2024 IEEE 14th Annual Computing and Communication Workshop and Conference (CCWC) | 979-8-3503-6013-4/24/$31.00 ©2024 IEEE | DOI: 10.1109/CCWC60891.2024.10427933

Assuring Safe Navigation and Network Operations of Autonomous Ships

Luis Rivas
Information Security Institute
Johns Hopkins University

Spencer Stevens
Information Security Institute
Johns Hopkins University

Andrew Zitter
Information Security Institute
Johns Hopkins University

Vinayak Khandelwal
Information Security Institute
Johns Hopkins University

Amodini Vardhan
Information Security Institute
Johns Hopkins University

Chinmay Lohani
Information Security Institute
Johns Hopkins University

Chris Rouff
JHU Applied Physics Laboratory
Johns Hopkins University

Lanier Watkins
Information Security Institute
Johns Hopkins University

***Abstract*—Autonomous marine vehicles face escalating cybersecurity threats as connectivity and automation increase attack surfaces. Cybercriminals have increased security intrusions targeting the marine industry by 400% during the COVID-19 epidemic. This research is a work-in-progress study investigating the feasibility of developing a modular battleship simulator and a simultaneous navigation and security artificial intelligence (AI) monitoring system with machine learning (ML) technologies. We used a simple but representative threat model based on GPS spoofing and integrity attacks against the weapons system and ICS network operations. Results indicate the feasibility of the modular battleship simulator and the associated AI monitoring system. The simulator utilizes maritime vehicle physics, rudimentary weapons systems operations, and ICS network operations. Most importantly, we demonstrate the ability of AI monitors to guard against navigation and network operation anomalies that would jeopardize a ship and/or its associated mission.**

***Index Terms*—Industrial Control Systems (ICS), Autonomous Navigation, Battleship Simulator (BSS), Autonomy Assurance, Machine Learning (ML), Artificial Intelligence (AI)**

I. Introduction

Cyberspace is increasingly integrated with terrestrial, maritime, and airborne domains. The convergence phenomenon boosts human flourishing but poses cyber risks. National security depends on robust governance structures to enable autonomous vehicles to function safely, securely, and resiliently, alone or in groups. Recent cases have shown autonomous systems' vulnerability to hackers, data theft, and ransomware. Malware intrusion detected in global operations in 2017 caused a financial loss of up to $300 million [1]. Ransomware assaults hit French shipping operator CMA CGM and Norwegian marine classification group DNV [1]. Internet-connected sensors, navigation components, and Industrial Control Systems (ICS) on naval and commercial vessels increase the assault surface for determined adversaries.

In this study, we simulate and analyze a battleship's cyber-physical systems using artificial intelligence (AI) monitors for navigation and security. To support this simulation, we developed an open-source Battleship Simulator (BSS), which models naval surface ships, executes cyber attacks on those ships, and collects and reports data. It models engines, radar, weaponry, and low-level control logic as parts of an ICS network. Using these features, we created AI models to detect anomalies and influence the navigation systems.

Our key contributions include the feasibility investigation of a (1) modular and extensible maritime vehicle simulator and (2) a simultaneous navigation and security AI monitoring system for maritime vehicles. The maritime vehicle simulator includes physics and navigation simulation, ICS network simulation, and anomaly detection for both navigation and ICS network.

This paper describes Related Works in Section 2 and analyzes the BSS architecture with the ICS Network and Weapon System in Section 3. In Section 4, cyber threats and a Spoofing, Tampering, Repudiation, Information Disclosure, Denial of Services, and Elevation of Privileges (STRIDE) threat model for cyber attacks on battleships are briefly discussed [2]. Sections 5 and 6 cover the AI Navigation Monitor (AI-NM) and Security Monitor ML (SM-ML) training, feature engineering, and modeling. Section 7 discusses experimental evaluation and techniques, Section 8 examines the results and discussion, and Section 9 discusses limitations. Finally, Section 10 concludes with a summary and future work.

II. Related Works

G. Kavallieratos et al. [5] highlighted cyber-attacks on autonomous ships, including GPS spoofing, Denial of Service (DOS), and network attacks. Similarly to [3], we modeled attacks in this research to simulate GPS spoofing by providing false coordinates and tampering with the BSS location. In our work, the impact of such attacks is reflected in the system's behavior, and the SM-ML module highlights the security concerns if an attack is detected.

J.M Szatkowski et al. in [4] designed a Software-Defined Networking to manage the NAVAL Supervisory Control and Data Acquisition (SCADA) Network to enhance the system's scalability and security through a programmable modular generator. The framework uses a similar modular concept, taking advantage of the asynchronous message queuing technology to inform a downstream effect on the ship's security. We believe this approach enables faster communication exchange between components.

979-8-3503-6013-4/24/$31.00 ©2024 IEEE

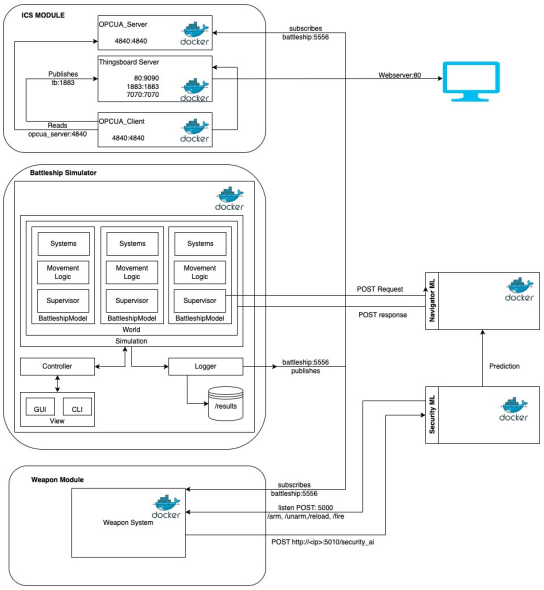

Fig. 1. Battleship Simulation Framework: This diagram illustrates the BSS, key components, ICS Network, Weapons Network, Battleship Simulator, and AI Navigation Monitor and Security Monitor Machine Learning module.

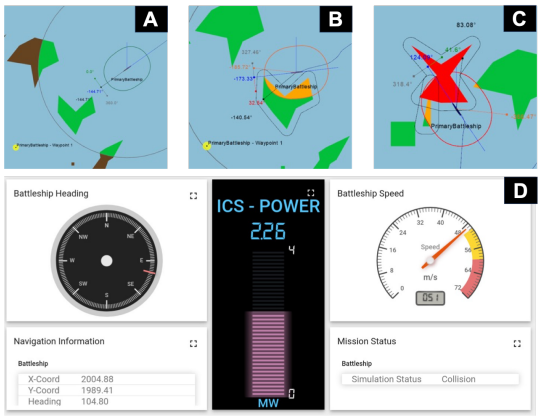

Fig. 2. BSS Status Panel and Collision Example. (A) The battleship observes obstacles in radar range (green), but none are close enough to pose an immediate threat. (B) As the battleship moves towards Waypoint #1, obstacles come within the "minimum safe area" and pose an immediate threat. (C) The battleship collides with an obstacle (red), and the scenario is considered a failure. The AI monitors are designed to avoid this scenario. (D) The *OPC UA* Server Dashboard collects information regarding Speed, Power, Coordinates, and AI Status.

Several other works have similar elements to our BSS, but overall, we believe our approach has advantages in that it also includes integrated navigation and security monitors with active capabilities. The works are as follows: (i) The AIS-driven target ship server developed by Y. Suo et al. [6] enables realistic traffic simulation in maritime navigation; (ii) according to Hasegawa et al. [9], an intelligent ship-handling simulator can avoid collisions, (iii) R. Zaccone and M. Martelli et al. [10] developed a collision avoidance algorithm for autonomous ship guiding using a modified Rapidly-exploring random trees (RRT*) technique, and (iv) the modular simulation technique in [14] generates scenarios and evaluates algorithms for cyber-physical security analysis.

Our platform differs from cyber frameworks like [7] and [8] because it focuses on the complex interactions between a battleship's cyber-physical systems. It combines waypoint-based movement and collision avoidance with AI models identifying physical or cyber-based threats and informing navigation components. CyberBattleSim and Cyber Operations Research Gym examine how AI, and people interact in business networks and cyber-adversarial situations [7], [8]. Our work, on the other hand, looks at the complicated maritime security scene where cyber and physical domains meet. It can be used as a research tool to learn about and improve how well military ships can handle a wide range of threats in a controlled, simulated setting.

III. Battleship Simulator

The experimental framework uses *Python 3.9+* and third-party libraries to provide modeling, analysis, and simulation capabilities. The communications architecture described in Fig. 1 facilitates modularity by employing microservice-based architecture for various functions such as scenario orchestration, environment representation, vehicle control logic, communications, data logging/transmission, and a command line/graphical user interaction, where these components are containerized using *Docker*-technologies. The Open Platform Communications Unified Architecture (OPC UA) and Message Queuing Telemetry Transport (MQTT) Framework facilitate communication within industrial controls. The machine learning module incorporates many algorithms, such as decision trees (DT), random forests (RF), and reinforcement learning, utilizing the *Scikit-Learn* and *Flask* libraries. This modular simulation, system security, and AI integration facilitate the research of ensured autonomous marine vehicle operations.

A. Battleship Operation

BattleshipSimulator, an open-source library, models and predicts battleship threats using machine learning. Fossen [11], [12] introduced physics-based movement modeling based on ship motion control concepts. Fig. 2 shows the simulator autopilot navigating pathways and avoiding collisions.

BSS has collision avoidance, a simple algorithm to turn away from objects, and waypoint tracking battleship navigation that can be overridden. Testing settings use procedural terrain generation with random waypoints and obstacles. BSS modular ICS network decoupling allows independent physics modeling. High-fidelity battleship modeling and modular architecture enable extensible autonomous marine vehicle security evaluation.

B. ICS Network

An ICS network is required to reproduce attacks on the BSS and improve data authenticity (Fig. 2). An *OPC UA* requires ICS data, facilitating the telemetry exchange. In network-constrained situations, *MQTT* offers efficient and lightweight communications. Using *Python* modules such as

'opcua' for *OPC UA* server and 'paho.mqtt' for *MQTT* client, the module seamlessly integrates with BSS. Data is exchanged and synchronized between the logger function and *ZeroMQ* as the MQTT server, guaranteeing simulator integrity and responsiveness. Dual technology integrates modules into the BSS system, allowing internal and external communication. The modular design is tested with a power generation module in the ICS network.

BSS uses the ICS power generator module to mimic power production. This module calculates the battleship power based on the ship's speed, drag, and navigation data. The power-generating computational formula considers drag coefficient, water density, cross-sectional area, and battleship velocity. The power calculation utilizes fluid mechanics coefficients to model naval power dynamics. The model applies a seawater density of 1,025 kg/m^3, a drag coefficient of 0.035, and a cross-sectional area of 1005 m^2 [17].

A *Python* script calculates the simulated power required for the battleship's movement by first computing the drag force acting on the vessel (1):

$$force_{drag} = 0.5(c_d)(\rho)(area)(speed) \tag{1}$$

where c_d represents the drag coefficient, ρ is the density of seawater, area (m^2) is the cross-sectional area facing the fluid flow, and speed (m/s) is the ship's speed in meters per second. Subsequently, the power is determined by multiplying this drag force by the ship's speed, providing the power in watts needed to overcome the drag and maintain the current speed. This module replicates power output changes and creates a realistic environment for analyzing and forecasting the SM-ML features. The SM-ML needs this module to detect cybersecurity and operational issues that can affect power emissions in a battleship.

C. Weapon System

The weapons simulator simulates a system that takes manual commands and decides actions based on certain environment variables. Just as a Close-In Weapon System (CIWS) used to defend against anti-ship missiles and other close threats, the emulated weapon system can decide when to fire based on the real-time distance between the battleship and the target, based on the coordinates provided by the navigation system.

The simulation considers weapon range to simulate real-life systems and arm/disarm/fire based on the change in value of the range. The distance to the target is calculated by a *Python* script that considers the battleship and targets current `x` and `y` coordinates using the formula (2):

$$\text{distance} = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2} \tag{2}$$

Fig. 3 depicts the logic of actions to take based on the distance to the target. The module allows user input using an Application Programming Interface (API), using HTTP POST requests to update the SM-ML with data from other modules. This makes it crucial to verify if there is any cybersecurity-related abnormality and, based on observation, take decisions

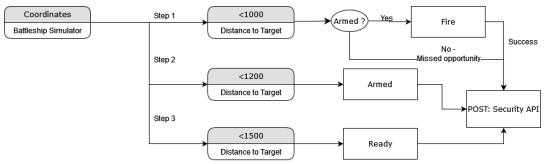

Fig. 3. Weapon System Automated Firing Logic Diagram

to change the course of action. This modularity increases the simplicity of simulating cyber attacks against the weapons system according to a STRIDE threat model.

IV. Threat Model

Advancements in AI have led to an increasing number of autonomous systems and controls in many areas. This paper focuses on autonomous naval vehicles and how these systems are vulnerable to cyber-attacks. This problem is well researched, with several threat models and cyber intrusions performed as in [5].

A. Attack Background

According to the International Association of Classification Societies (IACS) recommendation on cyber resilience [16], functional failures related to the control system's operation, propulsion, and steering can cause high integrity and availability risks. Thus providing the reason that the cybersecurity of a ship must be considered. There are many attacks against the cyber systems of naval vessels, including GPS attacks, DoS attacks, Automatic Identification System (AIS) attacks, and Electronic Chart Display and Information System (ECDIS) attacks [3]. While there are many other attacks, this paper will focus on GPS and firmware-based attacks. These attacks have proven possible and can be easy to exploit. GPS signals from orbiting satellites are weak enough for attackers to interfere or overpower. GPS spoofing will impact navigation and cause sequence problems, such as late or missed shots of the weapon system. Firmware-based attacks are typically performed using malicious programs that manipulate hardware in an unauthorized manner. The key examples related to this paper are power control ICS and weapon targeting systems, which could be spoofed or manipulated with malware.

B. STRIDE Threat Model: Current and Potential Attacks

The STRIDE Threat Modeling risk assessment demonstrates the modular capabilities implemented in this paper, similar to [5]. This assessment describes some current and potential attacks the simulator can model. Table I depicts the STRIDE Threat Model, showing a variety of potential attacks given a threat vector.

The actual attacks used in this paper were to spoof the GPS data for the vessel; this was simulated by changing the location/speed of the BSS. For the weapon targeting system, a tampering attack was simulated by altering the parameters of the weapon system. Finally, another tampering attack was simulated by altering power ICS data for the power control ICS.

TABLE I
STRIDE THREAT MODEL

Threat Category	Security Tenet Violated	Definition
Spoofing	Authenticity	An adversary capability to mask as another entity or pass data as authentic
Tampering	Integrity	An adversaries capability to alter data inside the system
Repudiation	Non-repudiation	The inability to attribute actions with evidence
Information Disclosure	Confidentiality	Revealing information to those without proper permissions
Denial of Service	Availability	Taking down the system and removing the availability/operation of the system
Privilege Escalation	Authority	Executing unauthorized protocols or programs

TABLE II
FEATURE DESCRIPTIONS FOR AI NAVIGATION MONITOR

Description of Selected Features
desired_heading: Direction battleship is trying to move towards
option_port: Difference between current heading and desired heading
out_of_bounds: Indicates ship movement outside simulation boundary
RadarSonar.collision_warning: Warning for potential collisions
RadarSonar.collision_event: Indicates collision occurrence
Engine.desired_speed: Desired speed of the battleship
Distance_to_point_1: Distance to waypoint 1
Distance_to_point_2: Distance to waypoint 2
Distance_to_point_3: Distance to waypoint 3
Distance_to_point_4: Distance to waypoint 4
Distance_to_radar_obj_1: Distance to object 1 detected by radar
Distance_to_radar_obj_2: Distance to object 2 detected by radar
Distance_to_radar_obj_3: Distance to object 3 detected by radar
Distance_to_radar_obj_4: Distance to object 4 detected by radar
Direction_port: Direction for next move (port)
Direction_starboard: Direction for next move (starboard)
Security_ML: Security status from SM-ML

Key features selected during the training of the Navigator-ML.

V. AI NAVIGATION MONITOR

The AI-NM utilizes the Navigator-ML and BSS information to make battleship decisions. Navigator-ML is trained to predict battleship heading and speed similar to BSS native navigation and is used by AI-NM to check BSS behavior continuously.

A. Features and Modeling

Navigator-ML is trained on BSS native navigation and SM-ML outputs to predict battleship heading and speed. The datasets from BSS are preprocessed by removing missing values and dropping columns with high correlation based on the heatmap we developed. Columns with categorical and boolean values are converted using one-hot encoding. Additionally, features are extracted by calculating Euclidean distance between current battleship coordinates and waypoints. Final features are listed in Table II.

The dataset is divided into 80% for training and 20% for testing. The model is trained using three algorithms: DT, RF, and KNN Regression. All three models' hyperparameters are tuned using Grid-Search CV and a cross-validation set of 5.

B. Training

Mean Absolute Error (MAE) is the total of absolute differences between expected and actual values divided by observations. MAE outputs y-units. The regression loss function uses Mean Square Error (MSE), the most common regression metric, to calculate the squared error between predicted and actual values. R2 is the only context-independent model comparison metric. Regression models: DT, RF, and K-Nearest Neighbors predicted Battleship navigation. Performance measurements were MAE, MSE, and R-squared. The DT model was more accurate and precise, with an MAE of 0.1688 and an MSE of 0.3578. With 0.9999 R-squared, it accounts for variance well. The Random Forest model lost effectiveness as errors rose. MAE and R-squared for KNN matched the DT, but MSE was greater. The DT model performed better in navigation challenge accuracy and dependability, indicating its applicability. The DT was best for the scenario because of its efficiency without normalizing or scaling.

VI. AI SECURITY MONITOR

The SM-ML Module uses real-time data from components to monitor their behaviors and functionality. The SM-ML was inspired by using explainable AI to filter anomalous system component behavior and take action against it by classifying those behaviors by quantifying their system impact and defining their reasoning using these metrics [15]. This approach aims to be preventive, explainable [13], robust in spotting new threats and anomalies, and less dependent on humans. The data is integrated from the ICS Module and the Weapon System. The information these systems provide is crucial for the SM-ML due to its criticality to the battleship's operational stability.

A. Features and Modeling

The suggested cybersecurity architecture employs the 'DecisionTreeClassifier' model for training, as it can distinguish between three system functionality states. All systems operate optimally in State "1", or "Go". State "2" or "Down" implies the weapon system is non-operational, but the ICS module works. State "3" indicates a compromise that may affect the ICS module, weapon system, or both. These states act as a "Label" column classified as operational (1), compromised (3), or non-operational (2). Data was generated from compromised and non-compromised ICS modules and weapon systems for training purposes and labeled accordingly. The final features are described in Table III.

Transparency and speed are essential for understanding the model's decision-making process in a trustworthy domain in the context of real-time security, which the DT graph provides effectively by handling non-linear relationships and complex feature interactions while being quick to train.

B. Training

Three machine-learning classification models were used to assess the SM-ML system: DT, Random Forest, and KNN.

TABLE III
FEATURES FOR SECURITY NAVIGATOR ML

Feature
Power: Power(Watts) measured from ICS Module.
Weapon_Range_Status:. Weapon's range status.
Weapon_status: Weapon execution posture status.
Msg: Encoded message from Weapon System.

Key features selected during the training of the SM-ML module.

Each model was thoroughly examined to discover the best fit for our purpose.

Three machine learning models—Decision Tree, Random Forest, and K-Nearest Neighbors—are compared in this article. The efficacy of the models was assessed using various metrics, including precision, recall, F1 score, and accuracy. Regarding overall performance, the DT model demonstrated superior precision, recall, and F1 scores of 1.0 and accuracy of 0.998. This suggests that the threat detection and classification capabilities are exact. The Random Forest model performed marginally inferior. The performance of the KNN model was considerably inferior across all metrics. The findings indicate that the DT model exhibited superior performance compared to the other two models, showcasing the highest levels of accuracy and dependability in anomaly detection. It validates quantitatively the DT's efficacy as a security monitoring model.

VII. EXPERIMENTAL EVALUATION

A. Experimental Setup

The simulator and the ML models were developed on a Windows-based Operating System with an 11th Gen Intel Core i5 processor and 16GB RAM and used *Python 3.9+*. It uses microservices and third-party libraries for scenario generation, environment modeling, data transmission, and user interaction. *Arcade, Shapely, PyYAML, Numpy, ZeroMQ, OPC UA, MQTT, Scikit-Learn*, and *Flask* were used. The modular design allows simulation, analysis, and data flow orchestration with *Docker*-based technologies.

B. Experimental Procedure

We ran three sets of experiments in our study. First, as a control, we tested the battleship's ability to avoid obstacles without supervision from other modules. Next, we tested the ability of the AI-NM to take over and either avoid collisions on behalf of the native navigator or stop the ship before a collision. Finally, we tested the security monitor's ability to identify anomalies in the ICS network.

The following logic is implemented for the AI-NM in a series of steps: (1) the AI-NM continuously compares the BSS heading to Navigator-ML predictions. It triggers alerts and halts the ship if the BSS starts heading in a strayed-off direction, (2) the AI-NM initiates the Collision Avoidance Supervisor when Collision Avoidance warnings are valid, and Navigator-ML predicts values consistent with the BSS, and (3) the AI-NM brings the battleship to speed zero when SM-ML detects an anomaly in battleships location/speed, signaling a potential security compromise.

TABLE IV
BATTLESHIP SCENARIO ANALYSIS

#	Native Navigator	Navigation AI	Navigation AI Attacked
1	Collision	Avoided	Halted
2	Collision	Avoided	Halted
3	Collision	Collision	Halted
4	Collision	Avoided	Halted
5	Collision	Avoided	Halted
6	Collision	Avoided	Halted
7	Collision	Avoided	Halted
8	Collision	Avoided	Halted
9	Collision	Collision	Halted
10	Success	Success	Halted

Table IV illustrates the BSS navigation with the native algorithm, NM-AI, and under cyber threats through ten scenarios.

1) Native Navigator and AI Navigator Evaluation: To test the effectiveness of the BSS native navigation and the AI-NM's ability to oversee it, 10 scenarios were pseudo-randomly generated. These scenarios were then run with the native navigator, and the AI-NM was present to provide a suitable comparison. The results of these ten scenarios were tabulated and displayed in the first column in Table IV for the native navigator alone and in the second column with the AI-NM included.

2) AI Navigator and Security Monitor Communications Evaluation: To test the AI Security Monitor's ability to detect spoofed locations/speeds and communicate this to the AI Navigator, 10 spoofed locations/speed scenarios were pseudo-randomly generated. The results of these ten scenarios were tabulated and displayed in the third column in Table IV.

3) Security Monitor ML Evaluation: We ran the trained 'DecisionTreeClassifier' model against data from a compromised weapon system and ICS network to determine its ability to identify anomalous behavior. The accuracy of SM-ML's prediction is described in the Table V.

VIII. RESULTS AND DISCUSSION

This work demonstrates how our solution can model realistic battleship behavior, such as ICS and weapon systems, communications, and autonomous navigation. The BSS also includes self-monitoring using two ML-based systems that actively modify behavior when necessary. The section below describes the results generated by each element individually.

A. Results of Native Navigation in BSS

As shown in Table IV, the native navigation on the battleship does not show a significant success rate for each scenario. This native navigation allows some object avoidance, which is expected considering the simple steering heuristic.

B. AI Navigator Monitor Module

In column 1 of Table IV, scenario 10 shows the Native Navigator and AI-NM both succeeding or navigating all

TABLE V
EXPERIMENT RESULT OF COMPROMISED BATTLESHIP SYSTEM

S. No.	Case of Compromise	Success	Failure	Total
1	No More Rounds	500	0	500
2	Out of Range	392	0	392
3	Missile Misses	134	0	134
4	Loading	214	38	252
5	Reloading	215	0	215
6	Power	1000	0	1000

Outcomes of various compromise scenarios in a weapon system such as missile misses, sudden out-of-range status, empty rounds status, or unusual ICS system power due to compromise.

waypoints, implying a less complex navigation scenario. In column 2 of Table IV, the AI-NM acts as an oversight system to catch native navigation failures and thus prevents some collisions—the results of simulated attacks on a battleship. 'Collision' indicates a scenario where the ship collided with an obstacle, 'Success' denotes successful navigation of all waypoints, 'Avoided' indicates the AI-NM prevented a collision, and 'Halted' indicates SM-ML detected a compromise and instructed AI-NM to stop the ship. The results show that the AI Navigator avoided collisions successfully in 7 of 9 cases (77.7%). This demonstrates the improved navigation capabilities of the AI system compared to the native navigator. Finally, The AI-NM stopped the battleship when the SM-ML predicted a compromise.

C. Security Monitor ML Module

The SM-ML successfully predicted over 2000 potential compromises, such as anomalous power values, false sensor readings, or inconsistencies in the weapon system firing logic. Illustrated in Table V is the Security Monitor's capacity to anticipate Weapon System breach situations. The only exceptions were the "Loading" cases, which saw 38 failed predictions out of 252 cases, representing an approximately 15% failed prediction rate. The overall accuracy of Security AI prediction stood at approximately 98.5%.

IX. LIMITATIONS

BSS modules are still a work in progress with the potential to simulate threats against semi-autonomous or fully autonomous naval vehicles. The modules' capabilities are expandable and modular but currently limited. The BSS can simulate additional ships according to [12]; however, this research is limited to a specific class and type. The native collision avoidance algorithm is easy to replace but a simple heuristic prone to failure. This collision avoidance is also a limiting factor for the AI-NM since the ML is trained on a simple collision avoidance algorithm. The SM-ML is limited by the relatively small amount of systems stored in the ICS network simulation.

X. SUMMARY AND FUTURE WORK

This investigative study demonstrates the feasibility of developing a modular battleship simulator and an AI navigation and security monitoring system. Even though our work had limitations, we have shown that such a feat is feasible. This work contributes to the literature on maritime vehicle simulators and maritime systems cybersecurity. Future work could expand the simulation modules and threat model to focus on real-world problems in autonomous maritime platforms and control systems. The collision avoidance algorithm can be improved in subsequent developments. The emulation can include additional ICS modules representative of additional key systems in a platform.

REFERENCES

[1] J. Coker. (2023), "1000 Shipping Vessels Impacted by Ransomware Attack," in Infosecurity Magazine, 18 January 2023. Available: www.infosecurity-magazine.com/news/shipping-vessels-ransomware-attack

[2] M. Howard and S. Lipner, The Security Development Lifecycle. Redmond, WA, USA: Microsoft Press, 2006.

[3] J. M. Lanouette, "Naval Cyber Warfare – Are Cyber Operators Needed on Warships to Defend against Platform Cyber Attacks," Master's thesis, Canadian Forces College, Toronto, Ont., Canada, 2016, JCSP/PCEMI 42-17.

[4] J. M. Szatkowski, Y. Li, and L. Du, "Enabling Reconfigurable Naval SCADA Network through Software-Defined Networking," 2022 IEEE Transportation Electrification Conference and Electric Aircraft Technologies Symposium (ITEC+EATS), 2022, pp. 214-218.

[5] Kavallieratos, G., Katsikas, S., Gkioulos, V. (2019). Cyber-Attacks Against the Autonomous Ship. In: Katsikas, S., et al. Computer Security. SECPRE CyberICPS 2018. Lecture Notes in Computer Science(), vol 11387. Springer, Cham.

[6] Y. Suo, S. Yang, G. Chen, and W. Liu, "A Target Ship Server Driven by AIS Signal for the Navigation Simulator," in "2017 2nd International Conference on Computational Modeling", Simulation and Applied Mathematics (CMSAM 2017), Xiamen, Fujian, China, 2017, pp. 76-82, ISBN: 978-1-60595-499-8.

[7] Microsoft Defender Research Team, "CyberBattleSim," (2021). Available: GitHub repository, https://github.com/microsoft/cyberbattlesim.

[8] M. Standen, et al, "Cyber Operations Research Gym," (2022). Available: GitHub repository, https://github.com/cage-challenge/CybORG/tree/b5a71c4cf156bf450a0bb3ffdb63fc0340a10bd4.

[9] K. Hasegawa, J. Fukuto, R. Miyake, and M. Yamazaki,"An Intelligent Ship Handling Simulator with Automatic Collision Avoidance Function of Target Ships," in Proceedings of the 17th International Navigation Simulator Lecturers' Conference (INSLC 17), Rostock-Warnemuende, Germany, 2012.

[10] Raphael Zaccone and Michele Martelli (2020) A collision avoidance algorithm for ship guidance applications, Journal of Marine Engineering and Technology, 19:sup1, 62-75.

[11] T. I. Fossen (2021). Handbook of Marine Craft Hydrodynamics and Motion Control. 2nd. Edition, Wiley.

[12] T.I. Fossen (2023) Python Vehicle Simulator Software. Available: github.com/cybergalactic/PythonVehicleSimulator/tree/master

[13] V. Manoj et al, (2023) "Explainable autonomic cybersecurity for industrial control systems," in 2023 IEEE 13th Annual Computing and Communication Workshop and Conference (CCWC).

[14] T. C. My, L. D. Khanh, and P. M. Thao,"An Artificial Neural Networks (ANN) Approach for 3 Degrees of Freedom Motion Controlling," in International Journal on Informatics Visualization, vol. 7, no. 2, pp. 301-309, Jun. 2023.

[15] Lanier Watkins, D. Hamilton, K. Kornegay, and A. Rubin. (2021). "Triaging Autonomous Drone Faults By Simultaneously Assuring Autonomy and Security," in 2021 55th Annual Conference on Information Sciences and Systems (CISS), 2021, pp. 1-6.

[16] IACS (2022). Rec 166 – Recommendation on Cyber Resilience – New Corr.2 Apr 2022 Clean. Available: https://iacs.org.uk/resolutions/recommendations/161-180/rec-166-new-corr2-cln

[17] D. S. Cusanelli and G. Karafiath, (2012) "Hydrodynamic Energy Saving Enhancements for DDG 51 Class Ships," in ASNE Day 2012, 2012, pp. 1-16.