National College of Ireland

Configuration Manual

MSc Research Project
Cloud Computing

Akhila Kandadi
Student ID: 23386347

School of Computing
National College of Ireland

Supervisor: Shaguna Gupta

National College of Ireland
Project Submission Sheet
School of Computing

Student Name:	Akhila Kandadi
Student ID:	23386347
Programme:	Cloud Computing
Year:	2025-26
Module:	MSc Research Project
Supervisor:	Shaguna Gupta
Submission Due Date:	28/01/2026
Project Title:	Configuration Manual
Word Count:	393
Page Count:	6

I hereby certify that the information contained in this (my submission) is information pertaining to research I conducted for this project. All information other than my own contribution will be fully referenced and listed in the relevant bibliography section at the rear of the project.

ALL internet material must be referenced in the bibliography section. Students are required to use the Referencing Standard specified in the report template. To use other author's written or electronic work is illegal (plagiarism) and may result in disciplinary action.

Signature:	Akhila Kandadi
Date:	26th January 2026

PLEASE READ THE FOLLOWING INSTRUCTIONS AND CHECKLIST:

Attach a completed copy of this sheet to each project (including multiple copies).	☐
Attach a Moodle submission receipt of the online project submission, to each project (including multiple copies).	☐
You must ensure that you retain a HARD COPY of the project, both for your own reference and in case a project is lost or mislaid. It is not sufficient to keep a copy on computer.	☐

Assignments that are submitted to the Programme Coordinator office must be placed into the assignment box located outside the office.

Office Use Only	
Signature:	
Date:	
Penalty Applied (if applicable):	

Configuration Manual

Akhila Kandadi
23386347

1 Introduction

This configuration guide includes an intricate instruction on how to deploy and set up the Scalable Threat Assessment Machine Learning Framework of CI/CD DevSecOps Pipelines. The system consists of a Transformer-GNN ensemble type of model running on the AWS EKS and Kubernetes orchestrator, Apache Kafka to process streams and GitLab CI/CD to deploy continuously. The purpose of this manual is to help a developer or a researcher recreate or expand on this framework.

2 System Requirements and Tools

Table 1 lists all tools and package versions used in this research. Create `requirements.txt` with this version.

Table 1: Tools and Version Requirements

Tool	Version
Python	3.12.9
PyTorch	2.5.0
PyTorch Geometric	2.6.1
scikit-learn	1.6.1
XGBoost	2.0.3
Docker Desktop	27.x
kubectl	v1.30+
AWS CLI	2.x
Git	2.x

3 Development Environment Setup

3.1 Clone Project Repository

```
git clone https://github.com/[username]/akhila-thesis.git
cd akhila-thesis/CICIDS2017/deployment
```

3.2 Python Virtual Environment Creation

Create and activate a Python virtual environment:

```
python3 -m venv venv
source venv/bin/activate  # Linux/macOS
# venv\Scripts\activate   # Windows
```

3.3 Installing Python Packages and Dependencies

```
pip install -r requirements.txt
```

The `requirements.txt` includes ML frameworks (torch, torch-geometric, scikit-learn, xgboost), API server (flask, gunicorn), monitoring (prometheus-client), streaming (kafka-python-ng), and cloud utilities (boto3, PyJWT).

4 Docker Configuration

4.1 Dockerfile Structure

```
FROM python:3.12-slim
WORKDIR /app
COPY requirements.txt .
RUN pip install --no-cache-dir -r requirements.txt
COPY inference_pipeline.py api_server.py ./
COPY kafka/kafka_consumer.py ./
COPY continuous_learning/ ./continuous_learning/
COPY *.pth *.pkl ./checkpoints/
ENV MODEL_PATH=/app/checkpoints PORT=8000
EXPOSE 8000
CMD ["gunicorn", "--bind", "0.0.0.0:8000",
     "--workers", "2", "--timeout", "120", "api_server:app"]
```

4.2 Building the Docker Image (for AWS Fargate

```
docker buildx build --platform linux/amd64 \
  -t threat-detection:latest .
```

4.3 Local Testing

Test the container locally before deploying:

```
docker run -p 8000:8000 threat-detection:latest
curl http://localhost:8000/health
```

5 AWS Cloud Infrastructure Setup (From Local

5.1 AWS CLI Configuration

```
aws configure
# Enter: AWS Access Key ID, Secret Access Key
# Default region: eu-west-1
```

5.2 ECR Repository Creation

```
aws ecr create-repository \
  --repository-name threat-detection \
  --region eu-west-1
```

5.3 EKS Cluster Setup with Fargate Profile

```
aws eks create-cluster \
  --name threat-detection-cluster \
  --region eu-west-1 \
  --kubernetes-version 1.30 \
  --role-arn arn:aws:iam::<ACCOUNT_ID>:role/EKSClusterRole

aws eks create-fargate-profile \
  --cluster-name threat-detection-cluster \
  --fargate-profile-name threat-detection-profile \
  --pod-execution-role-arn arn:aws:iam::<ACCOUNT_ID>:role/
     FargatePodRole \
  --selectors namespace=threat-detection
```

5.4 Push Docker Image to ECR

```
aws ecr get-login-password --region eu-west-1 | \
  docker login --username AWS --password-stdin \
  <ACCOUNT_ID>.dkr.ecr.eu-west-1.amazonaws.com

docker tag threat-detection:latest \
  <ACCOUNT_ID>.dkr.ecr.eu-west-1.amazonaws.com/threat-detection:
     latest

docker push \
  <ACCOUNT_ID>.dkr.ecr.eu-west-1.amazonaws.com/threat-detection:
     latest
```

6 Kubernetes Deployment

6.1 Namespace and Configuration

```
kubectl apply -f kubernetes/namespace.yaml
kubectl apply -f kubernetes/configmap.yaml
kubectl create secret docker-registry ecr-registry-secret \
  --docker-server=<ACCOUNT_ID>.dkr.ecr.eu-west-1.amazonaws.com \
  --docker-username=AWS \
  --docker-password=$(aws ecr get-login-password) \
  -n threat-detection
```

6.2 Deploy Threat Detection API

The deployment creates 3 replicas with specifications of 2Gi memory request (4Gi limit), 500m CPU request (2000m limit) and health check at `/health` endpoint, readiness probe at `/ready` endpoint.

```
kubectl apply -f kubernetes/deployment.yaml
```

6.3 Horizontal Pod Autoscaler

Auto-scaling based on CPU/memory utilization:

```
kubectl apply -f kubernetes/hpa.yaml
```

Create with a minimum of 3 replicas (max:10) and scale up to 70% utilization

7 Kafka Streaming Pipeline

7.1 Kafka Cluster Deployment (kafka-cluster.yaml

```
kubectl create namespace kafka
kubectl apply -f kafka/kafka-cluster.yaml -n kafka
```

7.2 Topic Creation

Create required Kafka topics for network traffic and model feedback:

```
KAFKA_POD=$(kubectl get pods -n kafka -l app=kafka \
  -o jsonpath='{.items[0].metadata.name}')

kubectl exec $KAFKA_POD -n kafka -- \
  /opt/kafka/bin/kafka-topics.sh --create \
  --topic network-traffic --partitions 3 \
  --replication-factor 1 --bootstrap-server localhost:9092
```

7.3 Kafka Consumer for Stream Processing

```
kubectl apply -f kafka/kafka-consumer-deployment.yaml
```

8 Continuous Learning Pipeline

8.1 Feedback API Deployment

Deploy the feedback collection API:

```
kubectl apply -f kubernetes/continuous-learning.yaml
```

8.2 Model Versioning

Models are automatically versioned and stored in S3:

```
aws s3 mb s3://threat-detection-models-<ACCOUNT_ID> --region eu-
   west-1
aws s3 ls s3://threat-detection-models-<ACCOUNT_ID>/
```

9 GitLab CI/CD Integration

9.1 Webhook Receiver Deployment

Deploy the GitLab webhook receiver:

```
kubectl apply -f gitlab/webhook-receiver-deployment.yaml
```

10 Monitoring and Metrics

The API server exposes Prometheus metrics at `/metrics`:

```
kubectl port-forward svc/threat-detection-service 8080:80 \
  -n threat-detection
curl http://localhost:8080/metrics
```

Available metrics include: `predictions_total` (total predictions by class), `prediction_latency_se` (inference latency histogram), and `batch_size` (processed batch sizes).

11 Verification and Testing

11.1 Health Check Commands

```
kubectl port-forward svc/threat-detection-service 8080:80 \
  -n threat-detection &
kubectl port-forward svc/feedback-api 8081:80 \
  -n threat-detection &
```

```
curl http://localhost:8080/health
curl http://localhost:8080/model/info
curl http://localhost:8081/feedback/stats
```

11.2 API Testing

Test the prediction endpoint:

```
curl -X POST http://localhost:8080/predict \
  -H "Content-Type: application/json" \
  -d '{"features": [[0.1, 0.2, ..., 0.78]]}'
```

11.3 End-to-End Demo

Run the interactive demonstration script:

```
source venv/bin/activate
python3 demo_e2e_pipeline.py
```

This script demonstrates system health verification, model information retrieval, GitLab webhook simulation, real-time inference with CICIDS2017 data, feedback submission workflow, and model versioning operations.