National College of Ireland

Configuration Manual

MSc Research Project
MSc. Cloud Computing

Sri Lakshmi Prasanna Eda
x23304723

School of Computing
National College of Ireland

Supervisor: Ahmed Makki

National College of Ireland

MSc Project Submission Sheet

School of Computing

Student Name: Sri Lakshmi Prasanna Eda

Student ID: x23304723

Programme: MSc. Cloud Computing **Year:** 2025-2026

Module: MSc Research Project

Lecturer: Ahmed Makki

Submission Due Date:28/01/2026

Project Title: Application of Hierarchical Deep Reinforcement Learning for Dynamic Task Scheduling in Multi-Region Cloud Environments

Word Count: 1094 **Page Count:** 09

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 : Sri Lakshmi Prasanna Eda

Date: :28/01/2026

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

Sri Lakshmi Prasanna Eda
x23304723

1 Introduction

In this manual, you can find instructions on how to configure and operate the HDRL (Hierarchical Deep Reinforcement Learning) architecture of privacy-preserving distributed cloud task scheduling. The project adopts a two-level architecture wherein local PPO agents deal with the allocation of resources to particular regions whereas a global coordinator determines the cross-provider choices through the application of differential privacy and the secure multi-party computations. The system is tested on Google Cloud Trace datasets and implemented on the AWS multi-region infrastructure.

2 System Requirements

Local Hardware:
- Processor: x86-64 compatible CPU, ARMx64
- Memory: Minimum 8GB RAM (16 GB Recommended)
- Storage: 20 GB free disk space
- GPU: Google Colab Pro for model training

Operating System:
- macOS /Ubuntu/Windows with WSL2

AWS Deployment:
- AWS Account (Personal or Academy)
- AWS CLI v2
- Key pair for EC2 instance access

3 Software Dependencies

3.1 Core Softwares

Software	Version	Download Link
Python	3.10+	https://www.python.org/downloads/
Google Colab Pro	-	https://colab.research.google.com/
Visual Studio Code	Recent	https://code.visualstudio.com/download
AWS CLI	2.0+	https://aws.amazon.com/cli/

3.2 Python Dependencies

Package Name	Version	Description
h5py	3.15.1	HDF5 file format support
NumPy	2.2.6	Numerical Computation
TensorFlow	2.19.0	Deep learning framework
tensorflow-probability	Latest	PPO implementation
pandas	2.2.2	Data processing
scikit-learn	1.3.0	Data Science Framework
flask	2.3.2	API framework
boto3	Latest	AWS SDK for Python

4 Data Preparation

The project uses the Google Cloud Trace dataset containing 405,894 authentic task scheduling records from Google's production infrastructure.

The data is located at:
- ./HDRL_Research/data/raw/borg_traces_data.csv (raw dataset) – Downloaded from https://www.kaggle.com/datasets/derrickmwiti/google-2019-cluster-sample
- ./HDRL_Research/data/processed/train_tasks.csv (processed training set)

To run preprocessing again,

1_Dataset_Preparation_HDRL_v2.ipynb on Google Colab

The output generates:
- train_tasks.csv, val_tasks.csv, test_tasks.csv
- task_scaler.pkl

5 Model Training on Colab

5.1 Phase 2: Training PPO Agents

Open 3_PPO_Task_Segmentation_HDRL_v3_FIXED_MultiCloud.ipynb and run in Google Colab Pro on T4 GPU or higher.

Input: train_tasks.csv, val_tasks.csv, test_tasks.csv

Output file: - AWS-US-EAST-1_final.weights.h5, AWS-EU-WEST-1_final.weights.h5, training_history.pkl

5.2 Phase 3: Global Coordinator

Open 4_Global_Coordinator_HDRL_Phase3_AWS.ipynb in Google Colab Pro with T4 GPU.

Input: train_tasks.csv, val_tasks.csv, test_tasks.csv, AWS-US-EAST-1_final.weights.h5, AWS-EU-WEST-1_final.weights.h5
Output file: gc_final.weights.h5 (coordinator weights), task_segmenter_v3.pkl (K-means model), dp_layer_config_v3.json (privacy configuration), scalers.pkl

6 Project Structure

```
├── HDRL_Research/
│   ├── data/
│   │   ├── raw/
│   │   │   └── borg_traces_data.csv              # Original Google traces
│   │   └── processed/
│   │       ├── train_tasks.csv                   # Training set
│   │       ├── val_tasks.csv                     # Validation set
│   │       └── test_tasks.csv                    # Test set
│   ├── models/
│   │   ├── ppo_agents_v4_aws/
│   │   │   ├── AWS-US-EAST-1_final.weights.h5
│   │   │   └── AWS-EU-WEST-1_final.weights.h5
│   │   └── global_coordinator/
│   │       ├── gc_final.weights.h5
│   │       ├── task_segmenter_v3.pkl
│   │       ├── dp_layer_config_v3.json
│   │       └── scalers.pkl
│   ├── results/
│   │   ├── phase1_part2 /                        # Data Preparation Results
│   │   ├── phase2_aws/                           # Phase 2 PPO training results
│   │   └── phase3_aws/                           # Phase 3 Global Coordinator training results
│   ├── requirements.txt                          # Python dependencies
├── aws-deployment/
│   ├── global_coordinator_api.py                 # Coordinator Flask API
│   ├── local_agent_api.py                        # Agent Flask API
│   ├── validate_allocation.py                    # Allocation test script
│   ├── system_validation_test.sh                 # Full system validation
├── 1_Dataset_Preparation_HDRL_v2                 # Data Preparation Notebook
├── 3_PPO_Task_Segmentation_HDRL_v3_FIXED_MultiAccount # PPO model training notebook
├── 4_Global_Coordinator_HDRL_Phase3_AWS          # Global Coordinator Training notebook
```

7 AWS Deployment

7.1 AWS Configuration

$ aws configure
AWS Access Key ID: [Access Key]
AWS Secret Access Key: [Secret Key]
Default region name: us-east-1/eu-west-1
Default output format: Json

7.2 Infrastructure Setup

The deployment creates:

US-EAST-1:
- VPC: 10.0.0.0/16
- Public Subnet: 10.0.1.0/24
- Global Coordinator: m7i-flex.large EC2 instance
- Local Agent 1: t3.small EC2 instance
- S3 buckets: hdrl-models-us-east-1, hdrl-data-us-east-1, hdrl-results-us-east-1
- DynamoDB tables: hdrl-global-state, hdrl-task-queue, hdrl-metrics

EU-WEST-1:
- VPC: 10.1.0.0/16
- Public Subnet: 10.1.1.0/24
- Local Agent 2: t3.small Spot instance
- S3 bucket: hdrl-results-eu-west-1
- DynamoDB table: hdrl-regional-state-eu

Cross-Region:
- VPC Peering connection between regions
- Expected latency: 80-120ms (actual: ~155ms)

7.3 Connect to EC2 Instances

For Global Coordinator (us-east-1):

$ ssh -i hdrl-key-us.pem ubuntu@[public_ip_global_coordinator]

The same applies for both the local agents with their respective ip and key file.

7.4 Installing dependencies on EC2

$ python3 -m venv hdrl_env
$ source hdrl_env/bin/activate
$ pip install --upgrade pip

$ pip install tensorflow numpy flask boto3 requests

7.5 Uploading models to EC2

For Global Coordinator:

```
$ scp -i hdrl-key-us.pem HDRL_Research/models/global_coordinator/* \
  ubuntu@[public_ip_global_coordinator]:~/models/

$ scp -i hdrl-key-us.pem aws-deployment/global_coordinator_api.py \
  ubuntu@[public_ip_global_coordinator]:~/
```

For Local Agent 1 (us-east-1):

```
$ scp -i hdrl-key-us.pem HDRL_Research/models/ppo_agents_v4_aws/AWS-US-EAST-1_final.weights.h5 \
  ubuntu@[public_ip_local_agent1]:~/models/

$ scp -i hdrl-key-us.pem aws-deployment/local_agent_api.py \
  ubuntu@[public_ip_local_agent1]:~/
```

For Local Agent 2 (eu-west-1):

```
$ scp -i hdrl-key-eu.pem HDRL_Research/models/ppo_agents_v4_aws/AWS-EU-WEST-1_final.weights.h5 ubuntu@[public_ip_local_agent2]:~/models/

$ scp -i hdrl-key-eu.pem aws-deployment/local_agent_api.py \
  ubuntu@[public_ip_local_agent2]:~/
```

7.6 Verify Deployment

Test health endpoints:

```
$ curl http://[public_ip_global_coordinator]:8000/health
```

Expected response:

```
{
  "status": "healthy",
  "coordinator_ready": true,
  "models_loaded": true,
  "total_tasks_allocated": 0
}
```

Test agent endpoints:

```
$ curl http://[public_ip_local_agent1]:8001/health
```

$ curl http://[public_ip_local_agent2]:8002/health

7.7 Deployment Testing

Run the comprehensive validation on Global Coordinator E2 Instance

$ cd aws-deployment
$ chmod +x system_validation_test.sh
$ ./system_validation_test.sh

This tests:
- Health checks for all components
- Cross-region connectivity
- Task allocation with various workloads
- Privacy budget consumption
- Response latencies

8 References

Google Colab: https://colab.research.google.com/
AWS CLI v2: https://docs.aws.amazon.com/cli/v1/userguide/install-windows.html
Google Traces Dataset: https://www.kaggle.com/datasets/derrickmwiti/google-2019-cluster-sample