National College of Ireland

Developing an AI-Powered Zero-Trust Cybersecurity Framework for Malware Prevention

Configuration Manual
Cybersecurity

Manikanta Puttoj
Student ID: 23249889

School of Computing
National College of Ireland

Supervisor: Mosab Mohamed
Second Supervisor: Dr. Mosab Hamdan

National College of Ireland
Project Submission Sheet
Msc in cybersecurity

Student Name:	Manikanta Puttoj
Student ID:	23249889
Programme:	MSc in Cybersecurity
Year:	2026
Module:	Configuration Manual
Supervisor:	Mosab Mohamed
Submission Due Date:	28/01/26
Project Title:	Developing an AI-Powered Zero-Trust Cybersecurity Framework for Malware Prevention
Word Count:	N/A
Page Count:	7

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:	Manikanta Puttoj
Date:	28th 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):	

Developing an AI-Powered Zero-Trust Cybersecurity Framework for Malware Prevention

Manikanta Puttoj
23249889

1 System Configuration

This section outlines the hardware and software specifications required to successfully run the Zero Trust Architecture (ZTA) simulation framework.

1.1 Hardware Requirements

The simulation is computationally intensive, particularly during the machine learning model training phase (Isolation Forest). The recommended specifications are:

- **Processor:** Intel Core i5 / AMD Ryzen 5 or higher (Recommended: i7/Ryzen 7 for large-scale simulations).
- **RAM:** Minimum 8GB (Recommended: 16GB or higher to handle large Pandas DataFrames).
- **Storage:** Approx. 500MB free space for code, logs, and generated charts.
- **OS:** Windows 10/11, macOS 12+, or Linux (Ubuntu 20.04+).

1.2 Software Requirements

The project relies on Python and several data science libraries.

- **Python:** Version 3.9 or higher.
- **Package Manager:** pip (Python Package Installer).
- **IDE (Optional):** VS Code, PyCharm, or Jupyter Notebook for code inspection.

1.3 Dependencies

The following Python packages are required (listed in requirements.txt):

- numpy and pandas: For data manipulation and log processing.
- scikit-learn: For the Isolation Forest anomaly detection model.
- matplotlib and seaborn: For generating analysis charts.
- faker: For generating realistic synthetic user and device data.
- cryptography: For simulated encryption operations.

2 Installation Guide

Follow these steps to set up the development environment and install necessary dependencies.

2.1 Step 1: Unzip the Project Archive

Extract the contents of the submission zip file to a directory of your choice (e.g., D:/Zero-Trust-Archit

2.2 Step 2: Create a Virtual Environment (Recommended)

It is best practice to run the simulation in an isolated Python environment to avoid conflicts.

Windows:

```
cd path/to/project
python -m venv venv
.\venv\Scripts\activate
```

macOS / Linux:

```
cd path/to/project
python3 -m venv venv
source venv/bin/activate
```

2.3 Step 3: Install Dependencies

Once the virtual environment is active, install the required libraries using the provided requirements.txt file:

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

Verify the installation by running:

```
pip list
```

Ensure that packages like scikit-learn, pandas, and faker are present.

3 Usage Guide

This section details how to configure and execute the simulation.

3.1 Configuration

The simulation configuration file (config.py) located in the main project folder creates the simulation parameters and controls the scaling.

Key parameters include:

- NUM_USERS: Number of synthetic users (default: 200).
- SIMULATION_DAYS: Duration of the simulation (default: 60 days).
- EVENTS_PER_DAY: Traffic volume per day.
- BREACH_SCENARIOS: Probability distributions for different attack types.

3.2 Running the Simulation

To start the simulation, execute the main.py script from the project root:

```
python main.py
```

3.3 Understanding the Output

The simulation can be broken down into 9 stages, with the output being provided during each completed stage of the simulation until results are written to the following location:

1. **Console Output:** Displays real-time breach attempts (PREVENTED vs. SUCCESSFUL) and phase completion status.
2. **Charts (**charts/ **directory):**
    - security_metrics.png: Breach prevention and compliance rates.
    - authentication.png: Real-time active sessions view.
    - executive_dashboard.png: High-level summary of all KPIs.
    - breach_analysis.png: Breakdown of attack types.
3. **Reports (**results/ **directory):**
    - comprehensive_report.txt: A text-based summary of the entire run.
    - *.csv files: Raw data logs for users, devices, and access attempts for further analysis.

4 Simulation Configuration Parameters

The behavior of the Zero Trust Architecture simulation is governed by parameters defined in config.py. This section provides a detailed explanation of each parameter group.

4.1 Environment and Scale

- NUM_USERS: Defines the total population of the simulation. A larger number (e.g., 200+) provides better statistical significance but increases simulation time.
- NUM_DEVICES: The total pool of devices available. Typically set to 1.5x - 2x the user count to simulate multi-device usage (e.g., laptop + mobile).
- EVENTS_PER_DAY: The volume of traffic generated. Higher values (e.g., 500-1000) create denser logs for the AI model to learn from.
- SIMULATION_DAYS: The simulated duration. At least 30 days is recommended to observe trends.

4.2 Zero Trust Policy Thresholds

These thresholds determine the strictness of the Policy Decision Point (PDP).

Parameter	Description	Default
max_failed_attempts	Number of allowed login failures before account lockout.	3
session_timeout	Minutes of inactivity before re-authentication is required.	15
device_trust_min	Minimum score (0-100) for a device to be considered "secure".	75
user_risk_max	Maximum allowed user risk score before blocking access.	40
anomaly_threshold	Sensitivity of the Isolation Forest model (0.0 - 1.0). Lower means stricter detection.	0.65

4.3 Breach Scenarios

The BREACH_SCENARIOS dictionary controls the probability and severity of simulated attacks.

```
'lateral_movement': {
    'probability': 0.15,     # 15% chance per day
    'severity': 'high'
}
```

Adjusting these probabilities allows for stress-testing the system against specific threats (e.g., increasing 'insider_threat' probability to test internal controls).

5 Interpreting Simulation Outputs

Upon completion, the simulation generates artifacts in the results/ and charts/ directories. This section explains how to interpret these outputs.

5.1 Visualizations (charts/)

5.1.1 Security Metrics (security_metrics.png)

This bar chart compares the **Baseline** (Traditional) model against the proposed **ZTA** model.

- **Breach Prevention Rate:** Higher is better. ZTA should show significant improvement (¿80%).
- **Security Score:** A composite metric (0-100). ZTA typically scores 20-30 points higher.

5.1.2 Real-Time Active Sessions (authentication.png)

This Time-Series plot shows the number of concurrent active sessions over a 24-hour period.

- **Blue Line:** The actual number of active sessions.
- **Trend:** Shows peak usage times. Sudden spikes that do not follow the trend may indicate anomalies.

5.1.3 Breach Analysis (breach_analysis.png)

A comparative bar chart showing the number of successful vs. prevented attacks for each breach type (Lateral Movement, Credential Theft, etc.).

- **ZTA Success:** Calculated as: $\frac{\text{Prevented Attacks}}{\text{Total Attempts}} \times 100\%$

5.2 Data Logs (results/)

- comprehensive_report.txt: A summary text file containing high-level statistics, total execution time, and recommendations.
- users_data.csv: Contains details of all synthetic users, including their assigned roles and risk scores.
- breach_attempts.csv: A distinct log of every simulated attack, recording the timestamp, attack type, target asset, and whether it was blocked.

6 Troubleshooting

This section addresses common issues that may arise during installation or execution.

6.1 ModuleNotFoundError

Error: ModuleNotFoundError: No module named 'faker' (or similar).
Cause: The required Python packages are not installed in the current environment.
Solution: ensure your virtual environment is activated and run:

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

6.2 Memory Errors

Error: MemoryError during the training phase.
Cause: The simulation is generating too many logs for your available RAM.
Solution: Reduce the usage scale in config.py:

- Decrease NUM_USERS to 50.
- Decrease SIMULATION_DAYS to 7.

6.3 Permission Issues

Error: Permission denied when writing to results/ or charts/.
Cause: The script does not have write access to the directory, or a file is open in another program (e.g., Excel).
Solution: Close any open CSV/PNG files and try running the script as Administrator/Sudo.

6.4 Plotting Issues

Issue: Charts are not generated or look empty.
Solution: Ensure matplotlib is installed. If running on a headless server (no screen), you may need to set the backend to Agg:

```
import matplotlib
matplotlib.use('Agg')
```

(Note: The provided visualizer.py typically handles this, but it is a common environment-specific issue).

7 Code Structure Overview

Understanding the modular structure of the project helps in customization and debugging.

7.1 Core Modules

- main.py: The entry point. It initializes the environment, runs the simulation phases, and triggers reporting.
- config.py: Central configuration file for all simulation parameters.

7.2 Simulation Package (simulation/)

- environment.py: Manages the state of the hybrid workplace (Users, Devices, Apps).
- user_behavior.py: Generates realistic synthetic actions (Logins, Requests) based on user roles.
- breach_simulator.py: Injects malicious events to test the security controls.

7.3 Analysis Package (analysis/)

- ai_engine.py: Contains the **Isolation Forest** model logic. It trains on "normal" behavior and predicts anomaly scores.
- visualizer.py: Uses matplotlib and seaborn to generate the final charts.
- metrics_calculator.py: Computes the KPIs (Breach Prevention Rate, Security Score) from raw logs.

8 Advanced Customization

For researchers wishing to extend the project, the following modifications are possible.

8.1 Adding New Breach Scenarios

To test a new type of attack (e.g., ”Ransomware”):

1. Open simulation/breach_simulator.py.
2. Add a new method simulate_ransomware(self).
3. Define the logic (e.g., rapid file encryption attempts).
4. Register the scenario in config.py under BREACH_SCENARIOS.

8.2 Modifying the AI Model

The current implementation uses Isolation Forest. To test a different algorithm (e.g., One-Class SVM or Autoencoders):

1. Open analysis/ai_engine.py.
2. Update the train_model method to initialize your new algorithm.
3. Ensure the predict method returns a compatible risk score (normalized 0-1).

8.3 Integrating External Datasets

Currently, data is synthetically generated. To use real-world datasets (e.g., CIC-IDS2017):

1. Replace the generate_traffic logic in simulation/environment.py with a CSV loader.
2. Map the external dataset features (Packet Size, Port, Protocol) to the internal event structure expected by the AI_Engine.