Problem Statement

This project aims to develop a Graph Neural Network (GNN)-based anomaly detection system for securing robotic behavior in high-risk, dynamic environments such as post-disaster zones. The system focuses on detecting cyber-physical threats like GPS spoofing, sensor tampering, and trajectory deviations in real-time, enabling adaptive trust-aware task allocation in multi-agent rescue scenarios. By integrating anomaly detection into a knowledge graph and optimizing robot coordination with quantum-based solvers.

Literature Review 1

Title: Adversarial Attacks on Node Embeddings via Graph Poisoning

Authors: Aleksandar Bojchevski and Stephan Gunnemann

Publication Year: 2019

Key Ideas

The article focuses on graph-based models like DeepWalk, GCN, and GAT that learn node embeddings vulnerable to adversarial manipulation. It explored how poisoning attacks can be performed by adding or removing a small number of edges, which degrades the quality of the learned embeddings without obvious detection. It also discussed how attackers can target downstream tasks such as node classification or link prediction by maximizing loss or misclassifying specific nodes.

Contributions

The paper introduced a meta-learning-based poisoning attack that selects edges to flip in order to maximize downstream task damage. It proposed two attacks, general attacks that degrade overall embedding quality, and targeted attacks that focus on misclassifying specific nodes or disrupting links. This demonstrated that even restricted attackers who have access to a subset of the graph can inflict significant embedding damage. It also proved the transferability of poisoned graphs, one attack can harm multiple GNN models, not just the one used in training.

Limitations

The approach is offline and assumes knowledge of the full graph structure and downstream tasks. It mainly focuses on attack effectiveness, without addressing detection, defense, or recovery strategies. The adversarial edge flips cannot be easily distinguished from natural edges by degree or centrality, making trust modeling more difficult.

Extend/Improvement

My research plans to integrate real-time anomaly detection using GNNs to monitor poisoned or perturbed subgraphs. Instead of a static response, there would be a trust-aware task assignment model that penalizes compromised agents and reroutes assignments using QUBO. I plan to use trust scores, knowledge graphs, and dynamic reallocation in cyber-physical scenarios like robotic coordination under adversarial interference.

Conceptual Diagrams

The project considers a post-earthquake hospital scenario where multiple autonomous agents coordinate tasks such as patient rescue and supply delivery. Figure 1 shows the trust-based relationships between agents and their assigned tasks, while Figure 2 illustrates the threat model of cyber-poisoning attacks leading to trust degradation and the mitigation pipeline using QUBO-based optimization for resilient task assignment.

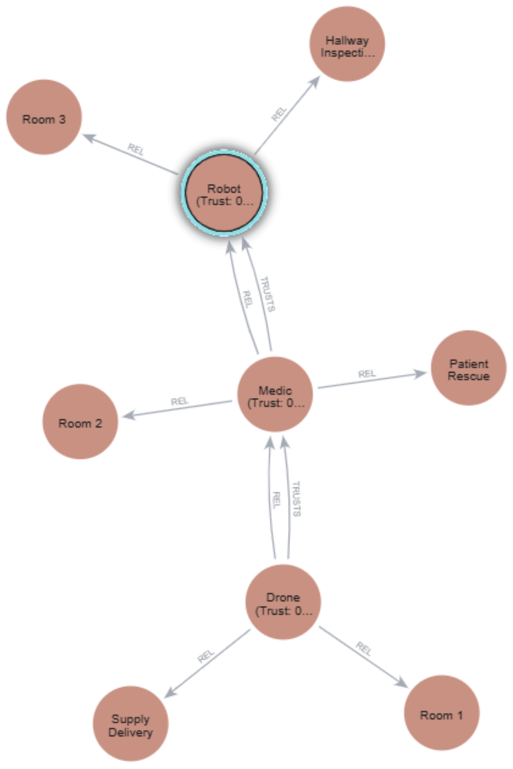


Figure 1. Agent-task coordination scenario with trust relationships.
This diagram illustrates the agents coordinating on different hospital tasks, showing trust scores and communication links.

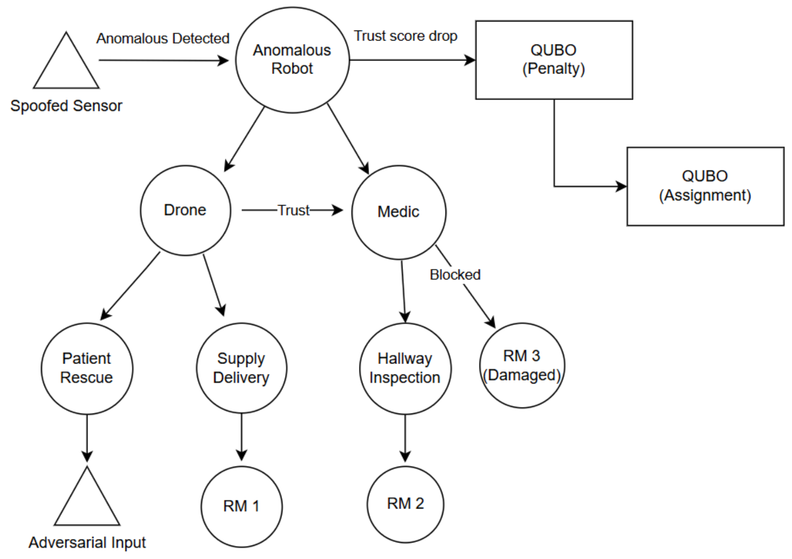


Figure 2. Threat model and mitigation pipeline for trust-aware assignment.
This flowchart describes how adversarial inputs (spoofed sensors) trigger trust degradation, which is detected and fed into a QUBO-based optimizer to enforce secure assignments.

Experimental Design

I conducted two main simulation experiments to study cybersecurity-aware coordination among robotic agents in a post-earthquake hospital setting.

Datasets

- Simulation Trust Data
  - Synthetic dataset of 100 agents over 1000 rounds, with 10% randomly compromised per round and countermeasure recovery logic. Final trust matrix generated as 100x100 agent-task scores using task difficulty modifiers
- QUBO Assignment Data
  - 5x5 agent-task submatrix sampled from the final trust matrix for optimization testing under poisoning vs. defended trust conditions

Baselines

- Trust Simulation
  - Comparing runs without countermeasures vs. with countermeasures
- QUBO Assignment
  - Comparing poisoned vs. defended trust matrices

Metrics

- Average trust per round
- Final trust score distribution
- Total task success and failure counts
- Assignment matrix sparsity

Libraries/Tools

- Python 3
- NumPy
- Matplotlib
- dimod (D-Wave Ocean SDK)

Table

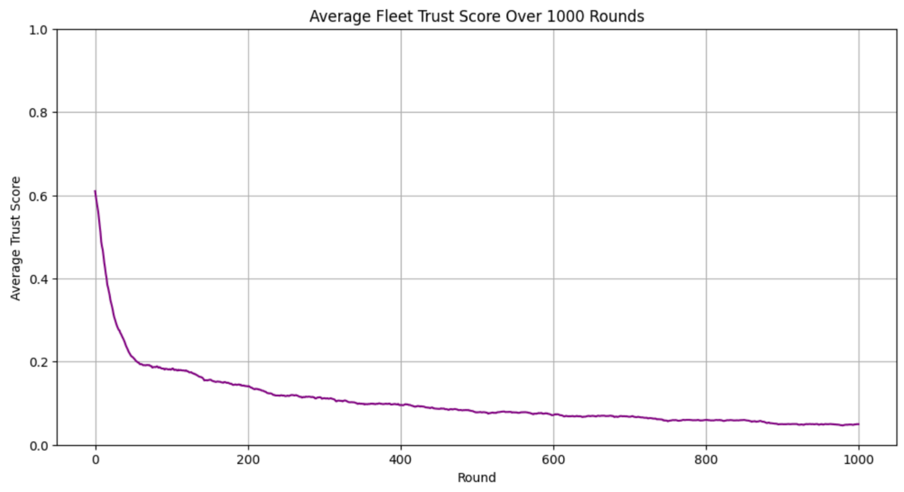
Experiment	**Model**	**Dataset**	**Metric(s)**	**Baseline Comparison**
Exp 1	Trust Simulation	100 agents × 1000 rounds of trust data	Avg. trust, success/failure count	No-defense vs. defense run
Exp 2	QUBO Task Assignment	5×5 trust submatrix (poisoned/defended)	Assignment matrix sparsity/structure	Poisoned matrix vs. Defended matrix

Preliminary Results & Key Findings

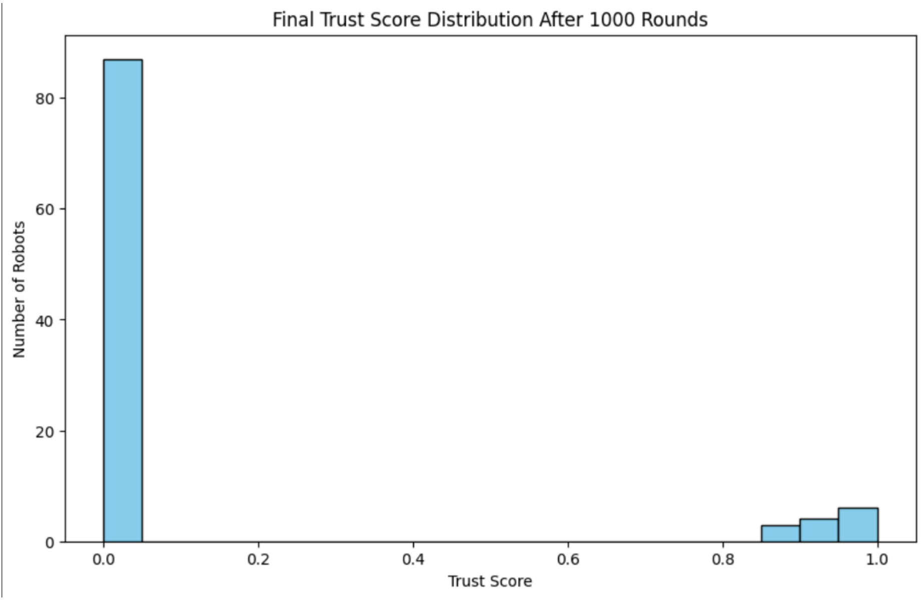
I conducted two main simulation experiments to evaluate the impact of cyberattacks on agent trust and the effectiveness of defenses in enabling resilient task assignment.

5.1 Trust Simulation Results

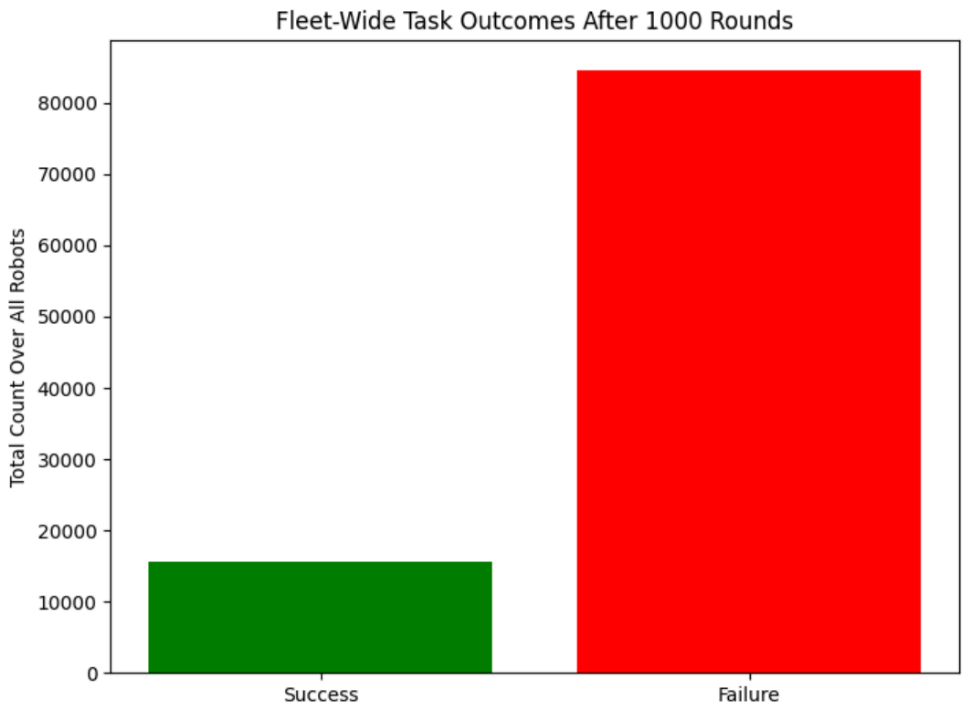
- Without Defense

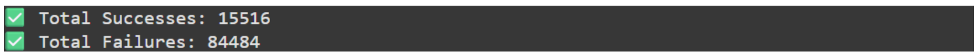


- Figure 1 shows the average trust score over 1000 rounds without any countermeasures. Trust degrades steadily due to repeated compromise events, with many agents falling below 0.3 trust



- Figure 2 shows the distribution of final trust scores, confirming widespread degradation

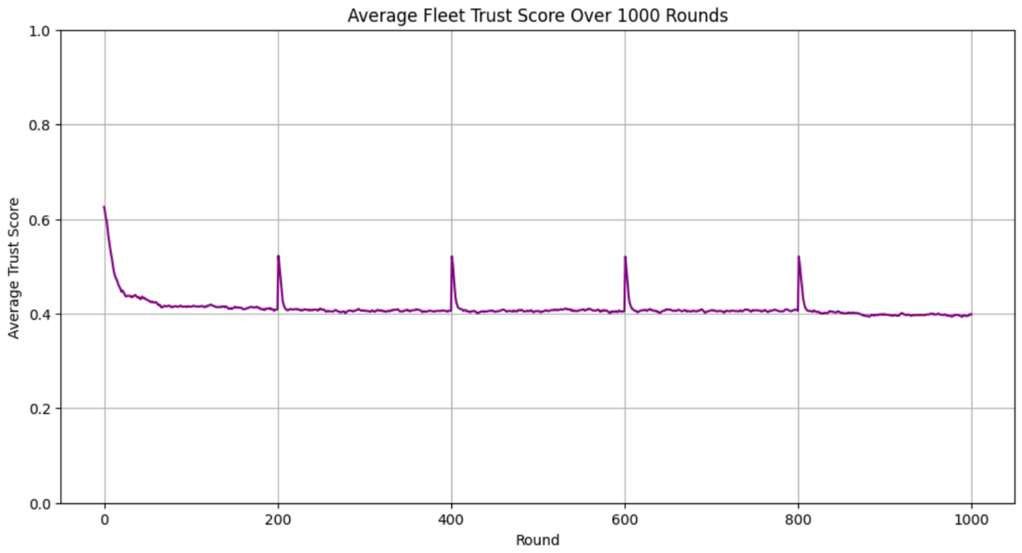




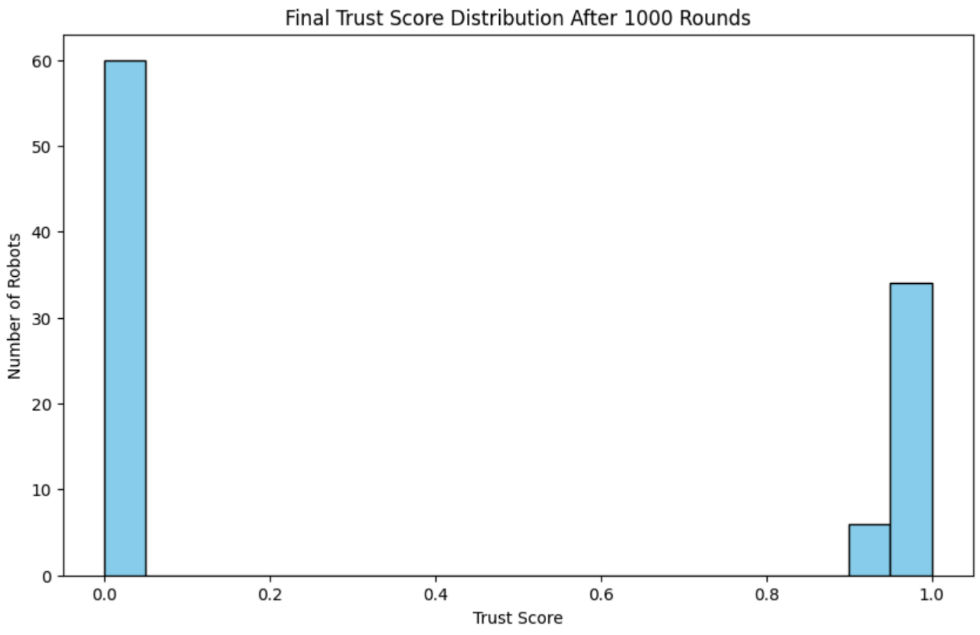
```
✅ Total Successes: 15516
✅ Total Failures: 84484
```

- Figure 3 highlights the higher number of task failures compared to successes

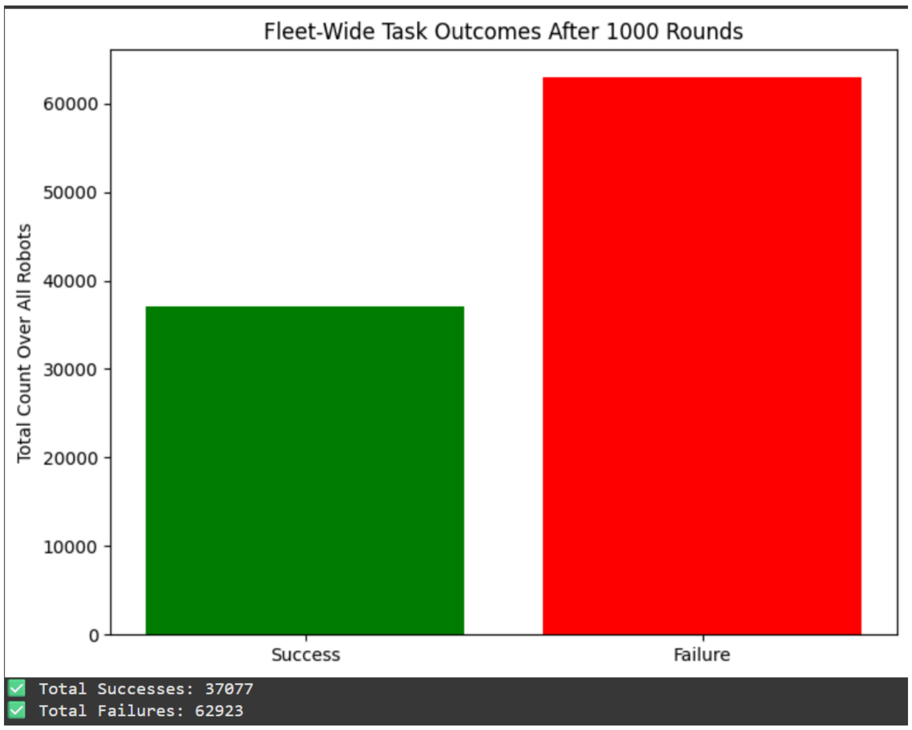
- With Defense



  - Figure 4 demonstrates the average trust score over time with countermeasures enabled. Trust levels remain more stable, with fewer agents dropping to low trust values



  - Figure 5 shows that the distribution of final trust scores shifts right, indicating better resilience under attack

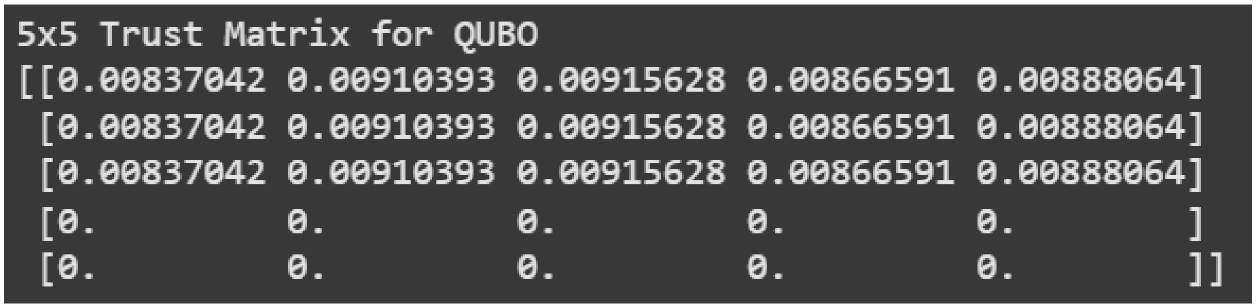


```
✅ Total Successes: 37077
✅ Total Failures: 62923
```

- ○ Figure 6 compares task outcomes, showing reduced failure rates

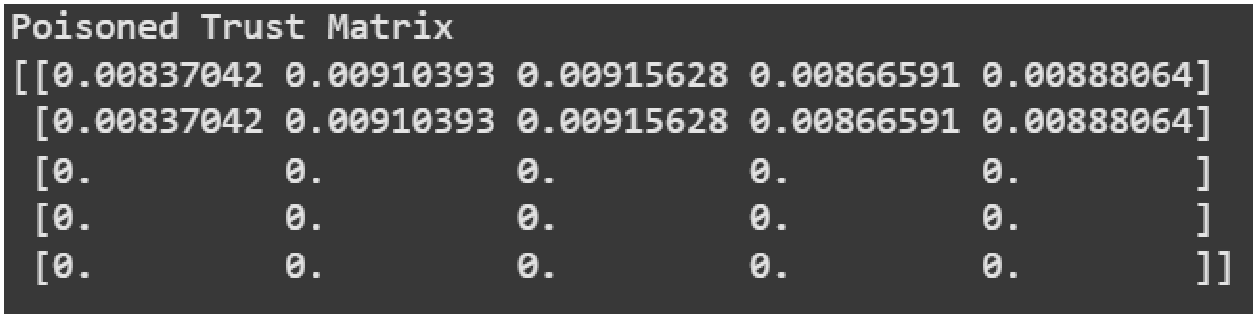
5.2 QUBO Assignment Results

I used the same 100-agent trust simulation described in Section 5.1 to generate the final agent trust scores. These trust scores were transformed into a 100x100 agent-task matrix by applying a task difficulty modifier. Then sampled 5x5 submatrices to test assignment optimization under poisoned versus defended trust conditions using a QUBO solver.



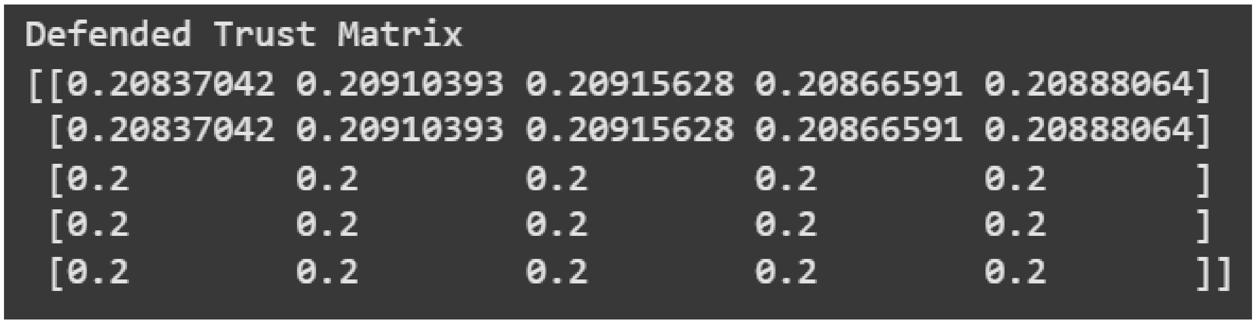
```
5x5 Trust Matrix for QUBO
[[0.00837042 0.00910393 0.00915628 0.00866591 0.00888064]
 [0.00837042 0.00910393 0.00915628 0.00866591 0.00888064]
 [0.00837042 0.00910393 0.00915628 0.00866591 0.00888064]
 [0.         0.         0.         0.         0.        ]
 [0.         0.         0.         0.         0.        ]]
```

- Figure 9: Sampled 5x5 Agent-Task Trust Matrix
    - ○ Constructed a 100x100 agent-task trust matrix by combining agent trust scores with task difficulty modifiers. To make the problem solvable with QUBO, I randomly sampled a 5x5 submatrix representing trust between 5 agents and 5 tasks. Higher values indicate higher confidence in successful assignments



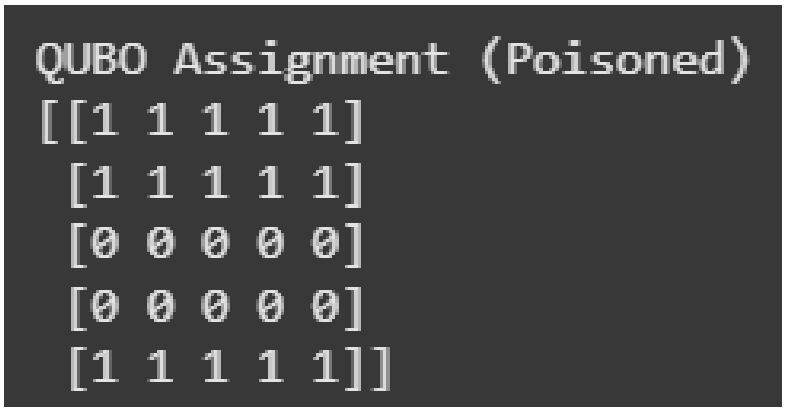
```
Poisoned Trust Matrix
[[0.00837042 0.00910393 0.00915628 0.00866591 0.00888064]
 [0.00837042 0.00910393 0.00915628 0.00866591 0.00888064]
 [0.         0.         0.         0.         0.        ]
 [0.         0.         0.         0.         0.        ]
 [0.         0.         0.         0.         0.        ]]
```

- Figure 10: Poisoned Trust Matrix
    - Simulated a cyberattack by lowering the trust scores of one agent (row) by 0.4. This produced near-zero trust values across all tasks for that agent, modeling adversarial compromise. Poisoning reduces assignment quality and complicates optimization



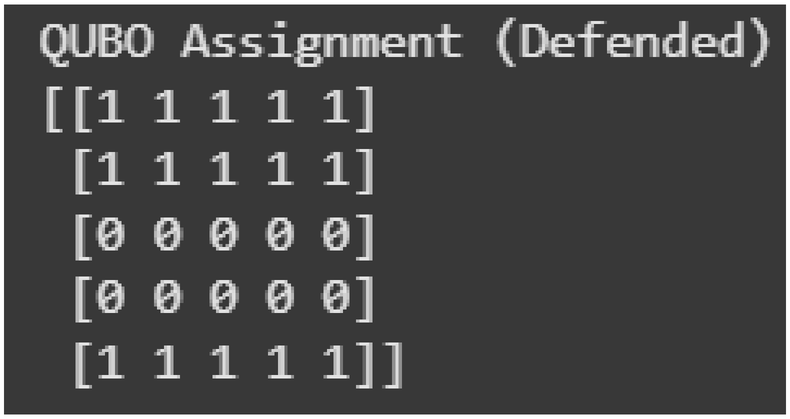
```
Defended Trust Matrix
[[0.20837042 0.20910393 0.20915628 0.20866591 0.20888064]
 [0.20837042 0.20910393 0.20915628 0.20866591 0.20888064]
 [0.2        0.2        0.2        0.2        0.2       ]
 [0.2        0.2        0.2        0.2        0.2       ]
 [0.2        0.2        0.2        0.2        0.2       ]]
```

- Figure 11: Defended Trust Matrix
    - Applied a simple countermeasure by adding 0.2 to all trust scores below 0.3. This partial restoration of trust simulates basic recovery measures, improving the trust landscape even under adversarial conditions



```
QUBO Assignment (Poisoned)
[[1 1 1 1 1]
 [1 1 1 1 1]
 [0 0 0 0 0]
 [0 0 0 0 0]
 [1 1 1 1 1]]
```

- Figure 12: QUBO Assignment (Poisoned Scenario)
    - This assignment matrix shows the QUBO solution under the poisoned trust matrix. Rows with multiple 1’s indicate over-assignment, while rows with all 0’s indicate agents excluded from tasking. Shows the difficulty of enforcing one-to-one constraints under severe poisoning



```
QUBO Assignment (Defended)
[[1 1 1 1 1]
 [1 1 1 1 1]
 [0 0 0 0 0]
 [0 0 0 0 0]
 [1 1 1 1 1]]
```

- Figure 13: QUBO Assignment (Defended Scenario)
    - The defended trust matrix also produced over-assigned patterns in the QUBO output. While the defense improved trust scores, by tuning of penalty weights and scaling remains necessary to achieve valid one-to-one assignments.

Working well

- The trust simulation successfully models degradation under repeated compromise, showing that agent trust drops significantly without countermeasures
- Implementing countermeasure logic improved overall trust stability, producing higher average trust scores and reducing task failures
- The framework shows how cybersecurity defenses can preserve coordination in a post-disaster scenario

Challenges or errors faced

- The QUBO solver results were highly sensitive to scaling and penalty parameters
- Without scaling, trust rewards were too small compared to penalty terms, often showing all-zero assignment matrices
- Excessive scaling led to over-assignment patterns with multiple 1's per row or column
- Even with partial defenses, enforcing strict one-to-one assignment constraints remained difficult

Alignment with hypothesis

- Overall, results support my hypothesis that trust-aware cybersecurity defenses can maintain higher trust and enable more resilient agent-task assignments, even under attack
- Optimization under adversarial conditions remains challenging

Reference

Bojchevski, Aleksandar, and Stephan Günnemann. “Adversarial Attacks on Node Embeddings via Graph Poisoning.” ArXiv.org, 2018, arxiv.org/abs/1809.01093.