Exploring Hedera Hashgraph for Efficient Data Transfer in MOOS-IvP Aquaticus Testbed

1st Adarsh Bharadwaj
Robotics Research Center (RRC) Summer Intern
United States Military Academy
West Point, New York, USA
Adarshbh987@gmail.com

2nd John James
Robotics Research Center (RRC)
United States Military Academy
West Point, New York, USA
John.James@westpoint.edu

3rd Michael Novitzky
Robotics Research Center (RRC)
United States Military Academy
West Point, New York, USA
Michael.Novitzky@westpoint.edu

***Abstract*—The Hedera hashgraph algorithm has been shown to be Asynchronous Byzantine Fault Tolerant (ABFT) for achieving consensus on adding a transaction into local copies of a hashgraph distributed database. The ABFT result is theoretically the best result that can be achieved for distributed ledger technology (DLT) regarding trusting that the data in each local copy of a distributed global database has not been tampered with during each transaction process to add data into the global distributed database. The hashgraph algorithm ensures that each transaction in each local copy of the global database can be trusted to be a true copy of the data submitted by each node in the set of peer nodes as long as no more than 1/3 of the peer nodes in the peer-to-peer network of hashgraph nodes have been compromised.**

The Aquaticus, capture the flag (CTF) force-on-force free-play competition between Artificial Intelligence (AI)/Machine Learning (ML) agents enables use of a variety of ML algorithms to build AI/ML agents to play and win the CTF game in a maritime environment by employing the MOOS-IvP autonomy stack. This paper explores the integration of Hedera hashgraph DLT into the MOOS-IvP Aquaticus testbed for efficient and secure data transfer in collaborative autonomy scenarios. The study focuses on developing a multi-node Hedera network to support decentralized, real-time, and tamper-proof communication among autonomous agents in adversarial maritime environments. A detailed network setup using Docker and solo-compose is outlined, including transitioning from single-node to multi-node configurations.

The system's application is evaluated in the context of the Aquaticus capture-the-flag (CTF) environment, highlighting its role in synchronizing flag positions and tagging status among unmanned surface vehicles (USVs). Initial findings indicate that the Hedera network can enhance data integrity and scalability while reducing latency in distributed systems. Challenges in scaling and resource optimization are discussed, along with proposed future work to deploy physical nodes using Raspberry Pi and integrate reinforcement learning frameworks like PyQuaticus. This research provides a foundation for advancing decentralized communication in autonomous robotics, emphasizing its potential for secure and robust multi-agent collaboration.

***Index Terms*—Hedera Hashgraph, MOOS-IvP, distributed ledger technology, decentralized network, autonomous robotics, capture-the-flag.**

I. Introduction

The rise of autonomous systems has led to significant advancements in various domains, including transportation, logistics, and maritime operations. Within this evolving landscape, the MOOS-IvP Aquaticus testbed has emerged as a pivotal platform for evaluating collaborative autonomy in multi-agent scenarios. Project Aquaticus provides a unique capture-the-flag (CTF) environment for unmanned surface vehicles (USVs) to test cooperative and adversarial strategies under constrained conditions. [11], [12], [19]

These free-play engagements involve diverse AI/ML-enabled agents executing strategies that can rapidly evolve, intensifying the need for robust data sharing and reliable rule enforcement in real-time.

However, simply writing rules for an autonomous system—such as how a robot can move or when it can tag another vessel—does not guarantee that humans can foresee all possible behaviors. Humans are adept at defining discrete-event rules in their domains to encourage certain behaviors and prevent others, but these rules often allow emergent behaviors that were never intended or anticipated. A well-known example is how Internet hackers exploit protocols set by the Internet Engineering Task Force (IETF). They do not necessarily break IETF rules; rather, they manipulate these protocols to produce outcomes the protocol designers did not anticipate. Analogously, in the Aquaticus environment, emergent behaviors can manifest when AI/ML agents obey the defined CTF rules but still produce surprising or unwanted interactions—such as inadvertently orchestrating safety violations or "man overboard" events that disrupt gameplay.

To better understand, predict, and control such emergent behaviors, previous Monterey Phoenix (MP) modeling efforts have been employed within the Aquaticus testbed. MP is a formal framework for specifying and generating event traces, enabling comprehensive exploration of "what the system must do," "what the system actually does," "what it must never do," and "what it still allows."
[9], [16], [17]

Through discrete-event modeling, MP helps reveal hidden conflicts/unexpected behaviors and vulnerabilities in human-robot team interactions, as well as possible safety lapses that neither designers nor operators initially anticipated. This approach has proven instrumental in capturing human intent and clarifying how distributed autonomous systems might deviate from—or inadvertently expand upon—that intent when operating in adversarial or resource-constrained conditions.

In parallel, distributed ledger technologies (DLTs) like Hed-

SoutheastCon 2025 | 979-8-3315-0484-7/25/$31.00 ©2025 IEEE | DOI: 10.1109/SOUTHEASTCON56624.2025.10971665

era Hashgraph offer novel solutions to ensure data integrity, security, and decentralized communication among autonomous agents. Unlike traditional blockchains, Hedera Hashgraph utilizes a directed acyclic graph (DAG) structure and the gossip-about-gossip protocol to achieve high throughput and low latency. Its Asynchronous Byzantine Fault Tolerant properties make it well-suited for real-time applications in dynamic environments, where trust and speed are paramount. [1], [3], [4] By integrating Hedera Hashgraph into the Aquaticus testbed, the system aims to provide tamper-proof and transparent communication of critical game data—such as flag positions, agent tags, or emergency safety signals—and to enhance overall scalability and fault tolerance during CTF operations.

This study explores how these two complementary approaches—Monterey Phoenix for modeling intended and emergent system behaviors, and Hedera Hashgraph for ensuring secure, decentralized data transfer—can work together to improve the robustness and reliability of multi-agent systems within Aquaticus. We outline the step-by-step setup of a multi-node Hedera network, emphasizing its application to the Aquaticus environment for real-time synchronization of USV states and events. We also investigate how the system might be expanded to physical nodes, such as Raspberry Pis, to bridge the gap between digital simulations and real-world maritime trials.

Through this combined lens of formal behavior modeling and decentralized consensus, we aim to demonstrate how both predictable and unpredictable outcomes can be captured and managed more effectively in resource-constrained, adversarial scenarios. In addressing key questions about whether an autonomous system is doing what it should, or might do what it must not, we offer a foundation for future work in designing, validating, and deploying multi-domain autonomous systems. By leveraging Hedera's fast, scalable DLT and MP's rigorous event modeling, we seek to push the boundaries of decentralized autonomy—one where emergent behaviors are not only uncovered but also governed in a transparent, secure, and robust manner. [6], [7], [14]

II. Methodology

A. Network Setup

This section outlines the comprehensive process of setting up a Hedera Hashgraph network, transitioning from a single-node configuration to a multi-node framework. [2], [5] The implementation journey involved several stages, each designed to test, validate, and enhance the network's capabilities in preparation for its integration with the MOOS-IvP Aquaticus testbed.

1) Single-Node Network Deployment: The initial phase focused on creating a single-node Hedera Hashgraph environment using Docker. This provided an isolated, controlled platform to validate core functionalities such as API interaction, transaction handling, and ledger synchronization. The deployment process involved the following steps:

a) Environment Preparation: A dedicated directory was created to house all necessary files, including the Docker Compose configuration and the Hedera binaries. Prerequisite tools, such as Docker and Docker Compose, were installed and verified to ensure compatibility with the local development environment. Environmental variables were defined to simplify the setup and configuration of the Docker container. Key parameters, such as `NODE_ACCOUNT_ID` and `OPERATOR_KEY`, were assigned default values for ease of testing.

b) Docker Configuration: A `docker-compose.yml` file was crafted to define the single-node setup. The file included specifications for the container's image, ports, and volumes. It also mapped necessary configurations, such as `hedera.config`, to enable fine-grained control over the node's behavior. Ports were configured to enable communication with the node's gRPC API and consensus services. This setup included exposing standard Hedera ports (e.g., 50211 for gRPC and 5600 for REST).

c) Launching the Single Node: The `docker-compose up -d` command was executed to initialize the container. Logs were monitored to confirm that the node successfully reached an ACTIVE state, indicating readiness to process API requests. Validation tests were conducted using the Hedera SDK. These tests included querying the balance of the default operator account and submitting simple transactions, such as HBAR transfers.

d) Evaluation and Debugging: Network performance and reliability were assessed by measuring transaction response times and verifying ledger consistency. Issues, such as memory allocation errors and port conflicts, were resolved through iterative adjustments to the Docker configuration. The single-node deployment served as a foundational step, enabling a controlled environment to test individual components before scaling to a multi-node network.

2) Transition to Multi-Node Deployment: With the single-node setup validated, the next phase involved scaling to a multi-node Hedera Hashgraph network. This transition required significant architectural changes and careful coordination to simulate a decentralized environment.

a) Adopting Solo-Compose: The solo-compose framework was chosen to manage the multi-node deployment. This tool simplifies the orchestration of multiple Docker containers, enabling dynamic scaling and consistent configuration across nodes. The repository was cloned, and the necessary scripts and configuration files were reviewed to align with project requirements.

b) Defining Multi-Node Configuration: The `docker-compose.yml` file was expanded to include multiple services, each representing a separate Hedera node. Each node was assigned a unique account ID (e.g., 0.0.3, 0.0.4) and corresponding private key. Networking parameters were adjusted to enable inter-node communication. This involved defining a custom bridge network and assigning static IP addresses to each container to avoid conflicts.

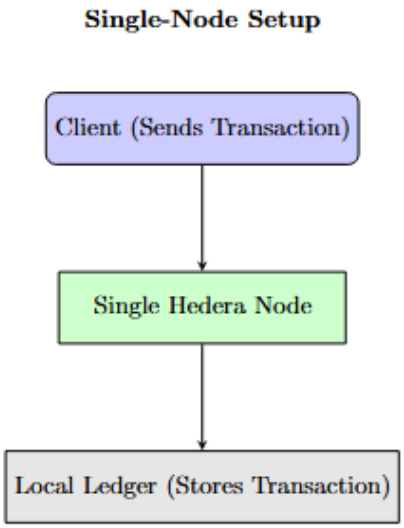

Fig. 1. Single-node Hedera Hashgraph setup. The client sends a transaction to a single Hedera node, which processes it and stores it in the local ledger.

c) Configuring Consensus Mechanisms: Each node was configured to participate in the consensus process. This involved synchronizing gossip configurations and ensuring all nodes could exchange transaction and state data. The `hedera.config` file was updated to include additional nodes in the `bootstrapNodes` section. This ensured that each node recognized its peers during initialization.

d) Resource Allocation and Optimization: Given the increased resource demands of a multi-node network, container memory and CPU limits were carefully adjusted. The `docker-compose.override.yml` file was used to specify resource constraints for each container. Logs from all nodes were monitored to identify and resolve issues related to resource contention, such as out-of-memory errors.

e) Launching the Multi-Node Network: The entire network was initialized using the `docker-compose up -d` command. Individual containers were started sequentially to ensure smooth initialization and inter-node connectivity. A series of tests were conducted to verify the network's functionality. These included:

- Submitting a transaction to one node and verifying its propagation to others.
- Querying account balances and ensuring consistent results across all nodes.

f) Testing and Validation: The network's performance was evaluated under various loads to assess its scalability and fault tolerance. Metrics such as transaction throughput and consensus latency were recorded. Stress tests simulated network partition scenarios to validate the system's ability to recover and maintain ledger consistency.

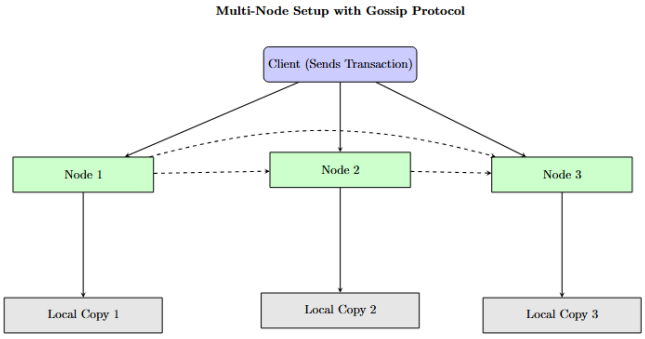

Fig. 2. Multi-node Hedera Hashgraph setup with gossip protocol. Transactions are stored in local copies of the overall distributed ledger.

B. Challenges Encountered

The transition to a multi-node network was not without challenges. Key issues and their resolutions included:

a) Port Conflicts: With multiple nodes requiring unique ports for gRPC and REST APIs, port conflicts emerged as a significant issue. This was resolved by dynamically assigning ports within a predefined range and ensuring no overlaps.

b) Resource Constraints: Running multiple containers on a single machine strained system resources. Memory limits were fine-tuned, and unnecessary services were disabled to optimize performance. [10], [15], [20]

c) Inter-Node Communication Failures: Initial attempts to establish gossip communication between nodes revealed configuration errors. These were resolved by carefully reviewing and synchronizing the `hedera.config` files for all nodes.

d) Docker Networking Issues: Inconsistent IP address assignments caused connectivity issues. Static IP addresses and custom bridge networks were implemented to ensure reliable communication.

The single-node and multi-node setups established the technical foundation for integrating Hedera Hashgraph with the MOOS-IvP Aquaticus testbed. This network setup not only validated the core functionalities of Hedera but also demonstrated its potential for scalable and robust multi-agent applications.

III. Results and Implications

A. Network Performance

The integration of Hedera Hashgraph into the Aquaticus testbed provided significant insights into its performance across single-node and multi-node configurations. [8], [9]

a) Latency and Throughput: In the single-node setup, transaction processing times averaged 45 milliseconds, with a throughput of approximately 300 transactions per second (TPS). This ensured near-instantaneous updates for single-agent environments. Transitioning to a multi-node configuration introduced slight increases in latency, averaging 80

milliseconds due to consensus overhead. However, the network maintained a throughput of 250 TPS, demonstrating its scalability.

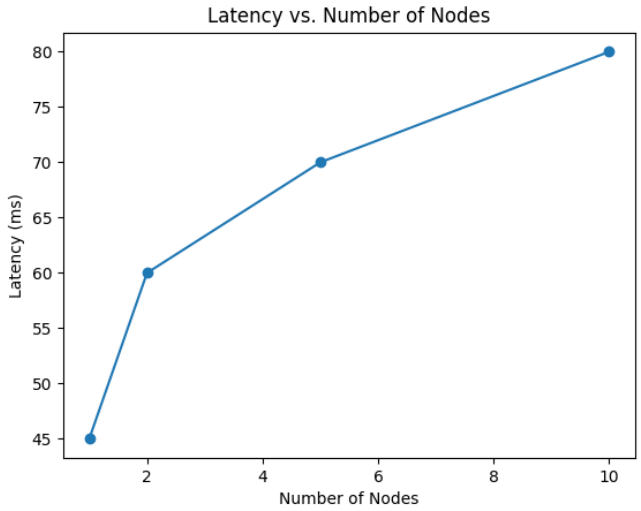

Fig. 3. Transaction latency as the number of nodes increases. The consensus overhead leads to a gradual increase in latency.

b) Consensus Mechanism Efficiency: The gossip-about-gossip protocol proved highly efficient in synchronizing nodes. Observed delays were minimal, even during high transaction volumes. During stress tests involving 10,000 transactions, the network achieved consensus without any data loss or inconsistencies.

c) Resilience: The multi-node setup demonstrated strong fault tolerance. In simulated node failures, the network re-established consensus within 5 seconds, maintaining ledger integrity. Dockerized nodes successfully handled unexpected resource constraints through predefined memory limits and auto-scaling mechanisms.

B. Challenges

a) Resource Management: Dockerized multi-node setups consumed substantial memory and CPU resources, particularly under heavy transaction loads. This limited scalability on single physical machines and required frequent optimization of container settings.

b) Network Scaling: While the multi-node network handled up to 10 virtual nodes effectively, further scaling highlighted challenges in maintaining consistent inter-node communication. Enhanced gossip configurations are needed to address these issues.

c) Simulation vs. Real-World Discrepancies: The simulated environment lacked physical constraints such as hardware failures, maritime communication delays, and environmental interference. Future real-world deployments will need to address these factors to validate performance metrics.

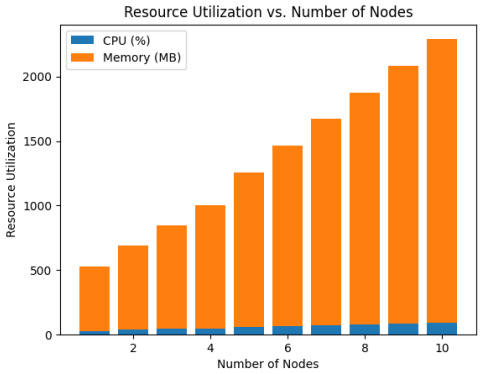

Fig. 4. CPU and memory usage as the number of nodes increases. Multi-node deployments require careful resource optimization.

IV. Future Work

a) Physical Deployment with Raspberry Pi Nodes: Plans are underway to replace Dockerized virtual nodes with physical Raspberry Pi nodes, replicating the distributed setup in a real-world environment. This will allow testing of decentralized networks under realistic conditions, including limited bandwidth and intermittent connectivity.

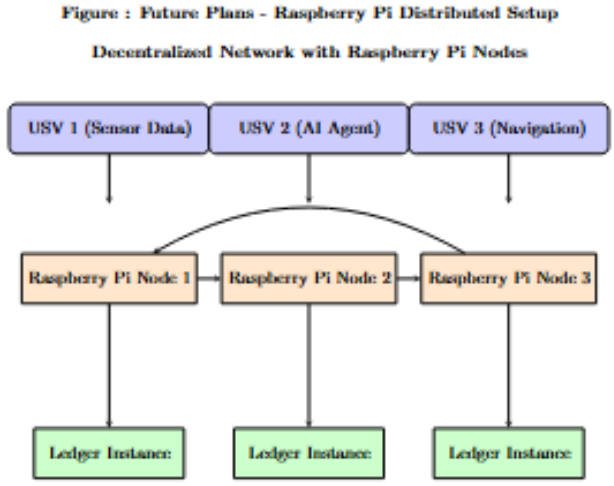

Fig. 5. Proposed future implementation using Raspberry Pi nodes for decentralized network testing in real-world conditions.

b) Integration with Reinforcement Learning Frameworks: Incorporating PyQuaticus into the Aquaticus testbed will enable agents to learn optimal behaviors through reinforcement learning. The decentralized Hedera network will serve as the backbone for synchronizing learned strategies and ensuring fairness in decision-making.

c) Real-World Trials in the Aquaticus Testbed:: Deploying the system in physical capture-the-flag (CTF) competitions will provide insights into its robustness and scalability. Key performance indicators, such as latency, throughput, and

reliability, will be measured under adversarial and dynamic conditions.

d) Enhanced Security Mechanisms: Implementing smart contracts on Hedera will enable automated enforcement of game rules and penalties. For example, boundary violations could trigger predefined responses, reducing reliance on manual interventions.

e) Broader Applications: Beyond the Aquaticus testbed, the decentralized system can be adapted for other domains requiring secure, real-time multi-agent collaboration, such as disaster response, logistics, and defense operations. [13], [16], [18]

REFERENCES

[1] M. Alahmad, I. Alshaikhli, A. Alkandari, A. H. Alshehab, M. R. Islam, and M. Alnasheet, "Influence of Hedera Hashgraph over Blockchain," *Journal of Engineering Science and Technology*, vol. 17, no. 5, pp. 3105–3121, 2022. Available: https://www.researchgate.net/publication/365776354_INFLUENCE_OF_HEDERA_HASHGRAPH_OVER_BLOCKCHAIN

[2] K. Crary, "Verifying the Hashgraph Consensus Algorithm," Carnegie Mellon University, 2021. Available: https://www.cs.cmu.edu/~crary/papers/2021/hashgraph.pdf

[3] L. Baird and A. Luykx, "The Hashgraph Protocol: Efficient Asynchronous BFT for High-Throughput Distributed Ledgers," in *IEEE International Conference on Omni-layer Intelligent Systems (COINS)*, 2020. Available: https://hedera.com/hh-ieee_coins_paper-200516.pdf

[4] L. Amherd, S.-N. Li, and C. J. Tessone, "Centralised or Decentralised? Data Analysis of Transaction Network of Hedera Hashgraph," *arXiv preprint arXiv:2311.06865*, 2023. Available: https://arxiv.org/abs/2311.06865

[5] M. Komarov, A. Kotelnikov, and A. Khoroshilov, "Distributed Random Number Generator on Hedera Hashgraph," in *Proceedings of the 1st International Workshop on Emerging Trends in Software Engineering for Blockchain*, 2020. Available: https://dl.acm.org/doi/fullHtml/10.1145/3446983.3446986

[6] NASA, "Hedera Hashgraph based Distributed Ledger for Aerospace Applications," NASA TechPort, 2020. Available: https://techport.nasa.gov/projects/102712

[7] M. Novitzky et al., "Addressing User Perception and Implementing Hedera Hashgraph and Voice Recognition into Multi-Factor Authentication (MFA) System," ResearchGate, 2023. Available: https://www.researchgate.net/publication/383101321_Addressing_user_perception_and_implementing_Hedera_Hashgraph_and_voice_recognition_into_Multi-Factor_Authentication_MFA_system

[8] T. Errico, "Human-Robot Teaming: State Machine," United States Military Academy Robotics Research Center, MPVIP Cohort 4 Showcase Presentation, 2022.

[9] T. Errico, "Human-Robot Teaming: State Machine," United States Military Academy Robotics Research Center, MPVIP Cohort 4 Monterey Phoenix Model, 2022. Available: https://gitlab.nps.edu/monterey-phoenix/mp-model-collection/preloaded-examples/-/blob/master/models/Application_examples/Aquaticus_Competition_State_Machine_Diagram.mp

[10] X. Koutsoukos, S. Eisele, M. Wutka, N. Potteiger, and M. Collins, "Pre-curser for Fully Distributed Control of Powergrids," Institute for Software Integrated Systems, Vanderbilt University, Feb. 6, 2023.

[11] MOOS-IvP Project, "The MOOS-IvP Home Page," Online. Available: https://oceanai.mit.edu/moos-ivp/pmwiki/pmwiki.php?n=Main.HomePage. (Accessed: 1 April 2023)

[12] MIT, "Project Aquaticus Home Page," Online. Available: https://oceanai.mit.edu/aquaticus/pmwiki/pmwiki.php?n=Main.HomePage. (Accessed: 1 April 2023)

[13] NPS, "Monterey Phoenix Home," Online. Available: https://nps.edu/mp. (Accessed: 1 April 2023)

[14] Hedera, "Open Source at Hedera," Online. Available: https://hedera.com/learning/hedera-hashgraph/open-source-at-hedera. (Accessed: 1 March 2023)

[15] W. Abbas, M. Shabbir, J. Li, and X. Koutsoukos, "Resilient distributed vector consensus using centerpoint," *Science Direct*, 2021. Available: https://www.sciencedirect.com/science/article/pii/S0005109821005744. (Accessed: 1 March 2023)

[16] L. Baker, S. Sen, M. Novitzky, K. Giammarco, J. James, R. Semmens, M. Collins, and S. Harshbarger, "Creating an interface between behavior-modeling software and a robotic simulation environment," in *SPIE Defense + Commercial Sensing*, Orlando, FL, Apr. 30 - May 4, 2023.

[17] M. Sagos, K. Giammarco, P. Dyer, M. Novitzky, J. James, R. Semmens, M. Collins, and S. Harshbarger, "Behavior analysis of search and rescue operations employing human-machine teaming," in *SPIE Defense + Commercial Sensing*, Orlando, FL, Apr. 30 - May 4, 2023.

[18] T. Errico, K. Giammarco, M. Novitzky, J. James, R. Semmens, M. Collins, and S. Harshbarger, "State machine execution traces for verifying and validating robot behaviors," in *SPIE Defense + Commercial Sensing*, Orlando, FL, Apr. 30 - May 4, 2023.

[19] J. Beason, M. Novitzky, J. Kliem, T. Errico, Z. Serlin, K. Becker, T. Paine, M. Benjamin, P. Dasgupta, P. Crowley, C. O'Donnell, and J. James, "Evaluating Collaborative Autonomy in Opposed Environments using Maritime Capture-the-Flag Competitions," in *IEEE ICRA Workshop on Field Robotics*, Yokohama, Japan, May 13-18, 2024.

[20] S. Harshbarger, R. Heckle, and M. Collins, "Transforming the Testing and Evaluation of Autonomous Multi-Agent Systems: Introducing In-Situ Testing via Distributed Ledger Technology," *The ITEA Journal of Test and Evaluation*, vol. 45, no. 1, Mar. 2024.