualization platform
- Logstash : Server side data processing pipeline
- Beats : Data Shippers (Filebeat, Metricbeat, Heartbeat, Elastic Agents)



Elastic Partners



https://partners.elastic.co/findapartner/

Elastic Stack Use Cases

- **Log Management and Analysis**: One of the primary use cases for Elastic Stack is log management and analysis. Organizations use it to collect, parse, index, and analyze logs generated by various systems, applications, and services.
- **Security Information and Event Management (SIEM)**: Elastic Stack can be used as a SIEM solution to collect, analyze, and visualize security-related data such as logs, network traffic, and system events.
- **Real-time Monitoring and Alerting**: Elastic Stack can be utilized for real-time monitoring of infrastructure, applications, and services. By collecting metrics, logs, and events in real-time, organizations can set up alerts and notifications.
- **Application Performance Monitoring (APM)**: Elastic APM, an extension of the Elastic Stack, is used for monitoring the performance of applications and microservices.

Elastic Stack Use Cases

- **Business Analytics and Insights**: Elastic Stack can be leveraged for business analytics and data visualization purposes. By indexing and analyzing large volumes of data from different sources, organizations can gain valuable insights into customer behavior, market trends, and operational performance.

- **Search and Information Retrieval**: Elasticsearch supports full-text search, fuzzy search, geospatial search, and faceted navigation, making it suitable for building search-driven applications and platforms.

- **Data Exploration and Discovery**: Elastic Stack can be used for data exploration and discovery in various domains such as scientific research, e-commerce, and content management.

Elasticsearch

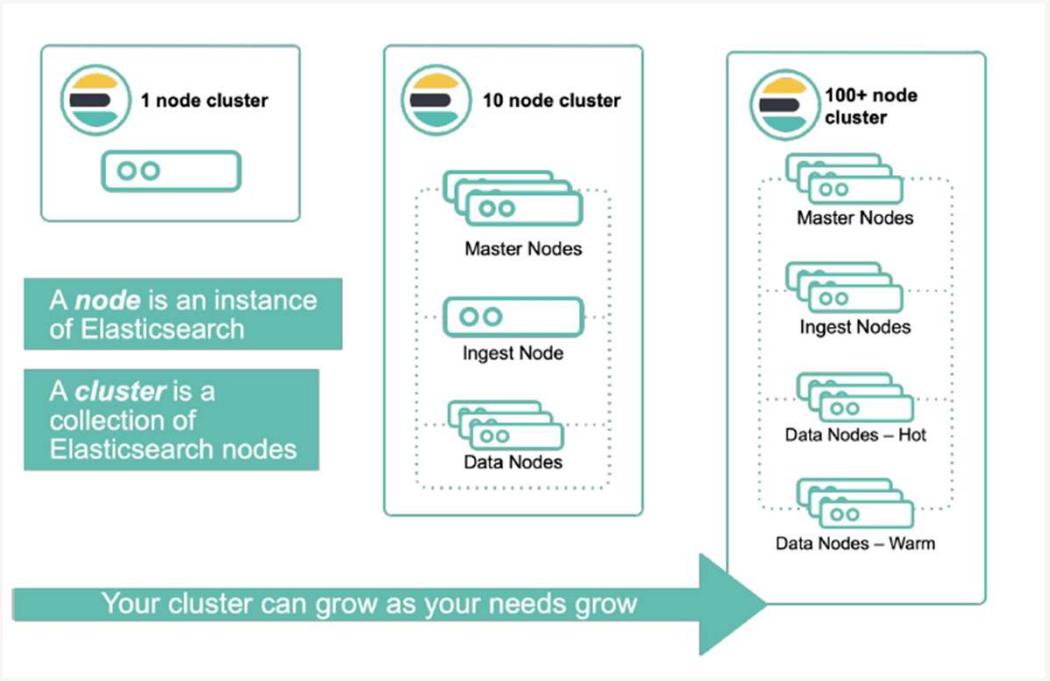
NoSQL Database

first_ame	name	gender	city	country
Ali	KOZNI	M	Rabat	Morocco
Achille	BADRE	F	Tetouan	Morocco


```
{
“first_name” : “Ali”,
“name” : “KOZNI”,
“gender” : “M”,
“city” : “Rabat”,
“country” : “Morocco”
},
{
“first_name” : “Achille”,
“name” : “BADRE”,
“gender” : “F”,
“city” : “Tetouan”,
“country” : “Morocco”
}
```

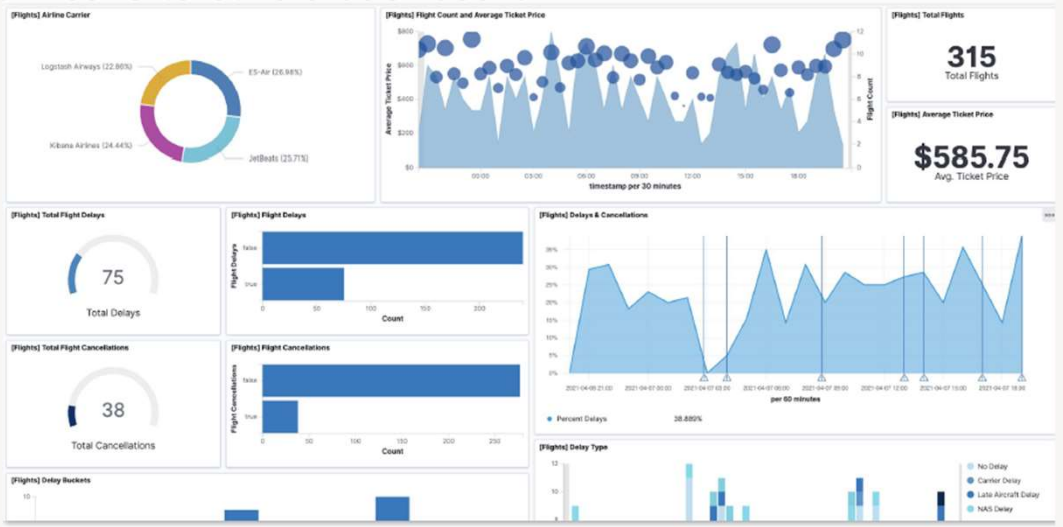
Elasticsearch

Distributed and Horizontally Scalable



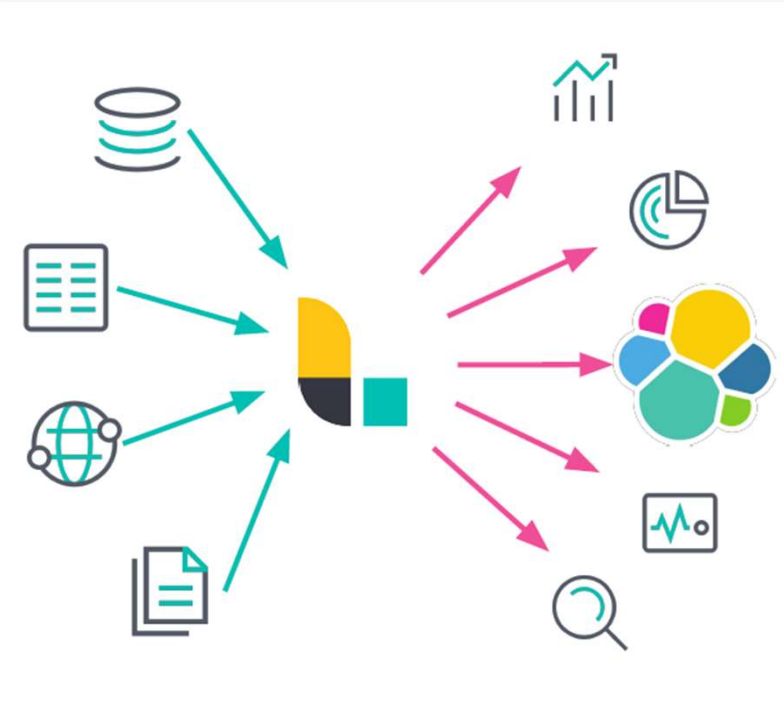
Kibana

Analytics and Visualization Platform



Logstash

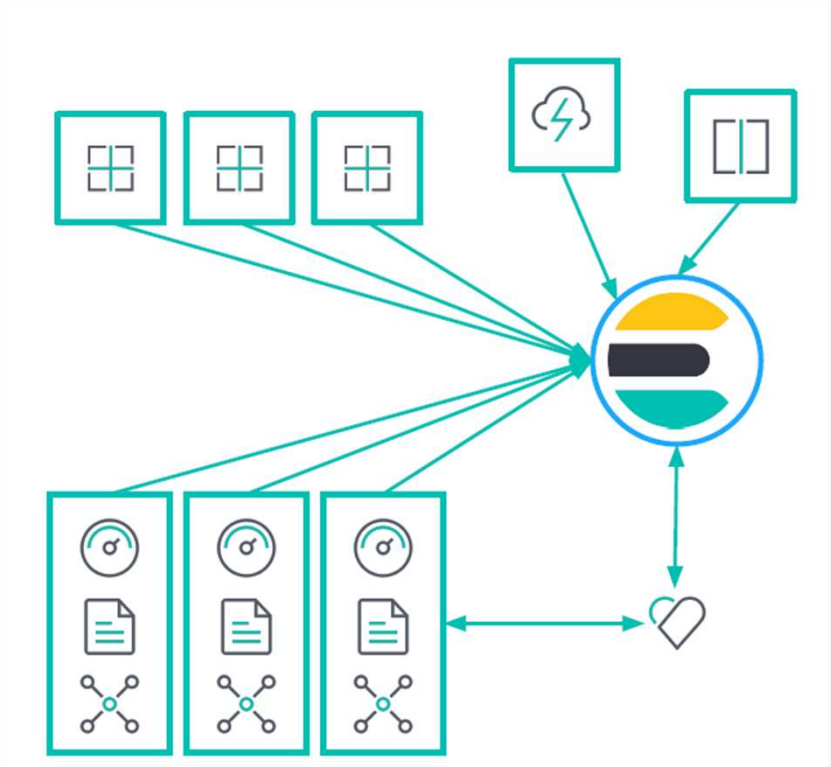
Data Processing Pipeline



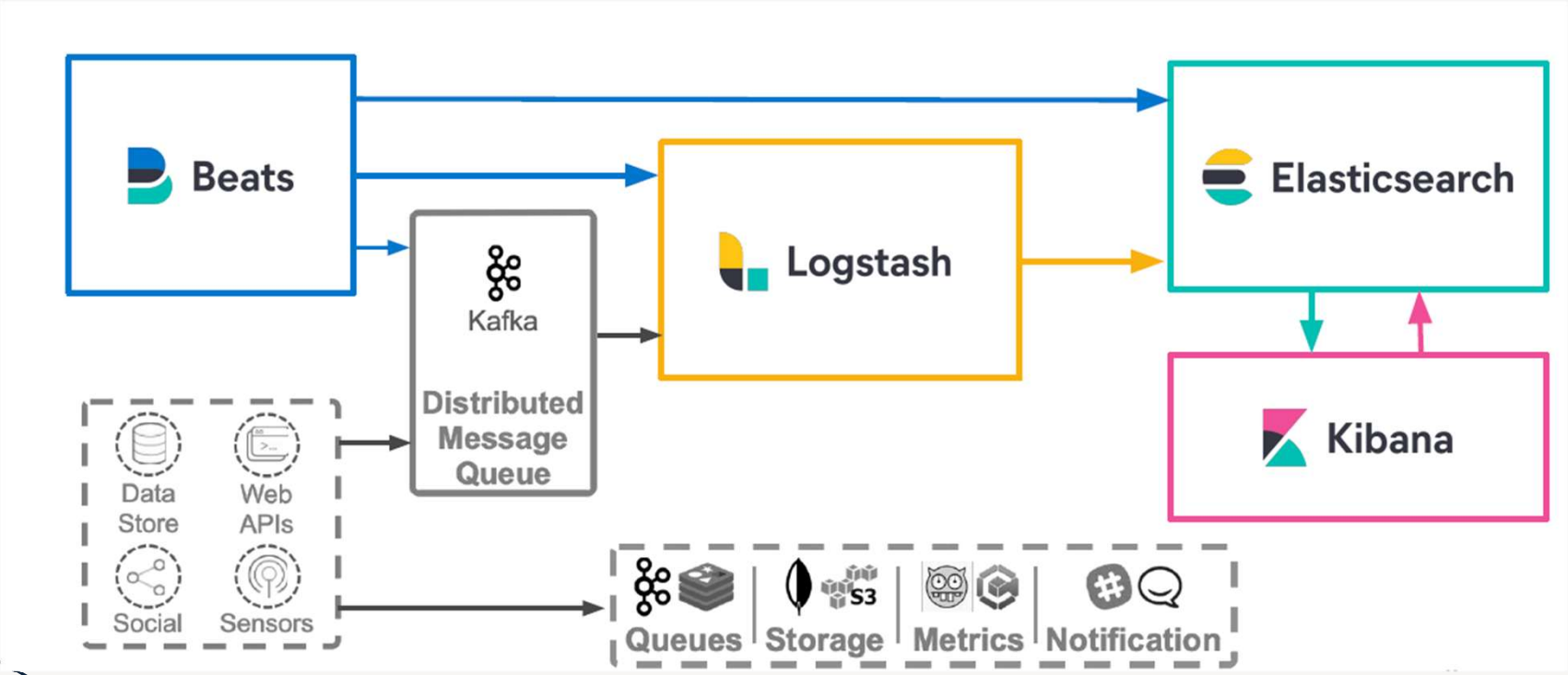
Beats

Open Source Data Shippers

Filebeat, Metricbeat, Heartbeat,
Packetbeat, Winlogbeat, etc.



Elastic Stack Architecture



Questions

Thank you

Mr. Iyanou Eraste AKANDE

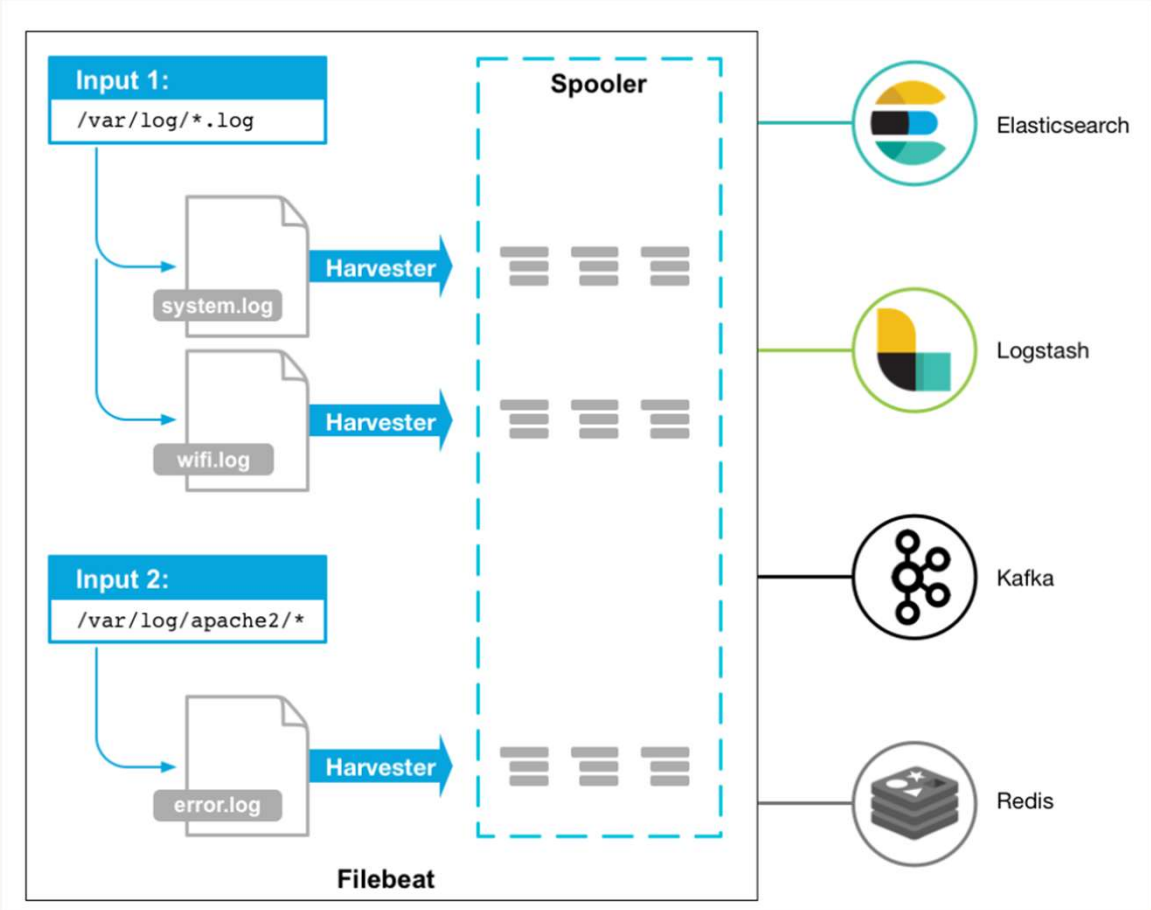
eraste.akande@gmail.com

Starting with Filebeat

Mr. Iyanou Eraste AKANDE
Elastic Certified Engineer
Data Engineer at Synaptique Maghreb

Filebeat

Architecture



Filebeat

Installation

- Download Filebeat and decompress it
  - https://artifacts.elastic.co/downloads/beats/filebeat/filebeat-8.12.2-windows-x86_64.zip
  - curl -L -O https://artifacts.elastic.co/downloads/beats/filebeat/filebeat-8.12.2-linux-x86_64.tar.gz && tar xzvf filebeat-8.12.2-linux-x86_64.tar.gz
- Setup Filebeat (cd filebeat_directory && filebeat setup -e)
- Start Filebeat (cd filebeat_directory && filebeat -e)

Filebeat

Filebeat.yml >> Input

Input Types

- AWS CloudWatch
- AWS S3
- Azure Event Hub
- Azure Blob Storage
- CEL
- Cloud Foundry
- CometD
- Container
- Entity Analytics
- filestream
- GCP Pub/Sub
- HTTP Endpoint
- HTTP JSON
- journald
- Kafka
- Log (deprecated in 7.16.0, use filestream)
- MQTT
- NetFlow
- Office 365 Management Activity API
- Redis
- Stdin
- Syslog
- TCP
- UDP
- Google Cloud Storage

Filebeat

Filebeat.yml >> Input

Filestream Input

https://www.elastic.co/guide/en/beats/filebeat/current/filebeat-input-filestream.html

```yaml
filebeat.inputs:
- type: filestream
  id: my-filestream-id
  paths:
    - /var/log/messages
    - /var/log/*.log
```

```yaml
filebeat.inputs:
- type: filestream (1)
  id: my-filestream-id
  paths:
    - /var/log/system.log
    - /var/log/wifi.log
- type: filestream (2)
  id: apache-filestream-id
  paths:
    - "/var/log/apache2/*"
  fields:
    apache: true
```

```yaml
filebeat.inputs:
- type: filestream
  ...
  parsers:
    - ndjson:
        target: ""
        message_key: msg
```

Filebeat

Filebeat.yml >> Input

Filestream Input

https://www.elastic.co/guide/en/beats/filebeat/current/filebeat-input-filestream.html

```
filebeat.inputs:
- type: filestream
  ...
  prospector.scanner.exclude_files: ['\.gz$']
  prospector.scanner.include_files: ['^/var/log/.*']
```

```
filebeat.inputs:
- type: filestream
  ...
  include_lines: ['^ERR', '^WARN']
  exclude_lines: ['^DBG']
```

Filebeat

Filebeat.yml >> Input

Filestream Input

https://www.elastic.co/guide/en/beats/filebeat/current/multiline-examples.html

Problem

```
[beat-logstash-some-name-832-2015.11.28] IndexNotFoundException[no such index]
    at
org.elasticsearch.cluster.metadata.IndexNameExpressionResolver$WildcardExpressionResolver.resolve(IndexNameExpressionResolver
.java:566)
    at org.elasticsearch.cluster.metadata.IndexNameExpressionResolver.concreteIndices(IndexNameExpressionResolver.java:133)
    at org.elasticsearch.cluster.metadata.IndexNameExpressionResolver.concreteIndices(IndexNameExpressionResolver.java:77)
    at org.elasticsearch.action.admin.indices.delete.TransportDeleteIndexAction.checkBlock(TransportDeleteIndexAction.java:75)
```

Solution

```
parsers:
- multiline:
    type: pattern
    pattern: '^\['
    negate: true
    match: after
```

Filebeat

Filebeat.yml >> Processors

Processors Types

https://www.elastic.co/guide/en/beats/filebeat/current/defining-processors.html

- add_docker_metadata
- add_fields
- add_host_metadata
- add_id
- add_locale
- append
- convert
- copy_fields
- decode_base64_field
- decode_csv_fields
- decode_duration
- decode_json_fields
- decode_xml
- decompress_gzip_field
- dissect
- drop_event
- drop_fields
- extract_array
- fingerprint
- include_fields
- move-fields
- rename
- replace
- script
- timestamp
- translate_sid

Filebeat

Filebeat.yml >> Processors

Processors Conditions

https://www.elastic.co/guide/en/beats/filebeat/current/defining-processors.html

- equals
- contains
- regexp
- range
- network
- has_fields
- or
- and
- not

Filebeat

Filebeat.yml >> Processors

Examples

```
processors:
  - add_fields:
      target: project
      fields:
        name: myproject
        id: '5747348851209524 59'
```

```
processors:
  - convert:
      fields:
        - {from: "src_ip", to: "source.ip", type: "ip"}
        - {from: "src_port", to: "source.port", type: "integer"}
      ignore_missing: true
      fail_on_error: false
```

Filebeat

Filebeat.yml >> Processors

Examples

```
processors:
  - rename:
      fields:
        - from: "a.g"
          to: "e.d"
      ignore_missing: false
      fail_on_error: true
```

```
- replace:
    fields:
      - field: "file.path"
        pattern: "/usr/"
        replacement: "/usr/local/"
    ignore_missing: false
    fail_on_error: true
```

```
processors:
  - drop_event:
      when:
        equals:
          log_status: "INFO"
```

Filebeat

Filebeat.yml >> Processors

Examples

Problem

"321 - App01 - WebServer is starting"
"321 - App01 - WebServer is up and running"
"321 - App01 - WebServer is scaling 2 pods"
"789 - App02 - Database is will be restarted in 5 minutes"
"789 - App02 - Database is up and running"
"789 - App02 - Database is refreshing tables"

Solution

```
processors:
  - dissect:
      tokenizer: '"%{service.pid|integer} - %{service.name} - %{service.status}"'
      field: "message"
      target_prefix: ""
```

Filebeat

Filebeat.yml >> Output

Output Types

- Elasticsearch Service
- Elasticsearch
- Logstash
- Kafka
- Redis
- File
- Console

Filebeat

Filebeat.yml >> Output

Elasticsearch Output

```
output.elasticsearch:
  hosts: ["https://localhost:9200"]
  index: "your_index_name"
  ssl.enabled: true  # Enable SSL/TLS
  ssl.certificate_authorities: ["/path/to/your/ca.crt"]
  username: "your_username"  # Elasticsearch username
  password: "your_password"  # Elasticsearch password
```

Logstash Output

```
output.logstash:
  hosts: ["localhost:5044"]
```

Filebeat

LAB

Tasks

You have a json log file named spring-boot-app-logs.json. Use filebeat to process the file and send it to your logstash instance listening on port 5044.

Questions

Thank you

Mr. Iyanou Eraste AKANDE

eraste.akande@gmail.com

Starting with Logstash

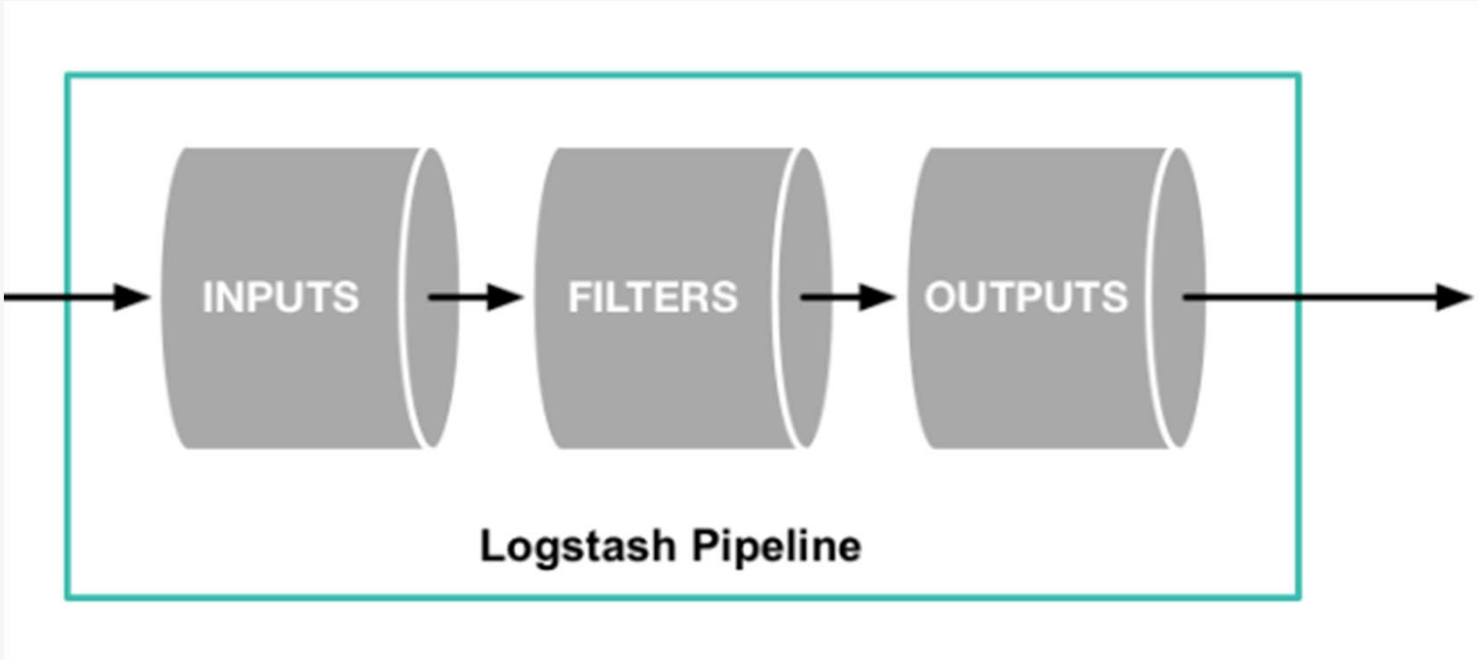
Mr. Iyanou Eraste AKANDE
Elastic Certified Engineer
Data Engineer at Synaptique Maghreb

Setup Logstash

- Download Logstash and decompress it (wget https://artifacts.elastic.co/downloads/logstash/logstash-8.12.2-linux-x86_64.tar.gz && tar -xvf logstash-8.12.2-linux-x86_64.tar.gz)
- Allow firewall on port 9600
- Start Logstash pipeline (“cd logstash_directory” && “bin/logstash –f config/configuration-file.conf”)

Configuration

Architecture



Configuration

Sample file

```
# Sample Logstash configuration for creating a simple
# Beats -> Logstash -> Elasticsearch pipeline.
input {
        beats {    port => 5044  }
}
filter {}
output {
        elasticsearch {
                hosts => ["http://localhost:9200"]
                index => "filebeat-%{+YYYY.MM.dd}"
                user => "elastic"
                password => "changeme"
         }
}
```

Configuration

Input

```
input {
  beats {
    port => 5044
  }
}
```

https://www.elastic.co/guide/en/logstash/current/input-plugins.html

Configuration

Input

```
input {

  file {
    path => ["/path/samples/input-file.csv"]
    start_position => "beginning"
    sincedb_path => "/path/log.sincedb"
    file_completed_action => "log"
    file_completed_log_path => "/path/log.processed"
    mode => read
  }

}
```

https://www.elastic.co/guide/en/logstash/current/input-plugins.html

Configuration

Filter

```
csv {
    source => "message"
    separator => ";"
    columns => ["field1", "field2", "field3", "field4"]
}

mutate {
  add_field => { "country" => "Morocco" }
}

mutate {
  convert => { "money" => "float" }
}

date {
      match => ["@timestamp", "yyyyMMdd HH:mm:ss"]
      timezone => "UTC"
      target => "@timestamp"
}

mutate {
  rename => {"Lieu de Naissance" => "birth_city"}
}
```

https://www.elastic.co/guide/en/logstash/current/filter-plugins.html

Configuration

Filter

```
mutate { remove_field => [ "message", "@version", "host", "path", "event"]}

if ([code] =~ /.+/ ) {
translate {
    source => "[code]"
    target => "[value]"
    dictionary_path => "/path/dicts/code.yml"
    exact => true
    override => true
    regex => false
  }
}

translate {
        source => "[code]"
        target => "[code]"
        override => true
        exact => true
        dictionary => {
            "0" => "Normal"
            "1" => "Free"
        }
}
```

https://www.elastic.co/guide/en/logstash/current/filter-plugins.html

Configuration

Filter

```
ruby {
    code => "
      event.set('salary', event.get('salary') / event.get('total_number'));
      event.set('id', '2024' + event.get('id'));
    "
}
```

```
# location : "region=Rabat,country=Morocco"
kv {
    source => "location"
    field_split => ","
    value_split => "="
}

if [loglevel] == "debug" {
    drop { }
}
```

https://www.elastic.co/guide/en/logstash/current/filter-plugins.html

Configuration

Filter

```
mutate {
    add_field => { "hostname" => "LENOVO" }
    #description : "The//name//of//the city"
    gsub => [
          "description", "//", " "
        ]
    uppercase => [ "fieldname" ]
    lowerercase => [ "fieldname" ]
    copy => { "source_field" => "dest_field" }
}

# message : "55.3.244.1 GET /index.html 15824 0.043"
grok {
  match => { "message" => "%{IP:client} %{WORD:method} %{URIPATHPARAM:request} %{NUMBER:bytes} %{NUMBER:duration}" }
}
```

https://www.elastic.co/guide/en/logstash/current/filter-plugins.html

Configuration

Output

```
elasticsearch {
    hosts                           => "server-address"
    user                            => "elastic"
    password                        => "changemee"
    cacert                          => "/path/cert/elk.crt"
    ssl                             => "true"
    ssl_certificate_verification    => "true"
    index                           => "logstash-test"
    action                          => "index"
}
```

https://www.elastic.co/guide/en/logstash/current/output-plugins.html

Configuration

Output

```
kafka {
    codec => json
    bootstrap_servers => "server_address:9092"
    topic_id => "topic-name"
    compression_type => "gzip"
}
```

https://www.elastic.co/guide/en/logstash/current/output-plugins.html

LAB

Tasks

You are receiving a spring boot app logs from filebeat on port 5044. Configure logstash to process these logs and send the results to elasticsearch.

1. Use json filter to parse the message and add a field named dataset with the value "Spring App"
2. Rename the field log_level to level
3. Remove the field event and format the field event_date with the date filter
4. Apply a gsub filter to replace "com.example." with a whitespace "" in the field logger
5. Define for the data stream "filebeat-8.13.0" an analyser named filebeat_analyser. This analyser will use the standard tokenizer and the lowercase and synonym token filters so that "failed" and "error" are synonyms.
6. Define an index template for the data stream "filebeat-8.13.0". Map these fields like this (logger → keyword, thread_id →keyword, user_id → keyword, request_id → keyword, @timestamp → date, event_date → date, message → text, level → keyword)
7. Apply the filebeat_analyser to the field message
8. Send the logs to your Elasticsearch Output

Questions

Thank you

Mr. Iyanou Eraste AKANDE

eraste.akande@gmail.com

Starting with Elasticsearch

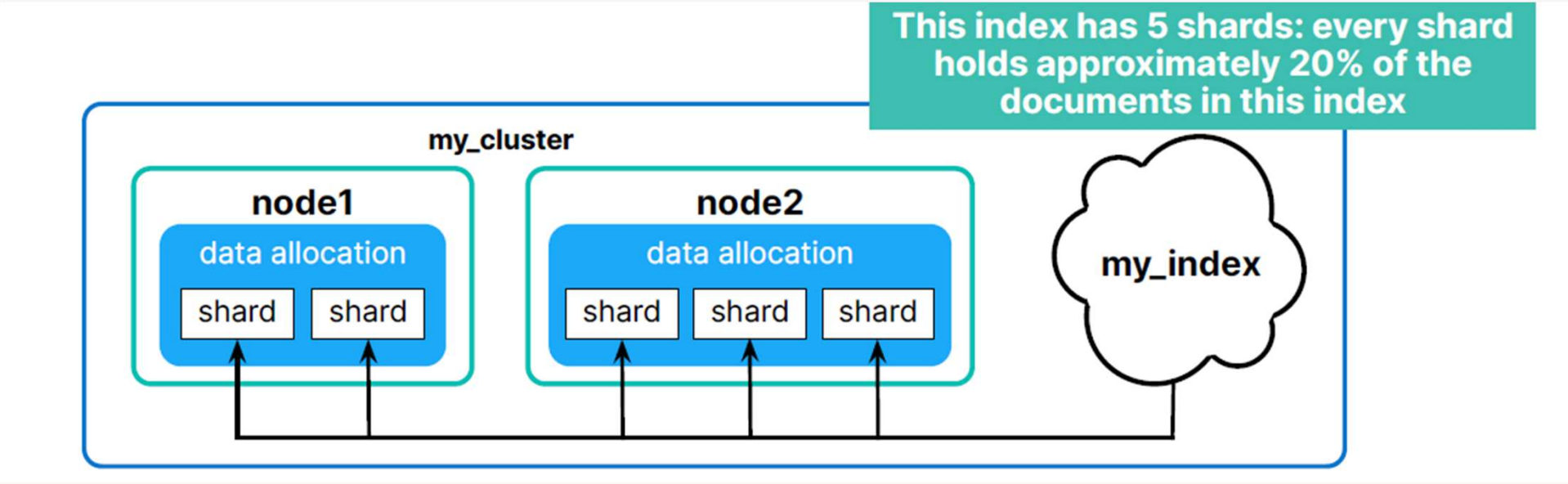
Mr. Iyanou Eraste AKANDE
Elastic Certified Engineer
Data Engineer at Synaptique Maghreb

Setup Elasticsearch

- Download Elasticsearch and decompress it (https://www.elastic.co/guide/en/elasticsearch/reference/current/install-elasticsearch.html)
- Make sure the user has write access on elasticsearch folders
- Allow firewall on port 9200
- Start Elasticsearch ("cd elasticsearch_directory" && "bin/elasticsearch")
- Copy username, password and Kibana enrollment token in a file
- Verify Elasticsearch Installation (https://localhost:9200, https://localhost:9200/_cluster/health?pretty)
- Run Elasticsearch as Service
  - ❑ Windows ("bin/elasticsearch-service.bat install" && "bin/elasticsearch-service.bat start")
  - ❑ Linux
    - ➢ Run as daemon ("bin/elasticsearch –d –p pid" to start and "pkill –F pid" to stop)
    - ➢ Use Systemctl (Define elasticsearch.service && "systemctl daemon-reload", "systemctl enable elasticsearch.service", "systemctl start elasticsearch.service")

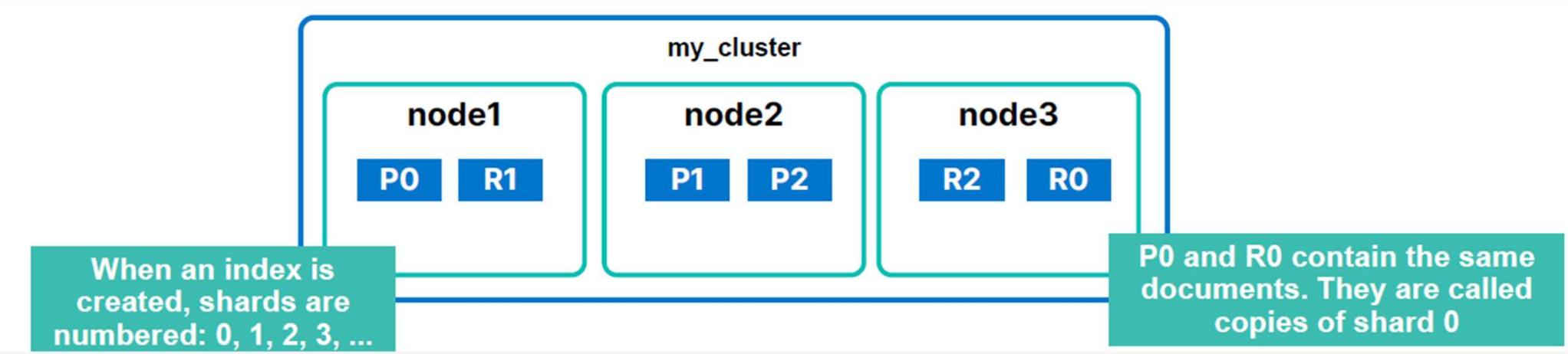
Architecture

Shards



Architecture

Primary Vs Replica

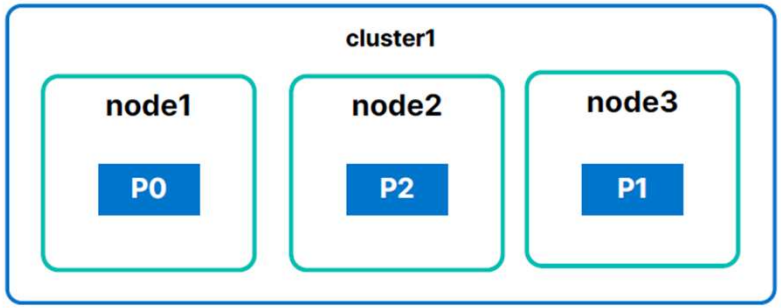


Architecture

Shards Configuration

request

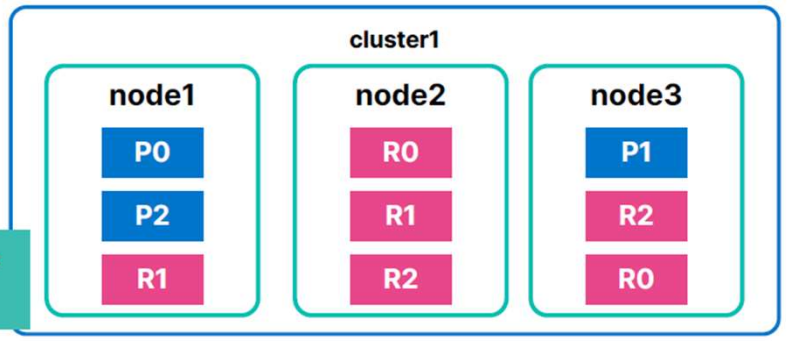
```
PUT my_new_index
{
  "settings": {
    "number_of_shards": 3
  }
}
```



request

```
PUT my_new_index/_settings
{
  "number_of_replicas": 2
}
```

3 primaries + 2 replica sets =
9 total shards



Mapping

Default Mapping

```
{
“first_name” : “Ali”,
“name” : “KOZNI”,
“gender” : “M”,
“city” : “Rabat”,
“country” : “Morocco”,
“age” : 25,
“company” : “INTEL”,
“starting_date” : “2022-04-23”
}
```

- Slow index speed
- Non optimized storage space
- Not improved searches

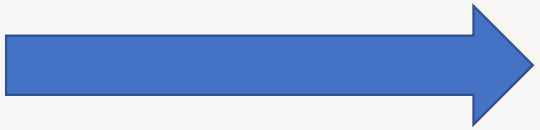
“first_name” : text
“first_name.keyword” : keyword
“name” : text
“name.keyword” : keyword
“gender” : text
“gender.keyword” : keyword
“city” : text
“city.keyword” : keyword
“country” : text
“country.keyword” : keyword
“age” : long
“company” : text
“company.keyword” : keyword
“starting_date” : text
“starting_date.keyword” : keyword

Mapping

Explicit Mapping

```
PUT employees_data
{
  "mappings": {
    "properties": {
      "first_name": {
        "type": "keyword"
      },
      "age": {
        "type": "integer"
      }
      "starting_date": {
        "type": "date“,
        “format”: “yyyy-MM-dd”
      }
    }
  }
}
```

- **Fast index speed**
- **Optimized storage space**
- **Improved searches**



“first_name” : keyword
“name” : keyword

“gender” : keyword

“city” : keyword
“country” : keyword

“age” : integer

“company” : keyword
“starting_date” : date

Mapping

Data Types

- **text**: for full-text (analyzed) strings
- **keyword**: for exact value strings and aggregations
- **date** and **date_nanos**: string formatted as dates, or numeric dates
- integer types: **byte**, **short**, **integer**, **long**
- floating-point numbers: **float**, **double**, **half_float**
- **boolean**

Mapping

Properties

- **Mappings** are Elasticsearch's data schema
- Mappings are defined **per index**
- If you do not define an explicit mapping, Elasticsearch will **dynamically** map the fields in your documents
- You cannot change the mapping of a field after the index has been created, but you can add new fields to a mapping
- Creating a custom mapping will produce savings in search and index speed, as well as memory and storage requirements

Component Template

- Reusable building blocks that can contain settings, mappings or aliases

- Reused across multiple templates

```
PUT _component_template/component_template1
{
  "template": {
    "mappings": {
      "properties": {
        "@timestamp": {
          "type": "date"
        }
      }
    }
  }
}
```

Index Template

- Multiple indices with the same settings and mappings

- Index template can contain
  - **settings**
  - **mappings**
  - **aliases**
  - **component templates**

```
PUT _index_template/template_1
{
  "index_patterns": ["spring-*"],
  "template": {
    "settings": {
      "number_of_shards": 1
    },
    "mappings": {
      "properties": {
        "host_name": {
          "type": "keyword"
        }
      }
    }
  },
  "priority": 500,
  "composed_of": ["component_template1"],
}
```

Analysers

How to analyse text fields ?

Example: A quick brown fox jumps over the lazy dog

- Character filters : Transform the stream by adding, removing, or changing characters (Ex: mapping, html_strip)
- Tokenizers : Breaking a text down into smaller chunks (Ex: whitespace)
- Token filters : A *token filter* receives the token stream and may add, remove, or change tokens (Ex: lowercase, stop, synonym)

Built in Analysers

https://www.elastic.co/guide/en/elasticsearch/reference/current/analysis-analyzers.html

Built in Tokenizers

https://www.elastic.co/guide/en/elasticsearch/reference/current/analysis-tokenizers.html

Built in Token filters

https://www.elastic.co/guide/en/elasticsearch/reference/current/analysis-tokenfilters.html

Analysers

Create a custom analyser

```
PUT /my-index-000001
{
  "settings": {
    "analysis": {
      "analyzer": {
        "my_analyzer": {
          "tokenizer": "whitespace",
          "filter": [ "stop" ]
        }
      }
    }
  }
}
```

```
PUT trim_example
{
  "settings": {
    "analysis": {
      "analyzer": {
        "keyword_trim": {
          "tokenizer": "keyword",
          "filter": [ "trim" ]
        }
      }
    }
  }
}
```

Analysers

Apply an analyser to a field

```
PUT my-index-000001
{
  "mappings": {
    "properties": {
      "title": {
        "type": "text",
        "analyzer": "whitespace"
      }
    }
  }
}
```

Query DSL

Search

```
GET index_name/_search
{
    "query": {
       "match": {
          "title": "lifestyle"
        }
    }
}
```

```
GET index_name/_search
{
    "query": {
       "term": {
          "name": "Alain"
        }
    }
}
```

https://www.elastic.co/guide/en/elasticsearch/reference/current/query-dsl.html

Query DSL

Aggregations

```
GET index_name/_search
{
  "aggs": {
    "my-agg-name": {
      "terms": {
        "field": "my-field"
      }
    }
  }
}
```

```
GET index_name/_search
{
  "aggs": {
    "my-second-agg-name": {
      "avg": {
        "field": "my-other-field"
      }
    }
  }
}
```

https://www.elastic.co/guide/en/elasticsearch/reference/current/search-aggregations.html

Reindex API

Local

```
POST _reindex
{
    "source": {
        "index": " source_index_name",
        "query": {...}
    },
    "dest": {
        "index": " destination_index_name"
    }
}
```

Reindex API

Remote

```
POST _reindex
{
    "source": {
        "remote": {
            "host": "http://otherhost:9200",
            "username": "user",
            "password": "pass"
        },
        "index": "remote_index",
    },
    "dest": {
        "index": "local_index"
    }
}
```

Update By Query

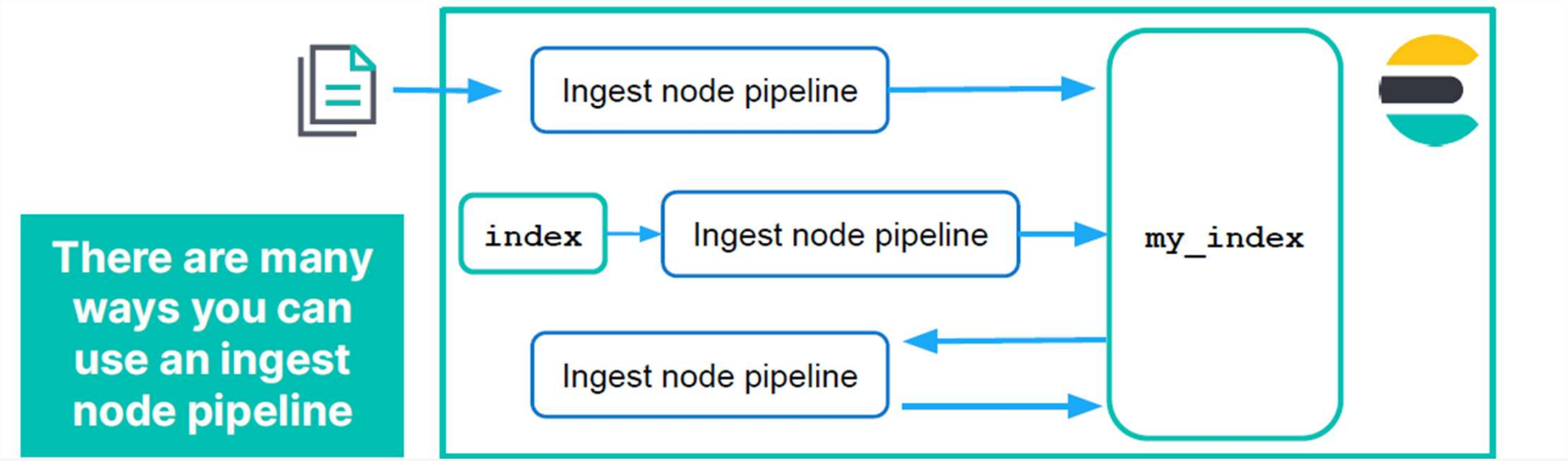
```
POST index_name/_update_by_query
{
    "query": {
        …..
    },
    "script": {
          "source": "ctx._source. name = Ali"
    }
}
```

Delete By Query

```
POST index_name/_delete_by_query
{
    "query": {
        "match": {
            “name": “Malton Jack"
        }
    }
}
```

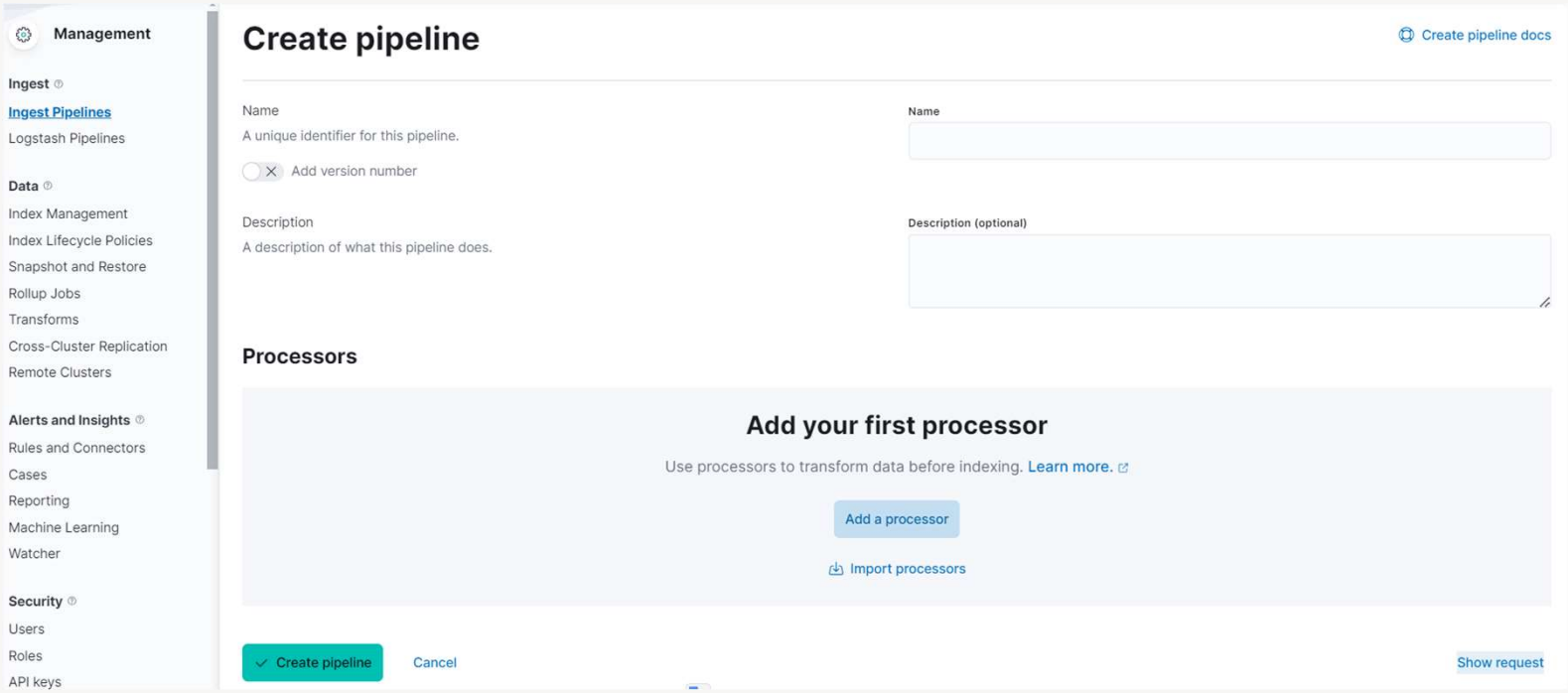
Ingest Node Pipelines

How it works ?



Ingest Node Pipelines

Pipeline Creation



Ingest Node Pipelines

Processors

Manipulate Fields

- set
- remove
- rename
- dot_expander
- etc

Manipulate Values

- split/join
- grok
- dissect
- gsub
- etc

Special Operations

- csv/json
- geoip
- user_agent
- script
- pipeline
- etc

Ingest Node Pipelines

Usage

- **POST** my_index/_update_by_query? **pipeline**=my_pipeline

- **PUT** my_index

```
{
    "settings": {
        "default_pipeline": "my_pipeline"
    }
}
```

- **POST** _reindex

```
{
    "source": {
        "index": "my_index"
    },
    "dest": {
         "index": "new_index",
         "pipeline": "my_pipeline"
}
```

Enrich Data

Explanation

stock		
id	product	price
1	rice	40
2	bag	30
3	hat	15
4	dog	200
5	desk	300
6	chair	150
7	table	250
8	computer	1000
9	phone	750
10	oil	60

provenance	
id	origin
1	Benin
2	Togo
3	Morocco
4	Congo
5	Mali
6	Niger
7	Soudan
8	Angola
9	Namibie
10	Rwanda

Enrich Data

Enrich Policy

```
PUT _enrich/policy/ product_mapper
{
    "match": {
        "indices": "provenance",
        "match_field": "id",
        "enrich_fields": ["country"]
    }
}
```

Enrich Data

Create Enrich Index

```
POST _enrich/policy/product_mapper/_execute
```

Enrich Data

Ingest Pipeline

```
PUT /_ingest/pipeline/product_mapper_pipeline
{
    "processors" : [
        {
            "enrich" : {
                "policy_name": "product_mapper",
                "field" : "id",
                "target_field": "country"
            }
        }
    ]
}
```

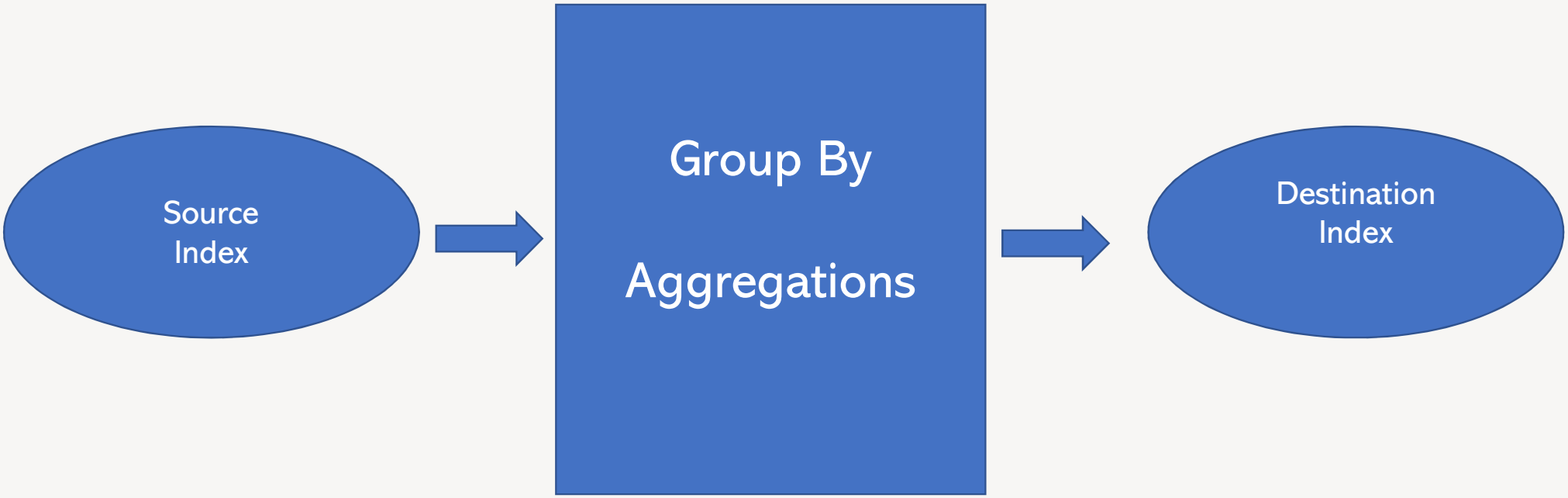
Enrich Data

Use the pipeline

```
POST stock/_update_by_query?pipeline=product_mapper_pipeline
```

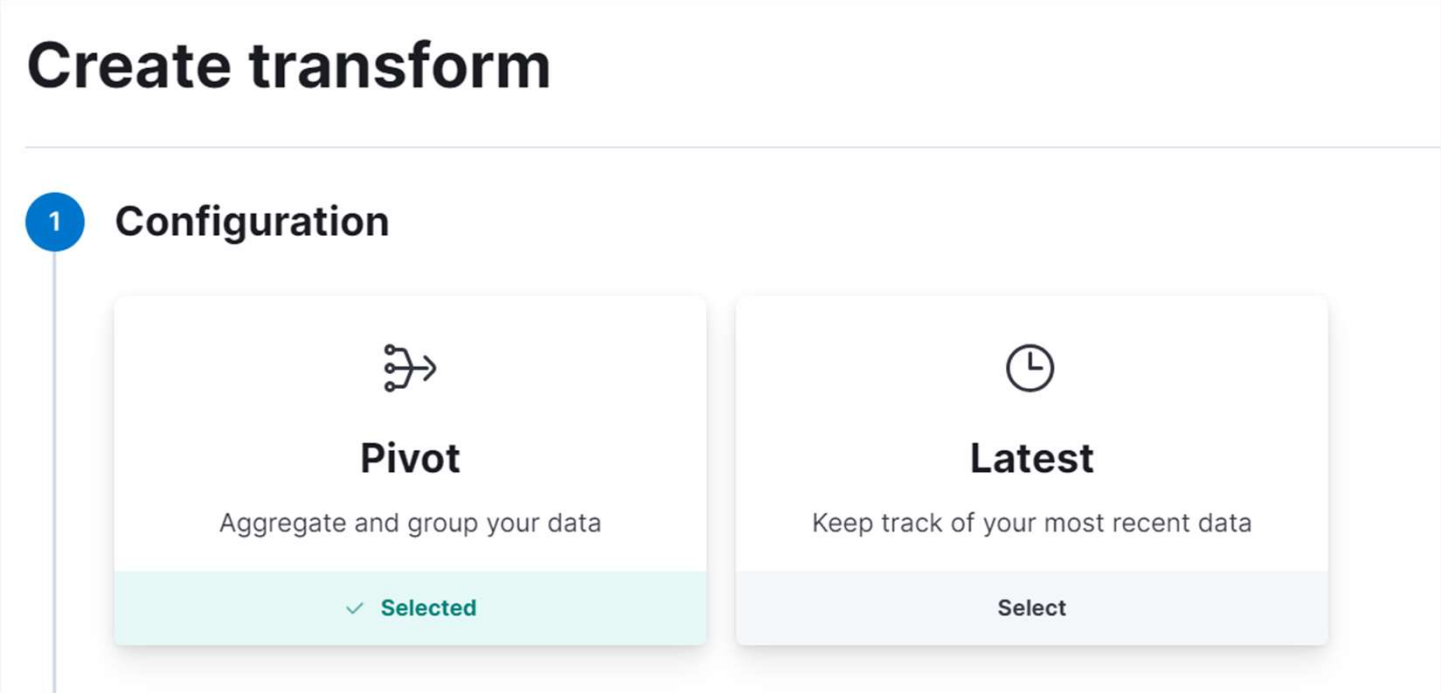
Transforms

Explanation



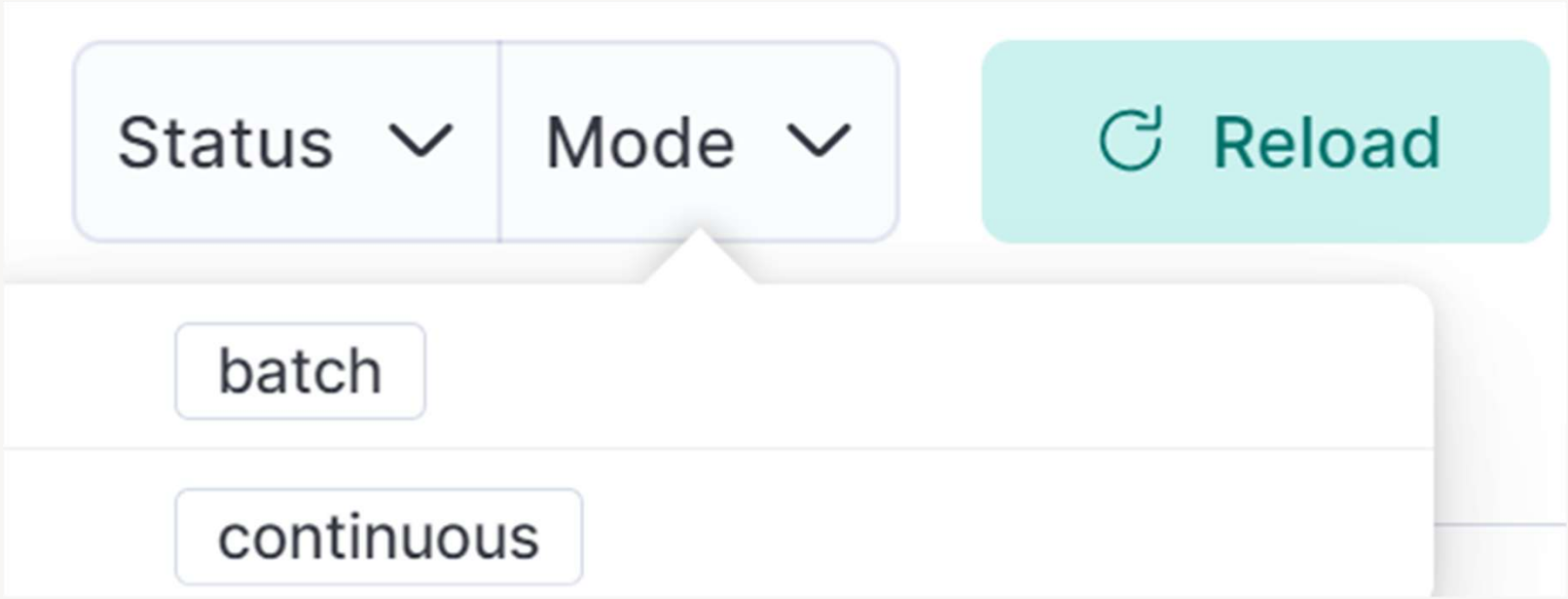
Transforms

Types



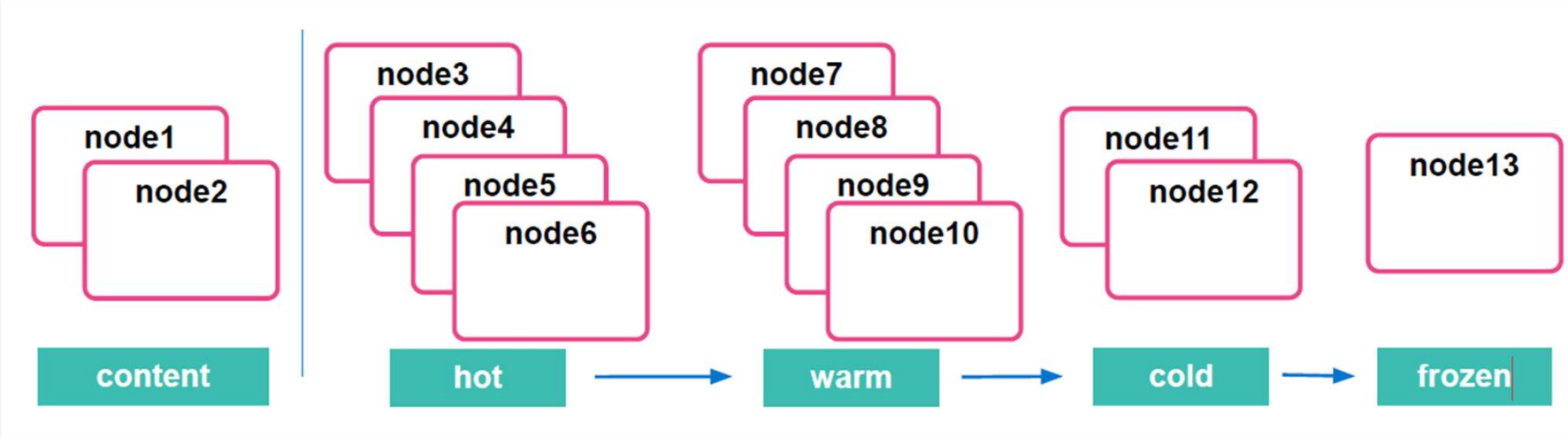
Transforms

Mode



Index Lifecycle Management

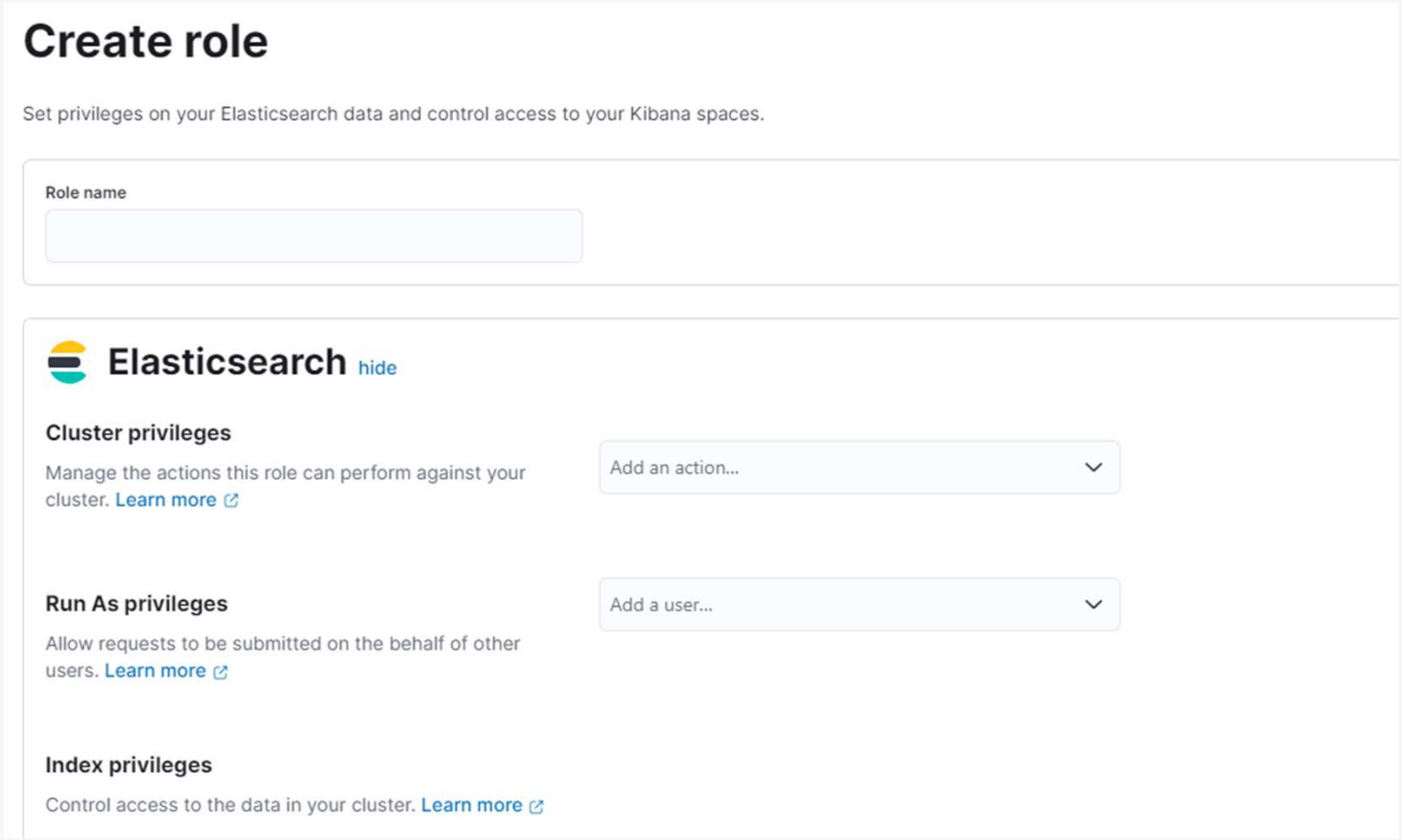
Data Tiers



node.roles: ["data_warm", "data_content"]

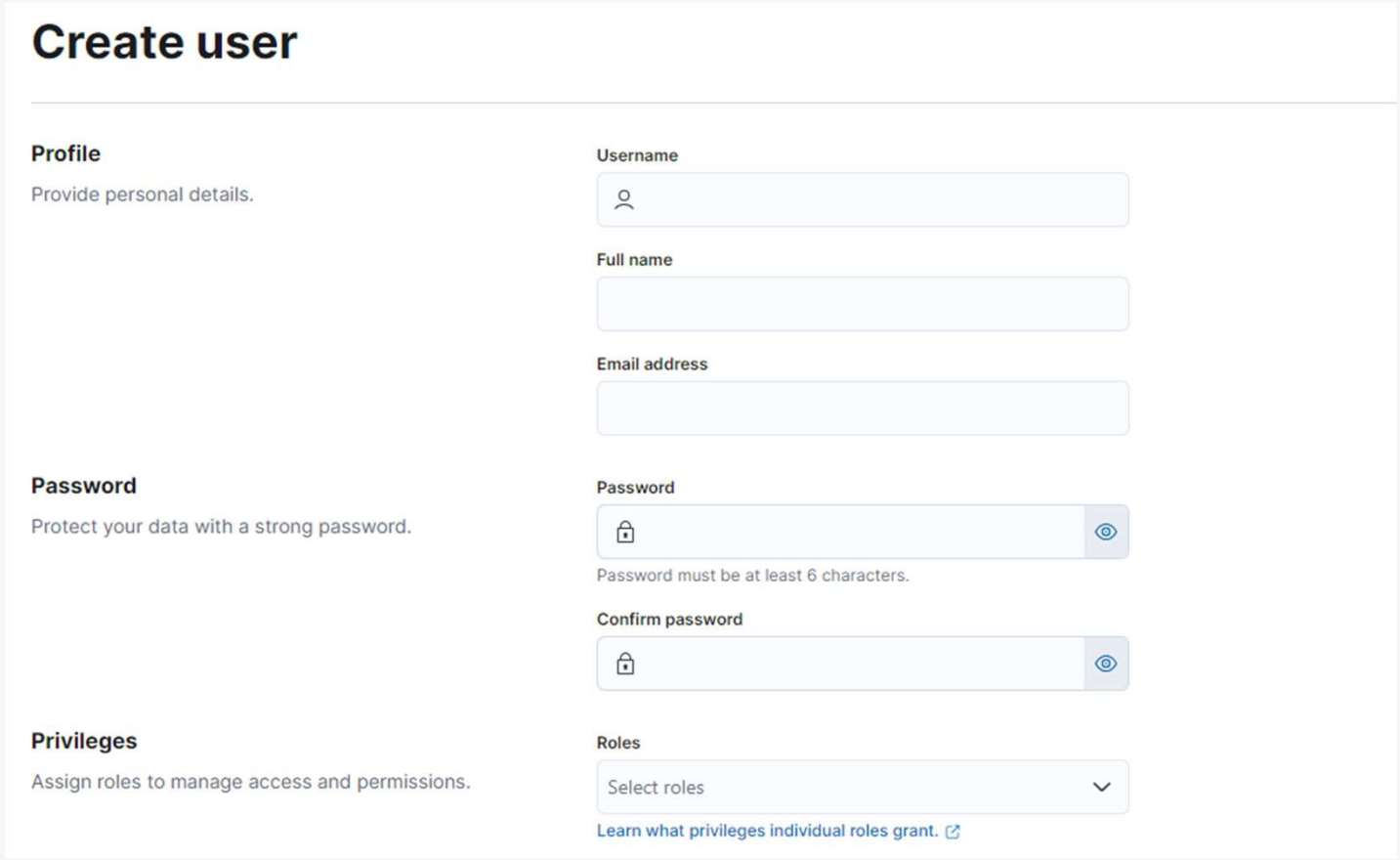
Security

Role



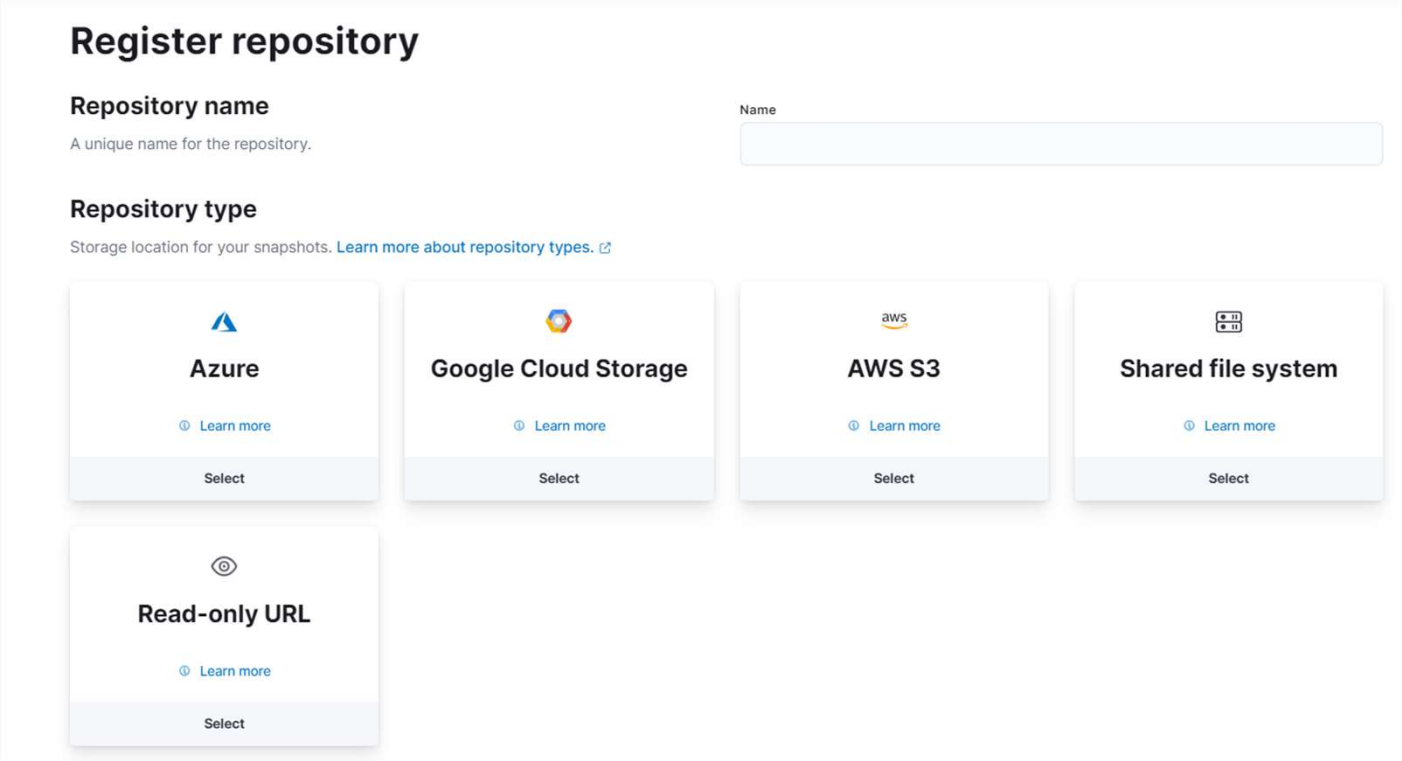
Security

User



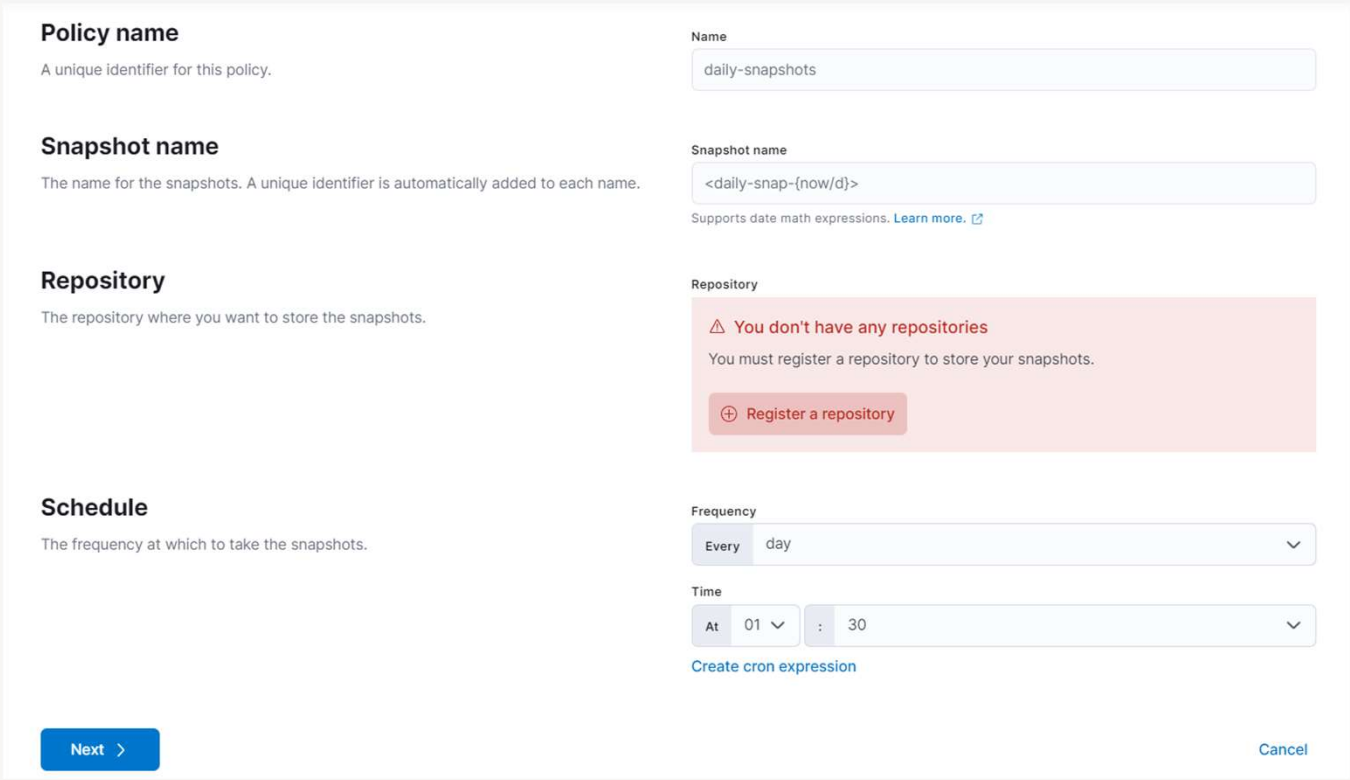
Snapshots & Restore

Repository



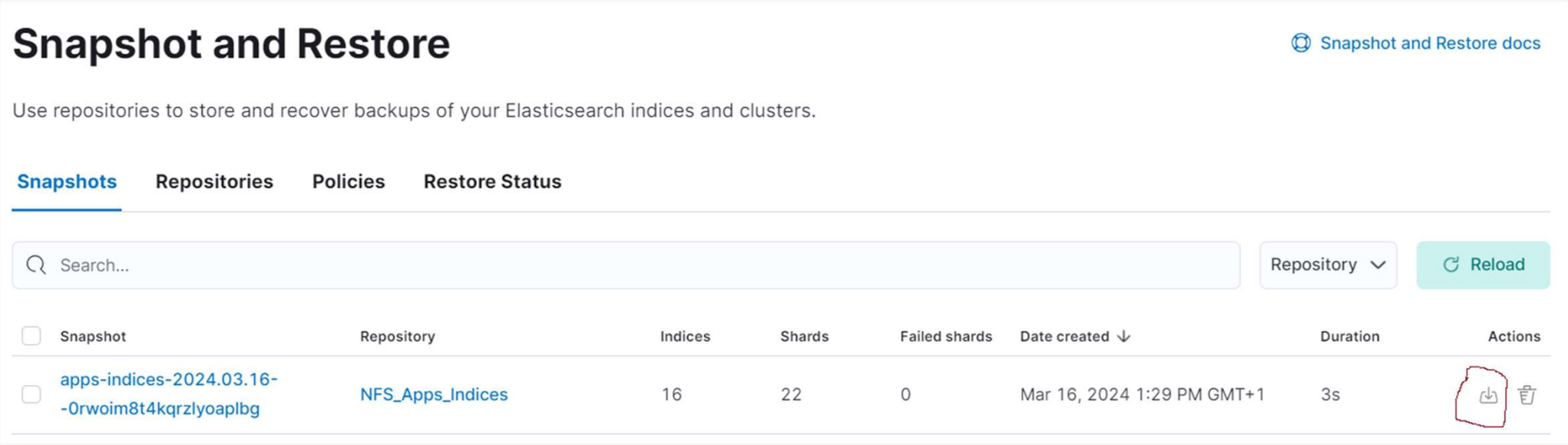
Snapshots & Restore

Policy



Snapshots & Restore

Restore

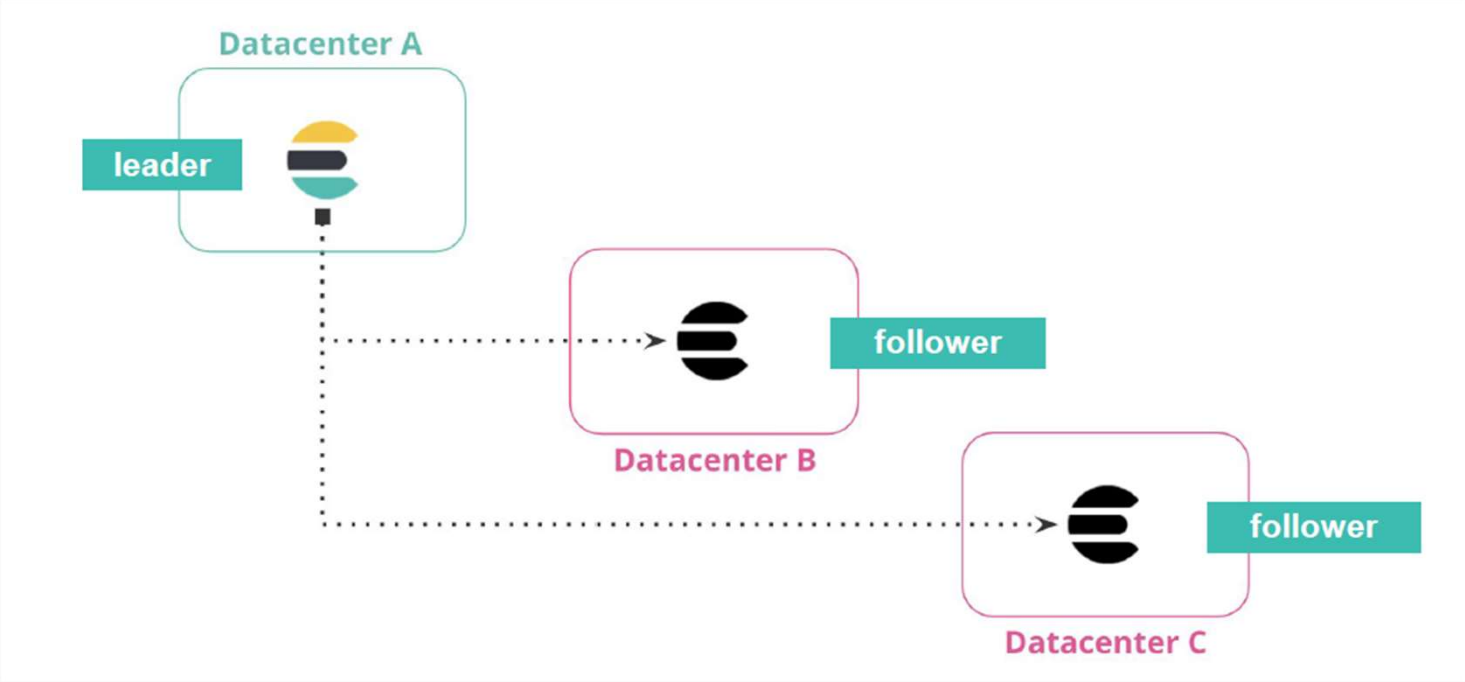


Troubleshooting

Command	Response
GET _tasks	the running tasks on the cluster
GET _cluster/pending_tasks	any cluster-level changes that have not yet executed
GET _nodes/hot_threads	threads using high CPU volume and executing for a long time
GET _cat/shards	the statistics of shards
GET _cat/health	the health of the cluster
GET _cluster/allocation/explain	explain the shards allocation

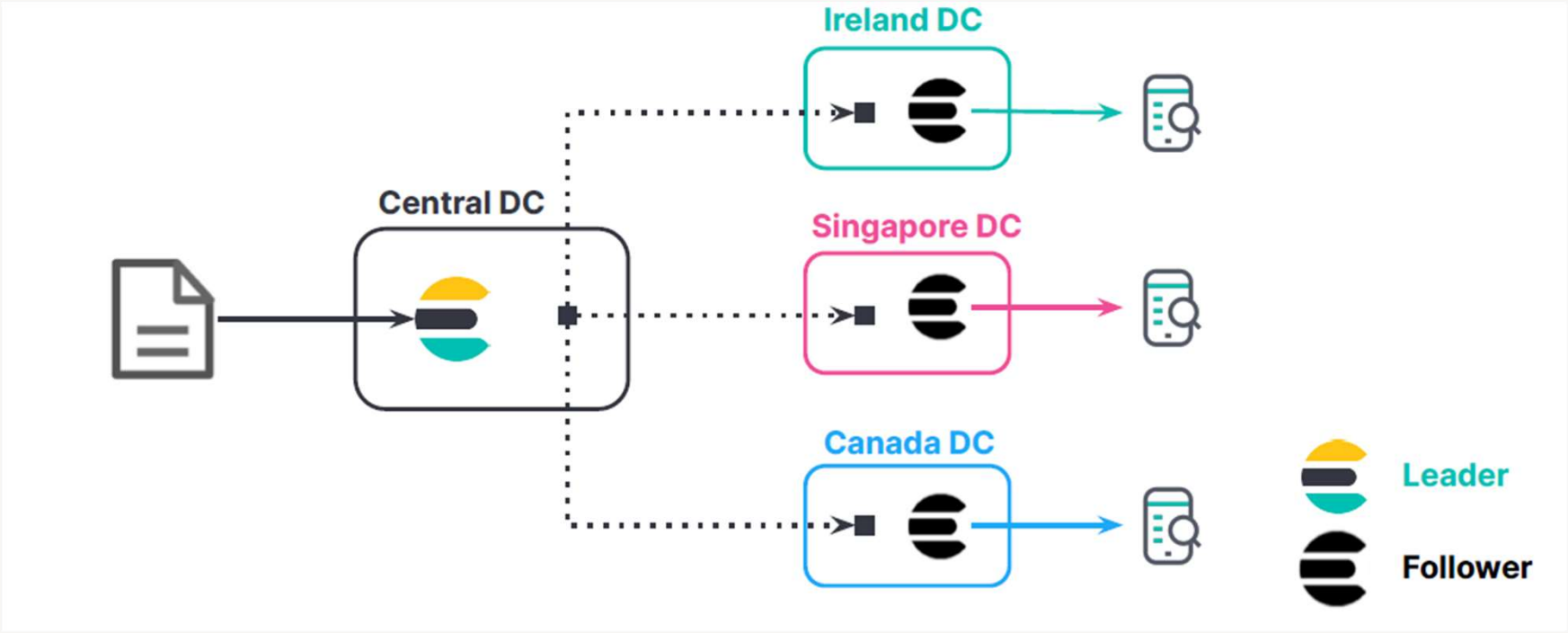
Cross Cluster Replication

Disaster recovery and high availability



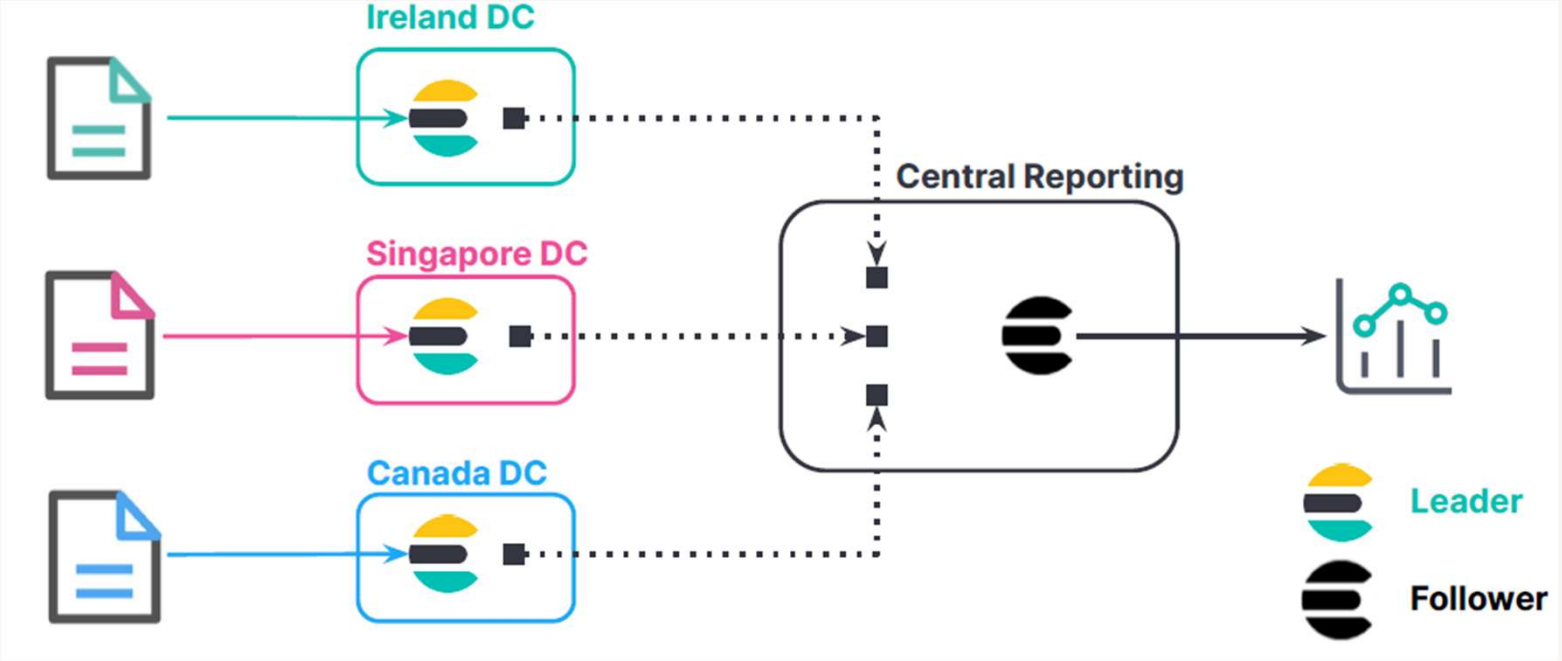
Cross Cluster Replication

Data locality



Cross Cluster Replication

Centralized reporting



Questions

Thank you

Mr. Iyanou Eraste AKANDE

eraste.akande@gmail.com