perplexity

Create a note

Here is a consolidated note summarizing the entire ELK stack setup, configuration, and troubleshooting you completed for your final-year project with Docker and Spring Boot on a VM:

ELK Stack Setup Notes for Final-Year Project

1. Overview

- ELK Stack: Elasticsearch, Logstash, Kibana
- Purpose: Real-time collection, storage, and visualization of Spring Boot app logs
- Deployed with Docker Compose on an Azure VM
- Integrated with MySQL database and Spring Boot backend

2. Docker Compose Configuration for ELK

Elasticsearch Service

```
elasticsearch:
  image: docker.elastic.co/elasticsearch/elasticsearch:8.10.0
  container_name: elasticsearch
  environment:
    - discovery.type=single-node
    - ES_JAVA_OPTS=-Xms512m -Xmx512m
    # For dev: can disable security with - xpack.security.enabled=false
  ports:
    - "9200:9200"
  volumes:
    - es_data:/usr/share/elasticsearch/data
  networks:
    - elk-network
```

Logstash Service

- Mount a directory (not file) for pipeline config to avoid mount errors

```
logstash:
  image: docker.elastic.co/logstash/logstash:8.10.0
  container_name: logstash
  ports:
```

```
    - "5044:5044"
    - "9600:9600"       # Optional for monitoring
  volumes:
    - ./logstash-pipeline:/usr/share/logstash/pipeline
  depends_on:
    - elasticsearch
  networks:
    - elk-network
```

Kibana Service

```
kibana:
  image: docker.elastic.co/kibana/kibana:8.10.0-1   # Correct tag version
  container_name: kibana
  ports:
    - "5601:5601"
  depends_on:
    - elasticsearch
  networks:
    - elk-network
```

Networks & Volumes

- Separate network `elk-network` for ELK stack isolation
- Persist Elasticsearch data using Docker volume `es_data`

3. Logstash Pipeline Configuration

- Create a folder `logstash-pipeline` in your project root
- Inside it, create `logstash.conf` with content:

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

filter {
  grok {
    match => { "message" => "%{TIMESTAMP_ISO8601:timestamp} %{LOGLEVEL:level} %{GREEDYDAT
  }
}

output {
  elasticsearch {
    hosts => ["elasticsearch:9200"]
    user => "elastic"            # Use if security enabled
    password => "YourPassword"  # Use if security enabled
    index => "springboot-logs-%{+YYYY.MM.dd}"
  }
```

```
  stdout { codec => rubydebug }
}
```

- This pipeline reads Spring Boot log files forwarded by Filebeat or other means and indexes into Elasticsearch.

4. Spring Boot Logs Setup

- Ensure logs are written to a shared volume, e.g., `/home/azureuser/springboot_logs`
- Mount this volume into Logstash container as read-only for file input
- Update Docker Compose accordingly:

```
volumes:
  springboot_logs:
```

Mount in services:

```
logstash:
  volumes:
    - ./logstash-pipeline:/usr/share/logstash/pipeline
    - springboot_logs:/home/azureuser/springboot_logs:ro
```

Spring Boot backend logs to this path.

5. Elasticsearch & Kibana Credentials

- Default username is `elastic`
- Password:
  - If `ELASTIC_PASSWORD` is unset in docker-compose, Elasticsearch auto-generates a password at first start
  - Retrieve password from logs:

```
docker logs elasticsearch
```

Look for:
`Generated password for user [elastic]: <password>`

- Alternatively, reset password inside container:

```
docker exec -it elasticsearch /bin/bash
bin/elasticsearch-reset-password -u elastic
```

- Kibana uses the same username/password as Elasticsearch

6. Kibana Enrollment Token Setup

- On first run, Kibana asks for enrollment token to connect securely
- Generate enrollment token inside Elasticsearch container:

```
docker exec -it elasticsearch /bin/bash
bin/elasticsearch-create-enrollment-token -s kibana
```

- Copy token and paste in Kibana setup UI at http://<VM_IP>:5601
- Next step in Kibana setup requires verification code:

```
docker exec -it kibana /bin/bash
bin/kibana-verification-code
```

- Paste code into Kibana UI to finalize connection

7. Common Issues and Fixes

Issue	Cause	Fix
Mount error: file vs directory conflict	Mounting file on directory or vice versa	Delete wrong dir/file and create proper config file or dir
HTTP plaintext warning in Elasticsearch	Using HTTP on HTTPS port	Disable security in dev or configure HTTPS properly
Container name conflict	Existing container not removed	`docker rm -f <container>` to remove old container
Kibana image not found	Wrong Docker image tag	Use correct tag like `8.10.0-1`
Can't find enrollment token	Running command in wrong container	Run token command inside Elasticsearch container

8. Ports to Open on VM

Service	Port(s)	Purpose
Spring Boot	8080	Backend API access
MySQL	3306	Database (limit external access)
Elasticsearch	9200	REST API
Elasticsearch	9300	Internal cluster communication
Logstash	5044	Beats input (Filebeat, etc.)
Logstash	9600	Monitoring port (optional)
Kibana	5601	Web UI to visualize logs

Security: Restrict 3306, 9300, and 9600 to internal or VPN access only.

9. Commands Summary

- Start all services:

```
docker compose --env-file ./docker-compose.env up -d --build
```

- Remove conflicting container:

```
docker rm -f <container_name>
```

- View container logs:

```
docker logs <container_name>
```

- Generate Kibana enrollment token:

```
docker exec -it elasticsearch /bin/bash
bin/elasticsearch-create-enrollment-token -s kibana
```

- Get Kibana verification code:

```
docker exec -it kibana /bin/bash
bin/kibana-verification-code
```

10. Next Steps (Optional)

- Configure Filebeat to forward Spring Boot container logs to Logstash automatically.
- Set up Kibana dashboards for Spring Boot log visualization.
- Harden security with TLS certificates and user roles.
- Automate deployment and updates with Watchtower.

If you want, I can prepare a neatly formatted PDF of this note or provide further scripts for automating the setup. Let me know!

⁂

1. paste.txt