INL/RPT-23-75112
Revision 000

Autonomous System Inference, Trojan, and Adversarial Reprogramming Attack and Defense

Cyber Threat Assessment of Machine Learning Driven Autonomous Control Systems of Nuclear Power Plants

September 2023

M3CT-23IN1105013

Idaho National Laboratory
Chris Spirito

Georgia Institute of Technology
Patience Lamb

INL is a U.S. Department of Energy National Laboratory operated by Batelle Energy Alliance, LLC

DISCLAIMER

This information was prepared as an account of work sponsored by an agency of the U.S. Government. Neither the U.S. Government nor any agency thereof, nor any of their employees, makes any warranty, expressed or implied, or assumes any legal liability or responsibility for the accuracy, completeness, or usefulness, of any information, apparatus, product, or process disclosed, or represents that its use would not infringe privately owned rights. References herein to any specific commercial product, process, or service by trade name, trademark, manufacturer, or otherwise, does not necessarily constitute or imply its endorsement, recommendation, or favoring by the U.S. Government or any agency thereof. The views and opinions of authors expressed herein do not necessarily state or reflect those of the U.S. Government or any agency thereof.

INL/
Revision 000

Autonomous System Inference, Trojan, and Adversarial Reprogramming Attack and Defense

M3CT-23IN1105013

Idaho National Laboratory
Chris Spirito

Georgia Institute of Technology
Patience Lamb

September 2023

Idaho National Laboratory
Idaho Falls, Idaho 83415

http://www.inl.gov

Prepared for the
U.S. Department of Energy
Office of Nuclear Engineering
Under DOE Idaho Operations Office
Contract DE-AC07-05ID14517

Page intentionally left blank

Page intentionally left blank.

CONTENTS

FIGURES

Page intentionally left blank.

ACRONYMS

AI	Artificial intelligence
API	Application Programming Interface
ASI	Advanced Sensor and Instrumentation
BWR	Boiling water reactor
CLI	Command line interface
DCS	Distributed Control System
DT	Digital Twin
DTDL	Digital Twin Definition Language
HMI	Human-Machine Interface
IAEA	International Atomic Energy Agency
IIoT	Industrial Internet of Things
I&C	Instrumentation and Control
JADE	Java application and developer environment
JSON	Javascript object notation
MCO	Moisture Carryover
ML	Machine learning
OT	Operational Technology
PWR	Pressurized water reactor
RBAC	Role-based access control
RDF	Resource Description Framework
REST	Representational state transfer
SDK	Software Development Kit
SLA	Service Level Agreement
SMR	Small Modular Reactor
XML	Extensible markup language

Page intentionally left blank.

[Spirito] One of our researchers this year completed her thesis on the topic of Cyber Threat Assessments of Machine Learning Driven Autonomous Control Systems of Nuclear Power Plants. Her focus was on: Autonomous control system design and implementation; Cyber threat assessment of machine learning-based digital twins; and Cyber-risk comparison of traditional machine learning and automated machine learning frameworks. As this work was funded under our research project with the scope of understanding Autonomous System Inference, Trojan, and Adversarial Reprogramming Attack and Defense, this M3 Deliverable takes the technical analysis completed and uses it explain these types of attacks and the defensive mitigation measures that can be used to lower the likelihood that this type of attack would be successful. We encourage you to read through the published paper and optionally request a copy of her thesis for a deeper dive into these concepts.

1. Introduction

In the world of ever-advancing technology, Autonomous Systems (AS) find extensive application, bolstering functionalities of critical infrastructures such as nuclear power plants. These systems, however, are increasingly becoming a target for nefarious activities, namely through inference attacks, trojan attacks, and adversarial reprogramming. This paper delves into a comprehensive exploration of machine learning (ML)-driven autonomous control systems within advanced nuclear reactor designs, revealing the vulnerabilities and proposing strategies for defense against potential cyber-attacks.

Advanced cyber-attacks against critical infrastructure and the energy sector are becoming more common. With the invention of autonomous control systems (ACS) within advanced nuclear reactor designs, system designers, reactor operators, and regulators must consider cybersecurity during the design and operational phases. This article provides a cyber threat assessment of machine learning (ML)-based digital twinning (DT) technologies in the context of advanced reactor ACS. A cyber-physical testbed was created to emulate nuclear reactor digital instrumentation and controls (I&C) and act as a basis for the ACS. The ACS was designed as two plant-level DTs predicting reactor malfunctions and determining control actions and two component-level DTs responsible for classifying component states and forecasting component inputs and outputs (I/O). Two duplicate ACS designs– one using a traditional ML framework and one using an automated ML (AutoML) framework– were created and tested against cyber-attacks on training data, real-time process data, and ML model architectures to determine their respective qualitative cyber-risk in terms of likelihood and impact. Both frameworks showed similar cyber-resilience against training, real-time, and ML architecture attacks, proving that neither is inherently more secure. Recommended safeguard and security measures are posed to system designers, reactor operators, and regulators to maintain the cybersecurity of ML-based DT technologies such as ACS, prompting a holistic view of shared responsibility for maintaining cyber-secure ML-based systems.

As global reliance on generation III reactors begins to be critically assessed, the evolution towards advanced reactor systems utilizing digital instrumentation and controls (I&C) becomes not merely preferable, but essential. The integration of semi and fully autonomous control systems (ACS), powered by digital I&C and machine learning (ML)-based digital twinning (DT) technologies, emerges as a potent strategy to mitigate operations and maintenance costs, thereby enhancing the economic feasibility of novel reactor designs [1]. However, with a staggering 500% and 380% increase in cyber-attacks reported against the energy sector by the United States Department of Energy (DoE) [2] and the European Union [3] respectively, a surge in cyber vulnerabilities specifically targeting the nuclear industry has been markedly observed. Notable incidents, such as the W32.Ramnit spyware infiltration at the Gundremmingen nuclear power plant in Germany [5] and the Dtrack spyware intrusion at the Kudankulam nuclear power plant in India [6], while not directly compromising core industrial control systems (ICS), underscore a compelling necessity to fortify cybersecurity protocols in safeguarding reactor systems against increasingly adept digital adversaries.

In light of this, our investigation extends beyond conventional cybersecurity parameters, diving into the intricate web of potential vulnerabilities woven into ML-based DTs and ACS in advanced reactor systems. A crafted cyber-physical testbed and preliminary ACS were devised to act as a mirror, reflecting potential configurations of advanced reactor control designs [7]. Moreover, this study is intertwined with a scrutinization of ML models, developed either through conventional, manually tuned methodologies or via automated means through AutoML, probing into their cyber-risk profiles within operational technology (OT) environments.

Expanding on this, two distinct ACS blueprints were forged – one navigating through the corridors of traditional ML and the other traversing the path of AutoML – in an effort to holistically encapsulate the considerations pivotal to ML-based DT control system design. Employing the SANS Institute Industrial Control System (ICS) Kill Chain [8] and the MITRE ATT&CK Tactics, Techniques, and Procedures (TTP) framework [9], a structured analysis was conducted, launching three targeted attacks against the training dataset, real-time dataset, and ML models, therein dissecting the potential cyber-attack implications against both ML frameworks within an ACS milieu. It is essential to note that three distinct categories of attacks were conducted against both ACS configurations, each encompassing three distinct ML-based DTs, cumulating in a total of 18 varied attacks. This exploration extends into the realms of Autonomous System Inference, Trojan, and Adversarial Reprogramming Attack and Defense, unraveling vulnerabilities, and opportunities for fortified defenses against such intrusions, particularly where ML-driven technologies, and by extension, ACS, are deployed. Final recommendations, articulated through a lens of security, safeguard, and implementation considerations, are presented for both traditional and AutoML models, anchoring upon the existing knowledge landscape and ML-based DT modeling for ACS, and are offered as a beacon to guide the nuclear industry through the intricate cybersecurity challenges that lie ahead.

2. Incorporating Autonomous Systems in Nuclear Power Plant Control

The interplay between advanced reactor systems and Digital Twin (DT) technologies pivots on an axis of detailed and responsive control, governed by real-time data from digital instrumentation and controls (I&C). Here, digital I&C, encompassing programmable logic controllers (PLCs) and field programmable gate arrays (FPGAs), plays a crucial role by harvesting component-level data and establishing seamless communication with Autonomous Control Systems (ACS) [10]. In a digital landscape perpetually metamorphosing with emerging cyber threats, rigorous testing and validation of these digital systems become imperative to safeguard compliance and thwart potential cyber-attacks. A hardware-in-the-loop (HiL) cyber-physical testbed, developed at the Georgia Institute of Technology, stands as a testament to the industry's dedication to penetrate standard hardware testing and ensure the cyber-resilience of digital I&C and ACS implementations [10].

The cyber-physical testbed is demarcated into five intrinsic systems:

1. The reactor simulator computer
2. The PLC
3. The ACS
4. The training database
5. An adversarial system.

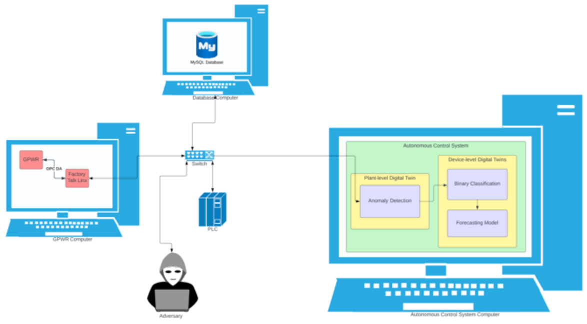


Figure 1: FANCY Cyber-physical testbed

Considering the diversity embedded in advanced reactor designs and the noticeable absence of high-fidelity I&C advanced reactor simulations, the Western Services Corporation (WSC) Generic Pressurized Water Reactor (GPWR) [11] was elected as the simulation platform for a Generation III Pressurized Water Reactor (PWR), complete with digital I&C and real-time communications facilities.

Autonomous System Inference: A Cybersecurity Linchpin

Autonomous System (AS) inference solidifies itself as an indispensable bulwark against cyber threats, adept at discerning potential malevolent inputs and perturbations, which may insidiously penetrate and compromise the reactor's operational safety and functionality. It underscores its necessity by distinguishing between legitimate system operations and a series of ostensibly innocuous operations that could cumulatively signal a trojan or an adversarial attack.

In the complex tapestry of maintaining system integrity against a backdrop of ever-evolving cyber threats, AS inference perpetually adapts and learns from I&C data, constructing predictive models to delineate and safeguard the standard operational parameters of the ACS. This acts as a safeguard against subversive manipulations aimed at reprogramming or corrupting ML models. Therefore, AS inference not only modulates system functionality but also serves as a sentinel, safeguarding communication, and operation fidelity within nuclear power plant controls.

As we navigate through this section, we will explore detailed strategies for deploying AS inference, illuminate its essential function in shielding systems from trojan and adversarial reprogramming attacks, and elucidate its practical application within the architectural configuration of the cyber-physical testbed, affirming the symbiotic relationship between ML-based DT control system design and cybersecurity [15, 18, 19].

3. Constructing Cyber-Physical Testbed for Analyzing Autonomous Systems

The adoption of Autonomous Control Systems (ACS) in advanced nuclear reactor designs indeed propels us into an era where the seam between technology and operational management is blurred, creating a conduit where machine intelligence becomes a cornerstone in maintaining and operating complex nuclear systems. A nexus between real-time data acquisition and machine-driven decision-making, ACS operates on a fulcrum of employing data to weave into decision fabrics, which are inherently stitched into the operational paradigms of Digital Twins (DTs). Moreover, the implementation of ACS dramatically reduces the need for constant human surveillance and intervention, paving the way for a more remote, yet robust, management of nuclear reactor operations [1].

The engineering of an ACS, especially one that leverages Machine Learning (ML)-based DTs, has illuminated the pathway to seamlessly simulating reactor subsystems within a thoroughly data-oriented framework, presenting a stark and advantageous contrast to traditional physics-based DTs. This approach offers a granular level of design flexibility, coupled with a powerful predictive and modeling capability, both of which become paramount in crafting a resilient and reliable nuclear reactor operational model. Within this framework, plant-level DTs are pivotal, especially two primary ones which bear the responsibility of detecting malfunctioning states of the reactor and subsequently pivoting towards control strategies that are congruent with those states. Beyond this, a dynamic array of component-level DTs is operationalized to both categorize the component state—delineating between transient and steady-state operations—and predict component Inputs/Outputs (I/O) to ensure steadfast adherence to predetermined operational thresholds.

Digging a layer deeper, the ACS, intertwined with ML-based DTs, opens up a plethora of opportunities, such as enhancing the predictability of subsystem performance, scaling the response mechanisms, and delivering robust autonomous control in scenarios where human intervention might be constrained or delayed. Furthermore, the integration of ACS facilitates a continuous and omnipresent virtual eye, which meticulously monitors, evaluates, and reacts to the myriad of parameters and variables within the reactor's operational sphere, thereby assuring a heightened level of safety and efficiency. It's also worth noting that while ACS champions autonomy, the system does not entirely obviate the need for human oversight but transforms it, ensuring that human operators are shielded from the mundanities of routine monitoring, and instead, can focus on more strategic and critical operational aspects. By doing so, ACS ensures that human expertise is leveraged where it is most impactful, enhancing not only the operational efficiency but also embedding a layer of strategic human oversight that is focused, precise, and exceptionally valuable. The interplay between ACS and DTs thus becomes a lynchpin in ensuring that nuclear reactor operations are not only optimized but also enveloped within a cocoon of safety, reliability, and strategic management, especially in scenarios where the complexity and risk are inherently escalated. A schematic representation, providing a visual insight into the intricate decision-making process crafted by the ACS, is depicted in Figure 2, illustrating a network that intertwines data, machine intelligence, and autonomous decision-making into a cohesive operational narrative.

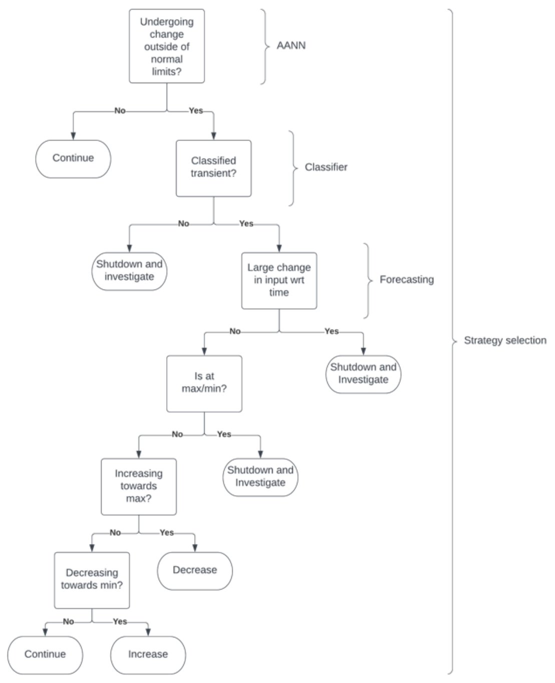


Figure 2: Strategy selection decision tree plant-level Digital Twin. Full size image is in Annex I.

Emphasizing Inference Capabilities in Cyber-Physical Testbed Design

Diligent scrutiny of the inference capabilities of ACS necessitates a holistic cyber-physical testbed that not only affords insights into system behaviors but also substantiates the system's capacity to deduce accurate operational inferences from a myriad of inputs. This becomes particularly paramount when considering potential adversarial attacks that might attempt to clandestinely manipulate these inferences. Auto associative Neural Network (AANN), acting as the plant-level DT, pioneers the detection of aberrant behavior. Concurrently, classifiers at the component-level DT discern whether a specific component is navigating through a transient. The forecaster, by predicting the component I/O, and a strategy selector, which amalgamates all DT information, deduces control actions to ensure sustained, safe operations. It is pertinent to note that the strategy selection model has been omitted in current implementations since control actions can be extrapolated directly from individual DT outputs.

In the context of ML modeling for DTs, traditional ML algorithms have been constructed utilizing TensorFlow Keras [22] and Scikit-learn [23] models and underwent meticulous manual tuning. In contrast, AutoML DTs were erected using PyCaret [24] and Tensorflow Keras' Keras-Tuner [22], leaning into hyperparameter tuning methodologies to automate the tuning process.

Simulating Adversarial Attacks in a Testbed Environment

Within the sanctuary of the cyber-physical testbed, the ability to simulate adversarial attacks holds paramount significance, providing a controlled environment in which to assess not only the vulnerabilities of the ACS but also its aptitude to detect and remediate such assaults, thereby fortifying the system against genuine threats. The simulation of these potential attacks enables a comprehensive analysis of the system's robustness and aids in identifying and mitigating potential exploits and vulnerabilities, creating a rigorously tested and resilient ACS capable of safeguarding nuclear reactor operations against a spectrum of cyber threats.

3.1 Malfunction Detection Plant-Level Digital Twin

In the quest to ensure safety and efficiency in advanced nuclear reactor systems, malfunction detection at the plant level emerges as a pivotal aspect, employing Decision Trees (DT) to discern and respond to discrepancies within 70 plant-health variables derived from the Generic Pressurized Water Reactor (GPWR). Leveraging both AutoML and traditional Machine Learning (ML) models, a semi-supervised training methodology was applied, utilizing non-anomalous data, an amalgam of transient and steady-state data, and subsequently testing on malfunction data. The outcome value matrix plays a crucial role in determining malfunction thresholds for each variable, graphically reconstructing the reactor power and enabling real-time assessment through data procured from the Programmable Logic Controller (PLC) to ascertain component states.

The meticulously crafted training dataset embodies potential load-following and steady-state operation capabilities, comprising both steady-state and transient power operations with the power stabilized at 100% for 30 minutes, scaled down to 70% for an ensuing 15 minutes, and escalated back to 100% for an additional 15 minutes, executed seven times in a beginning of life (BOL) operation scenario. The attentive monitoring of all 70 plant-health related variables forms a key part of the training dataset. A semi-supervised training approach introduced contamination into the dataset through a manipulated malfunction of the feedwater control valve for steam generator 1 (posFCFV0075) – deviating from a nominal 50% open to 30% open – at 11 minutes and 40 seconds into the operation, throttled for 32 seconds, and subsequently spliced into the uncontaminated dataset. Ensuring the contamination remained at 1.712%, this aspect was crucial in evaluating model accuracy, visualized comprehensively in Figure 3. The dataset, amalgamating contaminated and uncontaminated data, was meticulously bifurcated into training (seen) data and testing/validation (unseen) data, allocating 95% to the former and reserving 5% for the latter during the validation phase. Both AutoML and traditional ML model architectures were scrupulously evaluated for the malfunction detection model, ultimately selecting an Auto Associative Neural Network (AANN) – or autoencoder – for the traditional ML model. A spectrum of ensemble methods was concurrently assessed during training to pinpoint the optimum AutoML model.

Post-training, real-time data was meticulously ingested and decrypted from the PLC through Control and Information Protocol (CIP) packets utilizing Pylogix. Data from the 70 variables, polled at 50ms intervals, were supplied to the malfunction detection models in 100s windows for reconstruction. The validity of the real-time data processing was established through parsing and supplying the training dataset in 100s periods as real-time data for reconstruction, in addition to conducting steady-state, transient, and multiple malfunction GPWR operations. Engaging both the traditional ML and AutoML malfunction detection models successfully differentiated the malfunctions from conventional steady-state and transient operations, highlighting the pivotal role of cyber-physical testbeds in evaluating and safeguarding the inference capabilities of Autonomous Systems (AS) and in emulating potential adversarial attacks, thereby ensuring the reliability and robustness of ACS in an increasingly digitalized and cyber-threat prevalent environment.

3.1.1 Auto Associative Neural Network (AANN) Malfunction Detection Model

Harnessing the predictive prowess of the Auto Associative Neural Network (AANN) for malfunction detection stands pivotal, especially when embedded in complex nuclear reactor control environments. In the evaluative process, three renowned traditional Machine Learning (ML) multivariate anomaly detection models were pitted against each other: isolation forests, One-Class Support Vector Machines (OC-SVM), and AANNs. The latter emerged superior, notably excelling in precision and accuracy metrics.

The architectural essence of the AANN used was laid out with 70 neurons apiece at the top and bottom layers, alongside a bottleneck hidden layer comprising 8 neurons. Each of the 70 variables was meticulously reconstructed in tandem with time. The Mean Squared Error (MSE), as depicted in Equation 1, served as the veritable metric to ascertain model accuracy during the various phases of training, validation, and real-time data testing, wherein "n" symbolizes the number of variables embedded in the reconstructed and original input dataset, "Ri" is the reconstructed value of the ith input, and "Oi" delineates the actual value of the dataset. Remarkably, the training and validation MSE anchored at 2.208 x 10^(-5) and 2.108 x 10^(-5) respectively, post 30 epochs of training, offering an average MSE sprawled across all 70 variables.

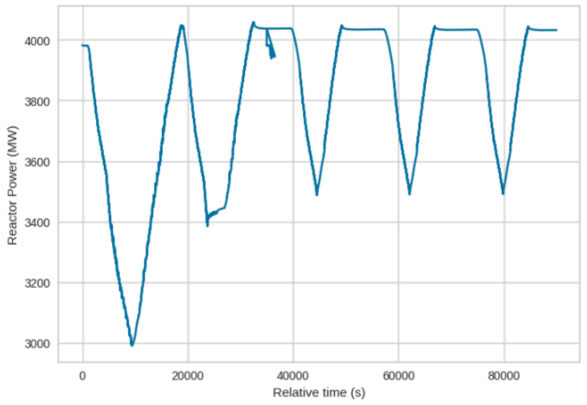


Figure 3: Malfunction detection training dataset

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(R_i - O_i)^2 \tag{1}$$

Figure 5 discernibly elucidates the minuscule error between the original and reconstructed testing power data, underscoring the AANN's capacity to reconstruct the original testing dataset with negligible error.

In carving out a meticulous threshold, the standard deviation (σ) was calculated employing the MSE of each point within the dataset for each jth data point (denoted MSEj) per Equation 2, delivering an average of MSE across all 70 variables.

$$\sigma = \sqrt{\frac{\sum_{j=1}^{m}(MSE_j - MSE)^2}{m-1}} \tag{2}$$

Incorporating the MSE and σ, a threshold "T" was formulated, pivotal in determining malfunction data points sprawled across all 70 variables as demonstrated by Equation 3.

$$T = MSE + 2\sigma \tag{3}$$

The malfunction data percentage (malfunction score "a") can be adeptly computed, embracing the number of data points with an MSE above the threshold, divided by the input dataset's size, as depicted in Equation 4. This malfunction score becomes instrumental in delineating the model accuracy as a percentage for both AutoML and traditional ML models.

$$a = \frac{\#\text{malfunction data points}}{\#\text{total data points}} \tag{4}$$

In the context of the traditional ML model, 1.744% of the training dataset was earmarked as malfunctioning, juxtaposed against the actual value of 1.712%, with a threshold oscillating at 0.0005611 average tag MSE between non-anomalous data and malfunction data.

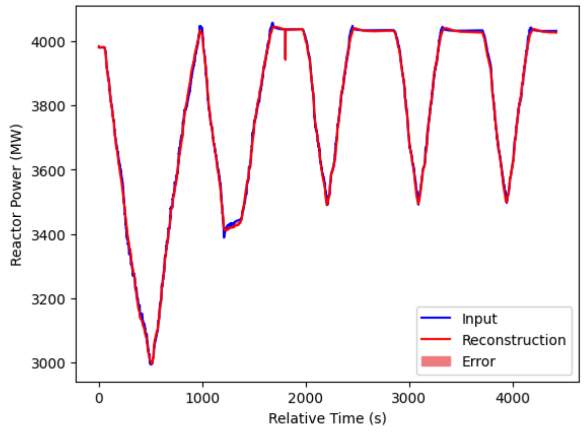


Figure 5: Traditional ML AANN Training Data Power Reconstruction

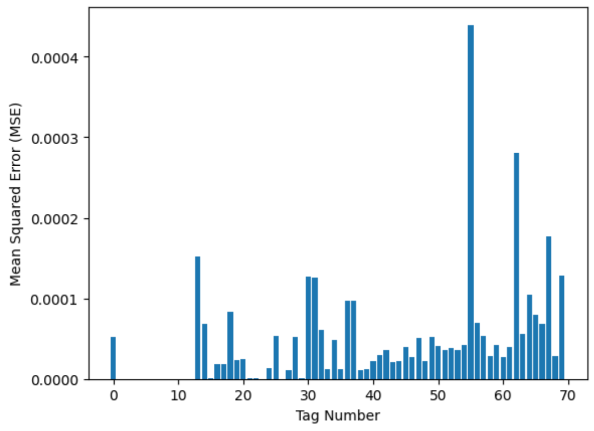


Figure 4: Traditional ML AANN MSE per tag during training

Figure 4 articulates the relatively diminutive reconstruction error of a combined steady-state and transient operation across all 70 tags, with the heftiest MSE still residing below the malfunction detection threshold.

This model is subsequently wielded to ascertain the presence of malfunctions within real-time data via the 100s window, with 50ms polling from the PLC. The MSE of the real-time dataset within this 100s interval is computed to determine if the dataset is showcasing a malfunction within this window.

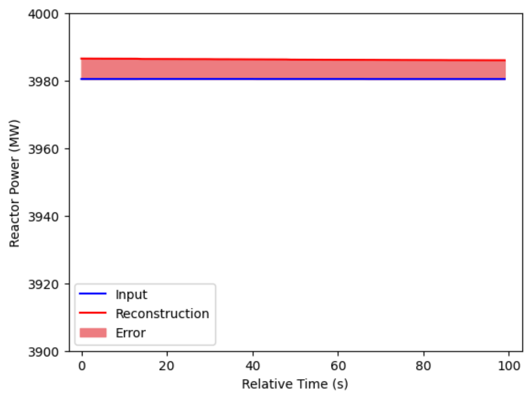


Figure 6: Traditional ML AANN Power Reconstruction on steady-state real-time data.

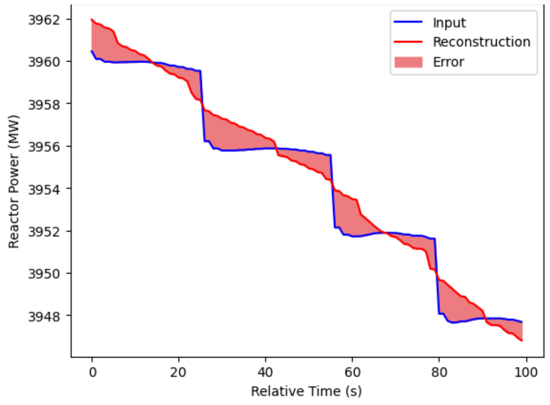


Figure 7: Traditional ML AANN power reconstruction on transient real-time data

Figure 6 and Figure 7 vividly illustrate the traditional ML AANN power reconstruction on steady-state real-time data and its adeptness at reconstructing transient data, respectively. Here, a 100s window of transient power manipulation from 100% power to 75% power is insightfully captured, presenting an efficacious mechanism for identifying not only the operational status but also ensuring malfunctions are judiciously flagged in both steady-state and transient operations, fostering a secure and reliable nuclear reactor control environment.

4. Understanding ML-based Digital Twins in Autonomous Systems

As technology inexorably advances, Machine Learning (ML)-based Digital Twins (DTs) have crystallized as quintessential elements in orchestrating sophisticated Autonomous Systems (AS), especially in domains where operational complexity is inherently intertwined with risk and precision. The forthcoming section will meticulously unravel the myriad facets of integrating ML with DTs in AS, illuminating the synergy between data-driven modeling and autonomous control in managing and mitigating multifaceted operational challenges. In Subsection 4.1.1, we shall delve into the realm of Automated Machine Learning (AutoML) and its pivotal role in sculpting a Malfunction Detection Model that is both resilient and adept at navigating through the operational intricacies of autonomous systems. Subsequently, Section 4.2 will shed light on the Component-Level Transient Classifier DT, dissecting its functionality and discerning its instrumental role in reinforcing the autonomous decision-making machinery. Further, Subsections 4.2.1 and 4.2.2 will respectively explore the divergent yet complementary approaches of Traditional ML and AutoML in crafting steady-state Classification Models, offering insights into their capabilities, distinctions, and strategic applications within the larger echelon of Autonomous Control Systems. Embark on this journey to comprehend the nuances, applications, and underlying mechanisms of ML-based DTs, thereby appreciating their intrinsic value in elevating and safeguarding autonomous operations.

4.1.1 Automated Machine Learning (AutoML) Malfunction Detection Model

Embracing a judicious blend of technological finesse and data processing acumen, the AutoML model, leveraging a PyCaret backend, has been quintessential in constructing and preprocessing data tailored for individual models. This strategy orchestrated seven disparate model architectures to operate concurrently, settling on the model that exhibited the closest adherence to the actual malfunction training data value of 1.712%. This initiative essentially culminated in the formulation of an improvised AutoML modeling methodology, distinctly earmarked for malfunction detection applications. Notably, the efficacy of the AutoML model is gauged utilizing the malfunction score, as methodically delineated by Equation 4.

The threshold emerged through the application of the isolation forest anomaly scoring method, intrinsically embedded within PyCaret, a methodological approach pioneered by Nair et al. [25]. Concurrently, the Mean Squared Error (MSE) (as per Equation 1) of the reconstruction on real-time data served as a metric to determine the deviation from the threshold, ensuring consistent accuracy and data fidelity.

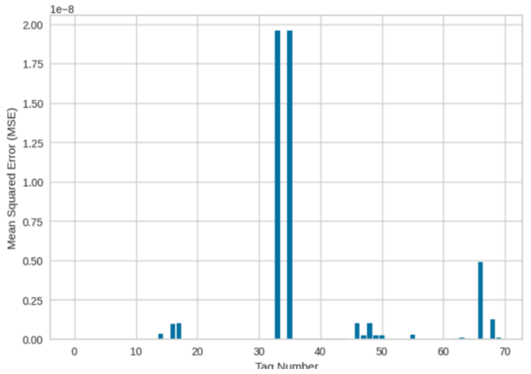


Figure 8: AutoML malfunction detector MSE per tag during

Against the backdrop of performance matrices, the AutoML model demonstrated a marginally superior performance compared to its traditional ML counterpart. Specifically, it identified 1.718% of the original training dataset as a malfunction. A meticulous exploration of minor reconstruction error across all 70 tags is presented in Figure 8, revealing errors predominantly orbiting around 1.000E * 08 MSE. The threshold was astutely calculated to be 1.743E*09, while the training and validation MSE were recorded at 7.248E *09 and 7.305E * 09, respectively.

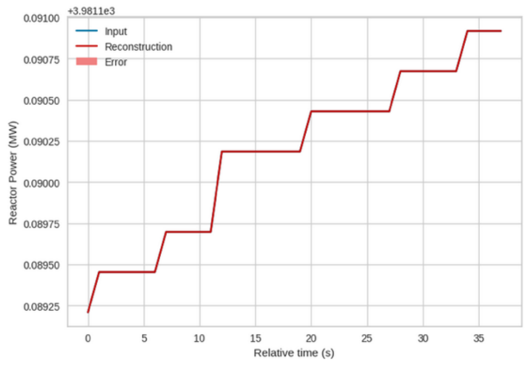


Figure 9: AutoML malfunction detector power reconstruction

As we cascade into real-time data reconstruction, the AutoML model showcased an MSE of 6.531E*10 on steady-state data during the identical 100s window in which the traditional ML model was scrutinized. Figure 9 delineates the real-time steady-state power reconstruction, illuminating that even under steadfast steady-state conditions, minute power fluctuations were reconstructed with high fidelity. This precision resulted in the error between the reconstructed and input dataset becoming virtually imperceptible.

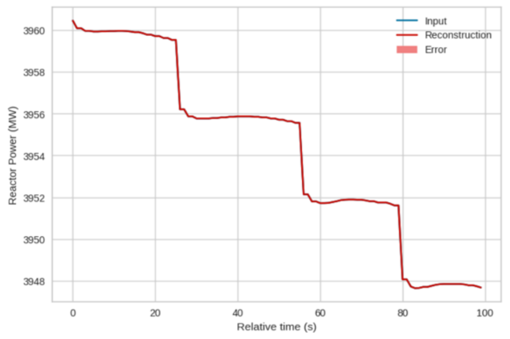


Figure 10: AutoML malfunction detector power reconstruction on transient real-time data.

Figure 10 visualizes the reconstructed real-time data engendered by the AutoML model on transient data, during the analogous 100s window wherein the traditional ML model underwent evaluation, producing an MSE of 7.815E * 10.

In a harmonious blend of accuracy and technology, both traditional ML and AutoML Decision Trees (DTs) were adept at reconstructing all 70 plant-level variables, a strategy that proved pivotal in determining system malfunction occurrences and pinpointing the malfunctioning component. The AutoML framework, enriched with its simplistic SKLearn backbone inherent in PyCaret, provided a platform for more adaptable modeling and economized on computation time. Contrastingly, the traditional ML AANN afforded the luxury of accommodating larger datasets and enhancing explainability, albeit at the sacrifice of computation time and fidelity on diminutive datasets.

4.2 Component-Level Transient Classifier Digital Twin (DT)

Navigating through the rich tapestry of autonomous systems and their vulnerability to malfunctions and potential cyber-attacks, this section elucidates the role of Component-Level Transient Classifier Digital Twins (DTs) in the nuanced domain of malfunction detection and classification, particularly weaving through the complex narrative of "Autonomous System Inference, Trojan, and Adversarial Reprogramming Attack and Defense."

Upon identifying a malfunctioning component, a classifier discerns whether the component is navigating through a transient. For the preliminary implementation, solely the steam generator 1 (SG1) I/O were employed to cultivate component-level transient classifier DTs. Both traditional and AutoML models harnessed the unadulterated steady-state and transient datasets utilized by malfunction detectors, providing a cohesive framework for the analytical process. Operations at 100% power in a steady state are denoted as Steady=1, while transient operations assume the label Steady=0. The dataset underwent a pruning process to exclusively embody seven variables linked to the SG1 state: SG1 wide ring level (% open), SG1 narrow ring level (% open), SG1 bypass valve level (% open), SG1 base feedwater (kg/s), SG1 feedwater flow in (kg/s), and SG1 flow total (kg/s). A 1/5 testing to training split of SG1 state variables was employed, with the reactor's state (steady-state or transient) utilized as the target in a supervised training setting.

Binary classification accuracy served as the metric to gauge the fidelity of both traditional ML and AutoML models during the training phase, defined per Equation 5. Here, TP, TN, FP, and FN respectively represent true positive, true negative, false positive, and false negative predictions related to SG1 transient or steady-state operations.

$$\text{Accuracy} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{TN} + \text{FP} + \text{FN}} \qquad (5)$$

In real-time applications, data was extracted from the Programmable Logic Controller (PLC) utilizing Pylogix, mapping into the seven SG1 state-related variables in 50ms intervals and subsequently ingested into the classification models within 100s windows. Mirroring the approach for malfunction detection, real-time data modeling underwent validation by parsing and inputting the training dataset in 100s windows as real-time data, paralleled with real-time GPWR operations. Model accuracy concerning real-time data was authenticated by computing the derivative of reactor power relative to time within a 100s window, utilizing Equation 6 where P signifies reactor power in MW, t indicates operational time in seconds, and 0.4 has been experimentally determined through discerning the derivative of power at the point wherein a transient commences.

$$\text{True Label} = \begin{cases} 1 & \left|\frac{dP}{dt}\right| < 0.4 \\ 0 & \left|\frac{dP}{dt}\right| \geq 0.4 \end{cases} \tag{6}$$

Through the lens of Equation 5, the True Label is juxtaposed with the predicted label, crafting a landscape to assess the models' accuracy in real-time scenarios. Within the overarching theme of autonomous system inference and adversarial reprogramming, ensuring the accuracy and reliability of these models becomes imperative, especially considering the potential of adversarial attacks and the necessity to devise robust defense mechanisms to safeguard against trojan attacks and system exploitation. The integrity and precision of these DTs become pivotal, fortifying the system against erroneous data injection, and ensuring the veracity of the autonomous system's inferencing capabilities amidst potential cyber-physical threats.

4.2.1 Traditional Machine Learning (ML) steady-state Classification Model

In the overarching narrative of "Autonomous System Inference, Trojan, and Adversarial Reprogramming Attack and Defense," understanding the pivotal role of classification models in detecting, deciphering, and mitigating malicious actions becomes essential. Herein, the Linear Support Vector Classifier (LinearSVC) model was singled out as the traditional ML transient classification model, primarily attributed to its prowess in swiftly and accurately classifying data, even with a minimal feature set like the SG1 dataset. The model was operationalized with a maximum of 1,000,000 iterations and a regularization parameter C=0.002. An admirable training phase accuracy of 97.25% was achieved, depicted in Figure 11's confusion matrix where respective axes represent actual and predicted labels.

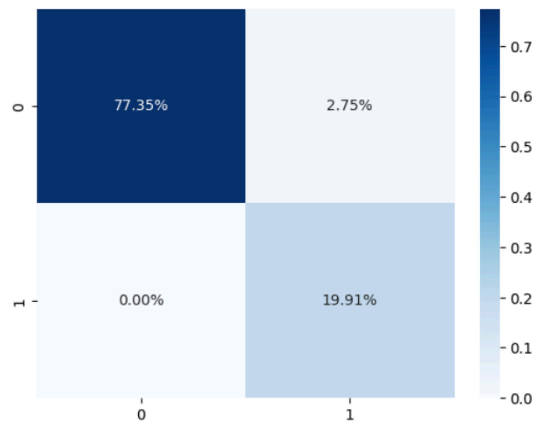


Figure 11: Traditional ML LinearSVC Steady-state Identification Confusion Matrix.

In real-time classification of steady-state data, the LinearSVC model championed a 99.28% accuracy in predicting steady-state operations and 98.56% accuracy in transient operations, evaluated within the same 100s windows as malfunction detectors.

4.2.2 Automated Machine Learning (AutoML) steady-state Classification Model

In the compelling context of ensuring the autonomous system is fortified against adversarial reprogramming attacks and illicit inference alteration, PyCaret's AutoML for classification becomes pivotal. This model, whilst navigating through varied architectures during training, also diligently manages requisite preprocessing. Employing a Stratified K-Fold and a K-nearest neighbors (KNN) model, PyCaret orchestrated a model demonstrating 100% accuracy on the training data, visualized in Figure 12.

During the real-time classification of steady-state data, the AutoML model astoundingly predicted steady-state operations with 99.98% accuracy and transient operations with 98.32% accuracy, parallelly evaluated within the identical 100s windows as the malfunction detectors.

The traditional ML and AutoML steady-state classifiers demonstrated formidable capability in classifying reactor states using SG1 variables with remarkable accuracy in a constrained training period. The variance between the two models was marginal due to the extensive sample and effect size, thereby affording high statistical power.

In the umbrella of "Autonomous System Inference, Trojan, and Adversarial Reprogramming Attack and Defense," it's imperative to highlight that these ML models, while being highly accurate, should be constantly scrutinized and validated to ensure they are not compromised or manipulated through adversarial attacks, ensuring the validity of the inference, especially in critical applications like nuclear reactor management where erroneous classification or inference could pose monumental risks and challenges. Thus, these models become crucial barriers, defending against illicit trojan and reprogramming activities while safeguarding system integrity and ensuring legitimate inferencing throughout system operations.

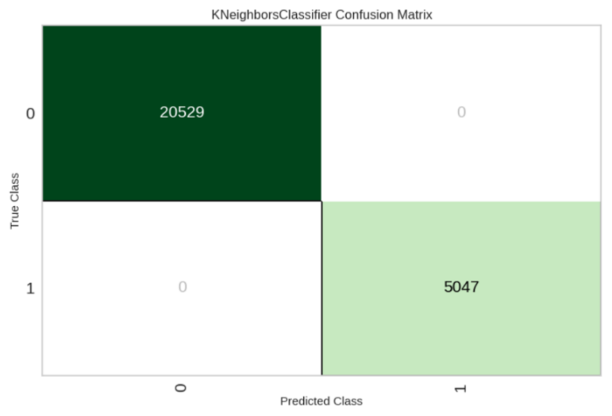


Figure 12: AutoML Steady-state Identification Confusion Matrix

5. Predictive Modeling of I/O in Autonomous System Components

In a bid to ensure optimal functioning within prescribed boundaries, following the classification, the component's I/O are prognosticated. Both the traditional and AutoML models utilize a sequential artificial neural network (ANN) to forecast all seven SG1 tags, with both models being educated on identical malfunction detector and classifier datasets. Real-time data, parsed in 50ms intervals, is absorbed into the models within the aforementioned 100 second window and parameters, scrutinizing model stability temporally. A 1/5 testing to training split is utilized, with targets mirroring the features to realize forecasting. Model fidelity during both training and real-time phases was quantified using the Mean Absolute Error (MAE) as specified in Equation 7.

5.1.1 Traditional Machine Learning (ML) Artificial Neural Network (ANN) Forecasting Model: Adversary-Resilient Forecasting

Ensuring resilience against trojan and adversarial reprogramming, the traditional ML employs a manually hyperparameter-tuned ANN, leveraging Gated Recurrent Unit (GRU) layers complemented by 20% dropout layers. This model forecasts the training dataset in a multivariate manner, utilizing a training window size of 250ms. Figure 13 illustrates the traditional ML's endeavor to predict SG1 total flow rate in 250ms intervals, achieving a training MAE of 0.0600.

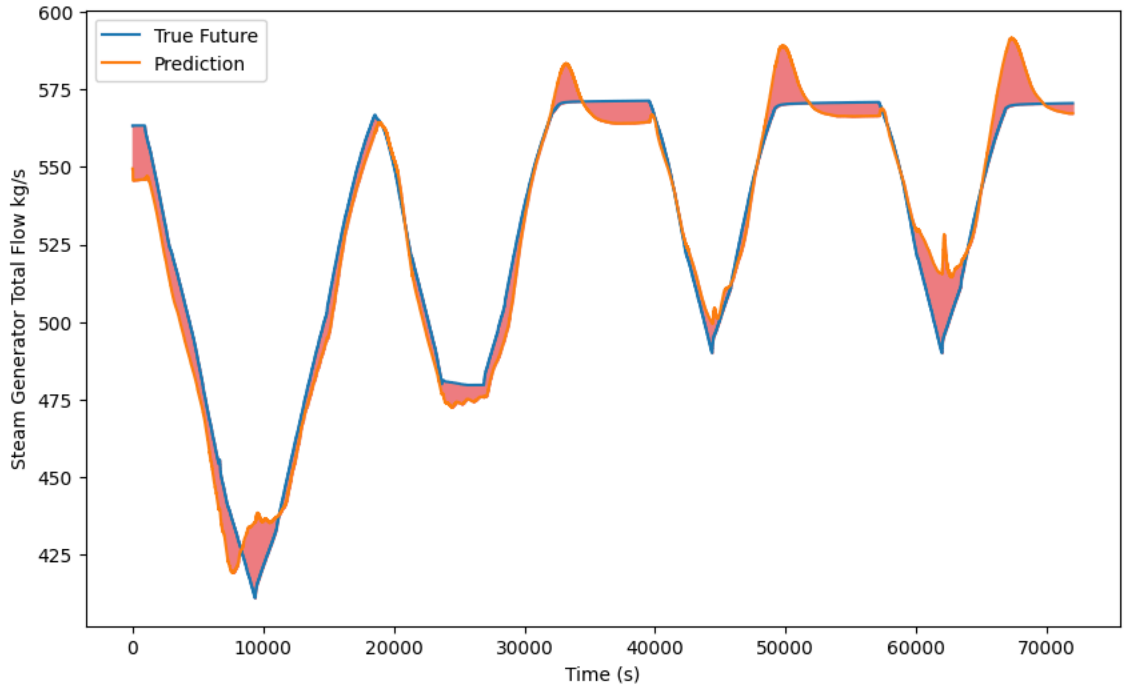


Figure 13: Traditional ML GRU Forecasting SG1 Total Flow During Training.

Under real-time steady-state data evaluation, as presented in Figure 14, the forecasting model modestly forecasted the SG1 total flow, yielding an MAE of 0.2023.

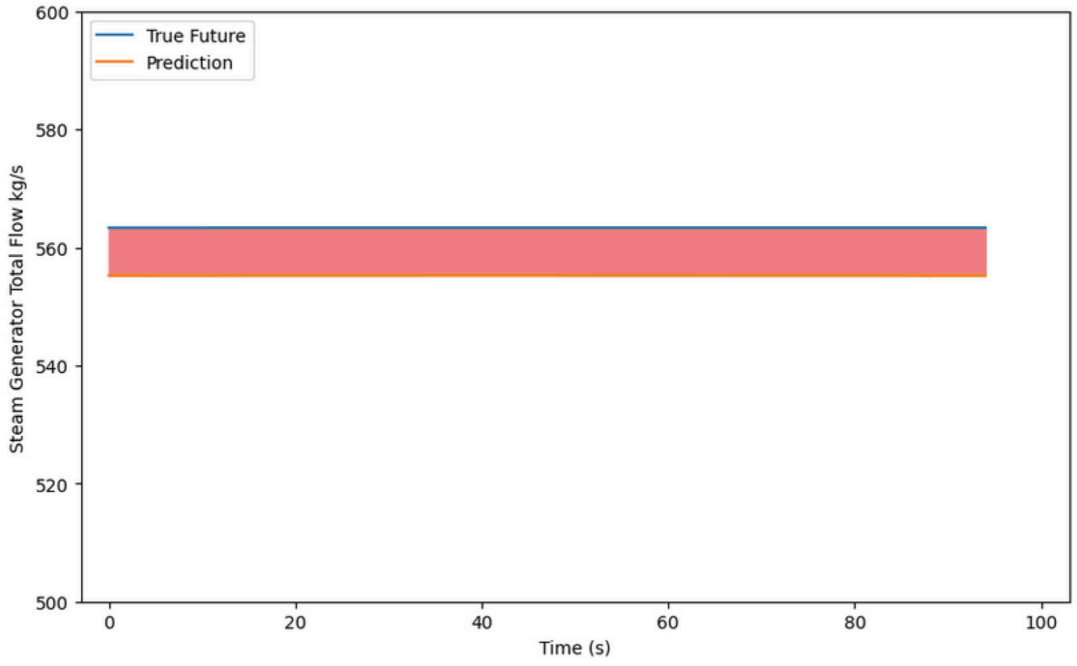


Figure 14: Traditional ML GRU Forecasting SG1 Total Flow During Real-Time Steady-State Operation

Concurrently, during a transient scenario, Figure 15 showcases that the traditional ML forecasting model paralleled steady-state operation performance, posting an MAE of 0.1992 during transient operation.

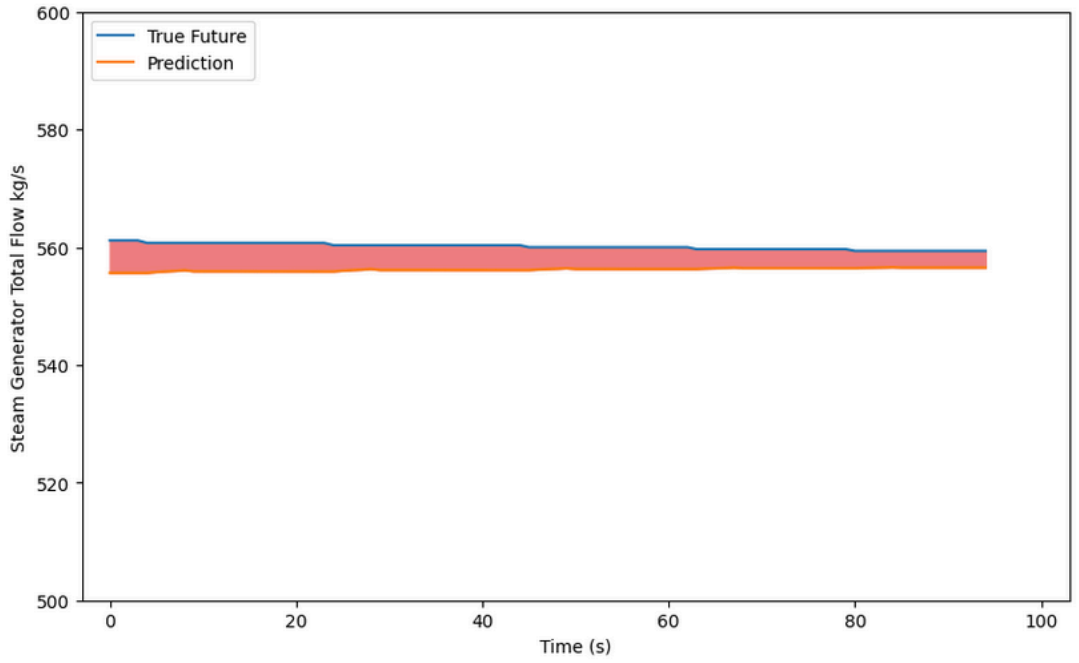


Figure 15: Traditional ML GRU Forecasting SG1 Total Flow During Real-Time Transient Operation.

This traditional ML forecasting component-level DT showcased a notable performance decrement compared to preceding models, attributed to the paucity of features in the SG1 component-level dataset and the substantial data volume requisite for training a multivariate ANN forecasting model.

5.1.2 Automated Machine Learning (AutoML) Artificial Neural Network Forecasting Model: Navigating Through Adversarial Waters

Throughout this study, numerous AutoML frameworks were scrutinized for the forecasting component-level DT. Predominantly, these frameworks were univariate or demanded a more substantial feature set than that SG1 could provision, rendering it appropriate to concoct a simple AutoML framework. This allowed for smaller feature size multivariate forecasting using KerasTuner for hyperparameter tuning across three distinct ML models. Post-training, the model achieving the lowest MAE was preserved as the best model and employed for real-time forecasting.

This framework amalgamates a Long Short-Term Memory (LSTM) ANN with dropout layers, a bidirectional LSTM (BiLSTM) ANN with dropout layers, and a GRU ANN with dropout layers. Layers, neurons, dropout layer size, and optimizer were fine-tuned using a hyperband tuner from the KerasTuner package. The paramount model, a BiLSTM model with optimized hyperparameters, produced an MAE of 0.0116 during the training phase on the identical training dataset used by the traditional ML forecasting model. Figure 16, while showing slight underpredictions during steady-state intervals, accurately forecasted the SG1 total flow.

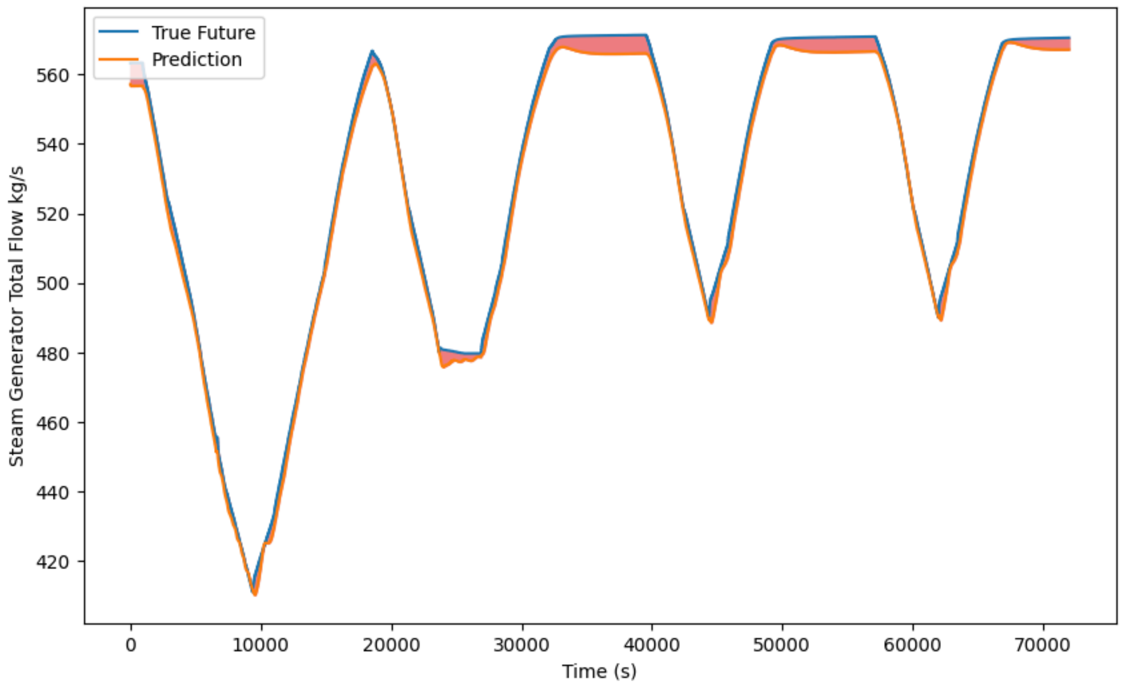


Figure 16: AutoML Forecasting SG1 Total Flow During Training

As it migrated into real-time steady-state validation, the AutoML forecasting model was able to predict the SG1 total flow with an MAE of 0.0166, visualized in Figure 17. The real-time steady-state data utilized the same window previously employed to assess the other models.

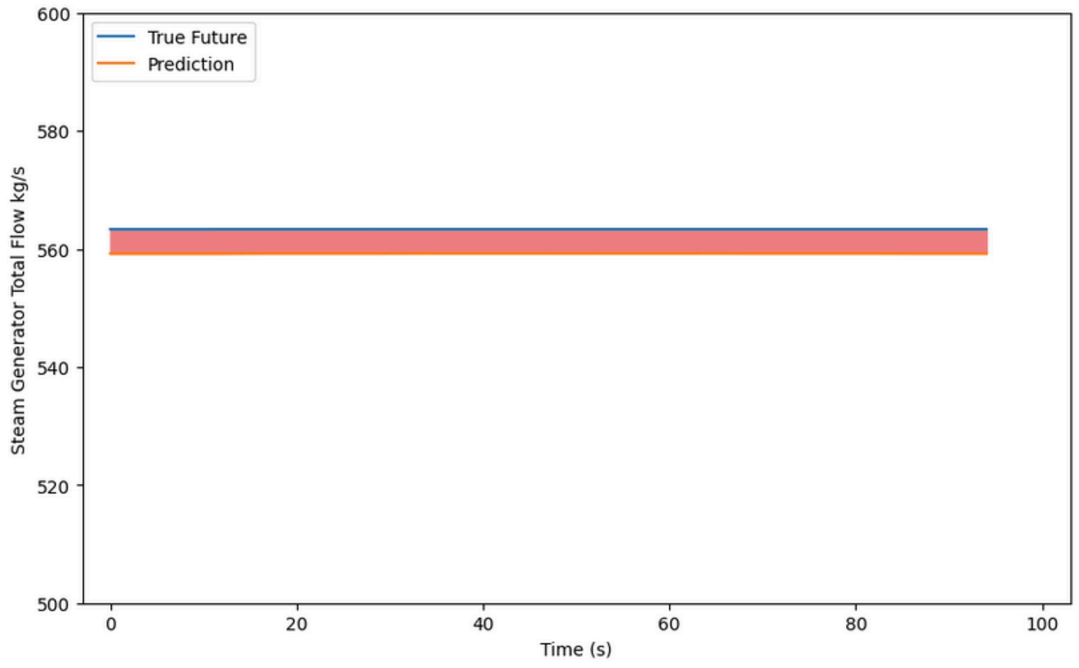


Figure 17: AutoML Forecasting SG1 Total Flow During Real-Time Steady-State Operation.

In a transient scenario, the MAE was 0.0138 on real-time data. Figure 18 demonstrates the predicted SG1 total flow juxtaposed with the true SG1 total flow over time, utilizing the same real-time dataset as the remainder of the models to assure model fidelity across an identical timeframe.

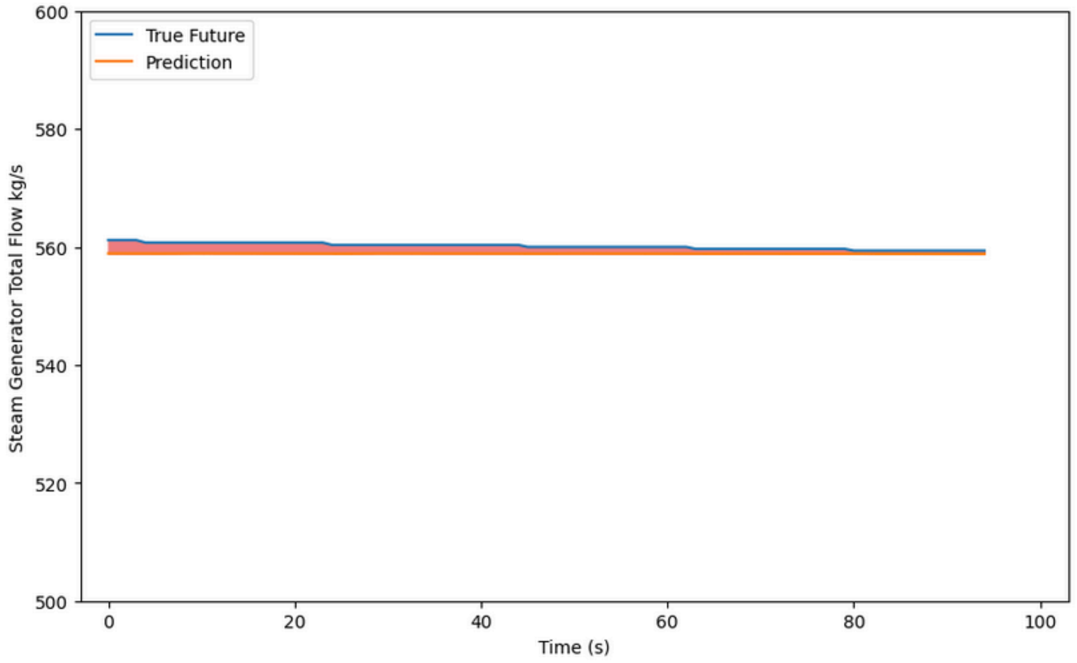


Figure 18: AutoML Forecasting SG1 Total Flow During Real-Time Transient Operation

Though the AutoML forecasting model yielded significantly lower MAE results than the traditional ML forecasting model, it did so at the sacrifice of markedly higher training time given identical batch sizes. The traditional ML forecasting model necessitated 15 minutes of training on an NVIDIA Tesla T4 GPU compared to 20 hours on the same GPU for the AutoML forecaster's training procedure. To mitigate computational expense, AutoML hyperparameter options can be limited, reducing the number of tested model architectures.

6. Simulated Cyber-Attacks: Targeting ML Inferences and DTs

With a surge in the implementation of Machine Learning (ML) and Automated Machine Learning (AutoML) in Advanced Control Systems (ACS) for managing critical infrastructure like reactor systems, the vulnerability of these autonomous systems to cyber-attacks, particularly those that manipulate their inference mechanisms, poses significant risks. Underscoring this, the interplay between ML models and ACS, given their intersectionality in enabling and optimizing controlled environments, necessitates a robust framework for safeguarding against potential cyber-physical exploits.

To formulate a credible threat model that focuses on exploiting the autonomous system's inference mechanisms in advanced reactor systems, which employ ML-based Digital Twins (DTs) and ACS controls, a detailed kill chain was constructed, utilizing the Industrial Control Systems (ICS) Kill Chain from the SANS Institute and ICS techniques from the MITRE ATT&CK framework.

The envisioned scenario involves an insider threat, wherein a third-party entity has access to the ACS internal network during a reactor outage. Enveloping this context, the subsequent discourse endeavors to comprehend the depth and scope of adversarial encroachments, scrutinizing their implications on system stability and exploring contingent mitigation strategies that can be structured into the operative realm of ACS.

A hypothetical adversary is presented with four prerequisites:

1. Physical access to the ACS network.
2. Utilization of external computers for cyber-attacks (transient cyber assets).
3. Wired connection to the ACS and SQL server via the SG1 cyber-physical testbed.
4. Comprehensive understanding of ACS architecture and hardware implementation.

These prerequisites provide a feasible pathway for initial access into the internal reactor network. A strategic approach could involve disguising the attack as ML retraining to counteract model drift. Three critical targets within the ACS functionality have been identified: training data, real-time data, and ML models. Both traditional ML and AutoML frameworks were subjected to each attack to evaluate the cybersecurity risks each framework posed under post-deployment adversarial action.

6.1 Cyber-Attacks Targeting Machine Learning (ML) Training Data

The assault on the ML training data began with the establishment of a MySQL 5.7 server on the SQL server computer, adhering to recommended security measures from Roller [26]. During the discovery phase, Wireshark [27] was deployed to sniff MySQL greeting and login request packets to identify the SQL server and the ACS, assuming that the Programmable Logic Controller (PLC) was offline due to a shutdown during the outage.

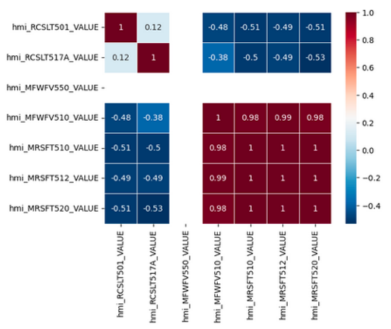


Figure 19: SG1 tag correlation matrix.

The exploitation process involved utilizing the salted Secure Hashed Algorithm-1 (SHA-1) hashed password from the login request packet, with Hashcat [28] facilitating the brute-force decryption of the password utilizing a comprehensive password dictionary. Once a valid username and password were obtained, the adversary gained access to the SQL server as part of the persistence phase, while the ACS login credentials were set to read-only for the database.

In a cleverly designed Man-in-the-Middle (MiTM) attack, the adversary intercepted queried data from the SQL server, revealing the ACS's training dataset for the plant-level DT and SG1 component-level DTs. The training data was extracted and utilized to construct a Pearson correlation matrix of each of the SG1 tags to ascertain the most consequential tag for manipulation (Figure 19).

6.2 Impact of Training Data Attacks on Traditional ML Digital Twins (DTs)

Notably, a trojan was injected into the SG1 feedwater flow, manipulated through a series of SQL queries to consistently exhibit the maximum value within the training dataset, simulating an ongoing Large Break Loss of Coolant Accident (LBLOCA). Subsequent obfuscation of the SQL logs masked the adversarial intrusion and alterations to the server.

The repercussions were immediately evident upon the restart and retraining of the ACS, affecting both the traditional ML and AutoML ACS implementations symmetrically. Despite adversarial alterations to the SG1 feedwater flow in training data, the impact on the plant-level DT and the component-level transient classifier DT was minimal. The component-level forecaster DT experienced the most significant impact, predicting a scenario absent from the training dataset (Figures 20-22).

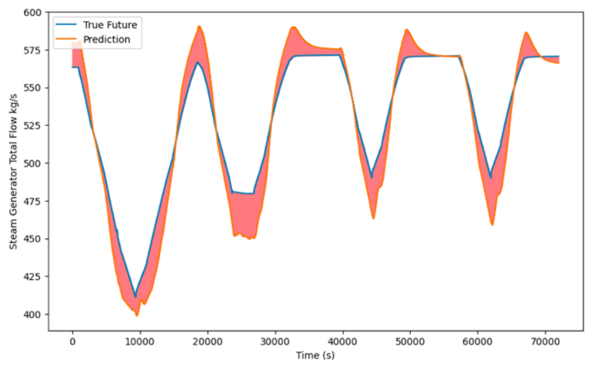


Figure 20: Training data attack effect on traditional ML GRU forecasting SG1 total flow during training.

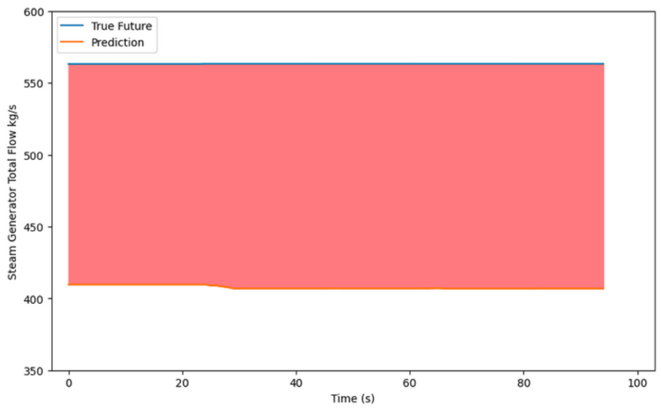


Figure 21: Training data attack effect on traditional ML GRU forecasting SG1 total flow during real-time steady state operation.

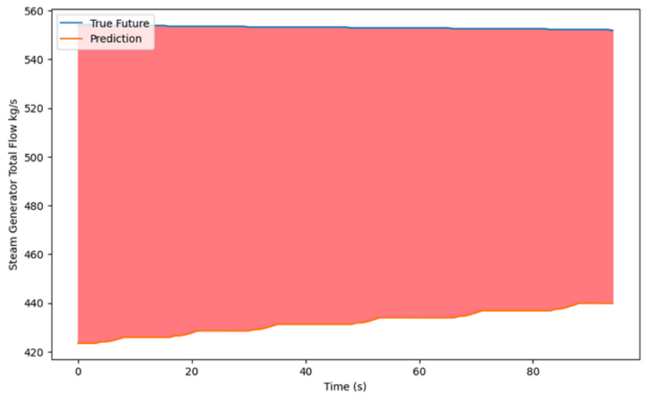


Figure 22: Training data attack effect on traditional ML GRU

The attack's effects multiplied during real-time transient scenarios, producing an MAE of 5.0167, as shown in Figure 22. The significant underprediction of the forecasting model without the malfunction detection plant-level DT and component-level transient classifier DT being affected can negatively impact ACS control decisions where the plant is not viewed in a malfunction state. However, the SG1 total flow is predicted to be less than the actual value, causing the ACS to potentially over-compensate for the perceived loss of water by increasing feedwater into the system with the potential of making the system go solid or inadvertently SCRAM.

6.3 Impact on AutoML Digital Twins (DTs) Resulting from Training Data Attacks

The AutoML forecasting model was notably more susceptible to the training data attack compared to the traditional ML model, exhibiting an increase in Mean Absolute Error (MAE) from 0.0116 to 0.0974 during the training phase Figure 23 shows the AutoML model's poor performance on training data forecasting, where the model was slightly more accurate than the traditional ML only by 4.36% MAE difference as opposed to the 4.84% MAE difference initially.

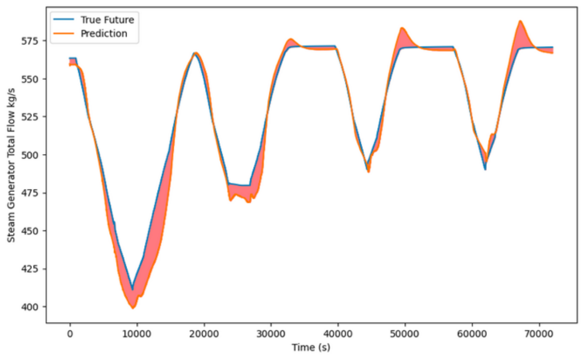


Figure 23: Training data attack effect on AutoML forecasting SG1 total flow during training.

During real-time steady-state data prediction, the AutoML forecasting model's MAE was 3.2476, which is higher than the MAE of the traditional ML forecasting model during the same time window. The large overprediction of the SG1 total flow is shown in Figure 24, implying that cyberattacks more impact the AutoML during real-time steady state forecasting than traditional ML methods.

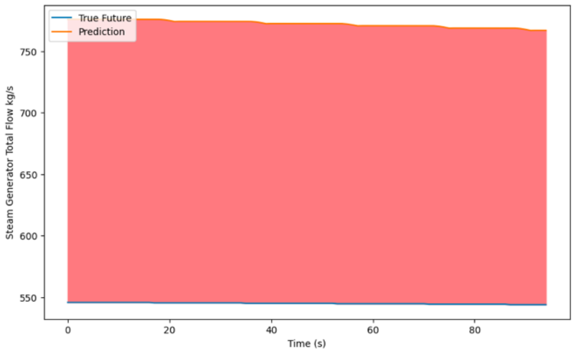


Figure 24: Training data attack effect on AutoML forecasting SG1 total flow during real-time steady-state operation.

When forecasting real-time transient operations, the AutoML forecasting MAE increased to 7.1770, overpredicting the SG1 total flow by 210 kg/s, as shown in Figure 25. The overprediction of flow without a malfunction being detected may result in the ACS under-compensating for the amount of water in SG1, causing the water levels to decrease to prevent the system from going solid, potentially causing SG1 to dry out or cause an inadvertent SCRAM.

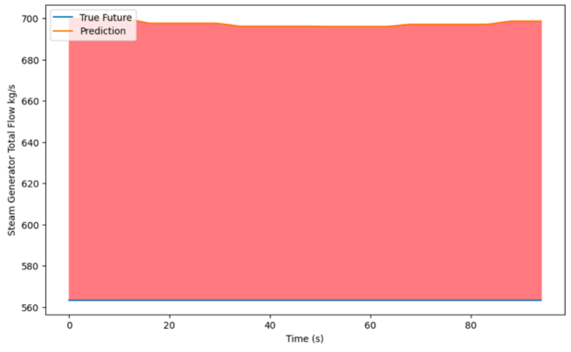


Figure 25: Training data attack effect on AutoML forecasting SG1 total flow during real-time transient operation.

The training data cyber-attack for traditional ML and AutoML are compensated through safety systems such as pressure relief valves and injection systems. Additional cyberattacks on the safety systems must be carried out to achieve a departure from nucleate boiling from SG1 going solid or drying out following incorrect ACS decisions. Additionally, SG2 could compensate for a faulty SG1 allowing for safe operation. The most plausible (and best-case) scenario is a shutdown and investigate action being taken following the SG1 compromise, allowing operators to perform root-cause analysis with the assistance of the plant's cybersecurity team.

6.4 Cyber-Attacks against Real-Time Data Acquisition in ACS

Subsequent cyber-attacks targeted real-time data ingestion into the ACS by injecting false data from the PLC to the ACS. Utilizing Wireshark, network traffic was sniffed, isolating CIP packets from the PLC to the ACS, and a MiTM attack positioned the adversary between the ACS and PLC to surreptitiously read and write CIP packets. The parsing of PLC commands and tags from CIP packets utilized Allen Bradley PLC documentation and Rockwell Automation's specific message request fields, available on Automation [29].

Table 1: Allen Bradley PLC CIP Read Tag Table Example

Message Request Field	Bytes (in hex)	Description – Symbolic Segment Addressing
Request Service	4C	Read Tag Service (Request)
Request Path Service	06	Request Path is six words (12 bytes) long
Request Path	91 0A 70 6F 73 46 43 46 56 30 30 37 35	ANSI Symbolic Segment for *posFCFV0075*
Request Data	01 00	Number of elements to read (1)
Reply Service	CC	Read Tag Service (Reply)
Reserved	00	
General Status	00	Success
Extended Status Size	00	No extended status
Reply Data	CA 00	REAL (single precision float) Tag Type Value
	56 0E 47 42	49.764 (decimal)

Table 2: Allen Bradley PLC CIP Write Tag Table Example

Message Request Field	Bytes (in hex)	Description – Symbolic Segment Addressing
Request Service	4D	Write Tag Service (Request)
Request Path Service	06	Request Path is six words (12 bytes) long
Request Path	91 0A 70 6F 73 46 43 46 56 30 30 37 35	ANSI Symbolic Segment for *posFCFV0075*
Request Data	01 00	Number of elements to write (1)
	00 00 F0	30 (decimal)
Reply Service	CD	Write Tag Service (Reply)
Reserved	00	
General Status	00	Success
Extended Status Size	00	No extended status

Targeting the SG1 feedwater control valve position (posFCFV0075) for false data injections demonstrated the capacity for nuanced manipulation of real-time data to initiate a reactor trip during steady-state operation, exemplifying a critical vulnerability in the autonomous system's inference mechanisms and emphasizing the imperative for robust defense mechanisms against such adversarial reprogramming attacks.

This rework concentrates on the vital aspect of autonomous system inference within the presented scenario, showcasing targeted adversarial reprogramming and attack strategies which deliberately manipulate the ML models and training data. Developing resilient, counteractive measures that prioritize the safeguarding of autonomous system inference mechanisms against such attacks is paramount for ensuring the cybersecurity and functional integrity of ACS in critical infrastructure. The breadth of this subject matter and the hypothetical scenarios presented herein underscore the necessity of further research and development in this domain, particularly in crafting and optimizing defense mechanisms that safeguard against the manipulation of autonomous system inference.

7. Assessment of Trojan Attacks on ML Training and Real-Time Data

Trojan attacks within the realm of Machine Learning (ML) pose significant security challenges, especially when considering applications within critical infrastructures like nuclear reactors. When models are compromised during the training phase, malicious agents can embed detrimental actions or responses that get triggered by specific inputs during real-time operations.

7.1 Implementing and Defending Against Trojan Attacks in ML Models

Trojan attacks in ML involve injecting malicious data or exploits during training that are designed to trigger specific misbehaviors when the model encounters certain inputs in deployment. These attacks could be considerably harmful, especially in applications involving autonomous systems where reliance on predictive data is pivotal. Analyzing previous sections, we can interpolate methodologies for trojan attack implementations in ML training and real-time data.

7.1.1 Mechanics of Trojan Attacks in ML Models

Typically, a trojan attack on an ML model involves manipulating the training data to include specific inputs (or "triggers") that, when recognized by the model during operation, initiate an altered, malicious response. The threat agents craftily integrate these triggers to be inconspicuous amidst normal data, ensuring they do not raise alarms during routine operations or audits.

For example, in the dataset you shared earlier about the power output of a reactor system, a trojan attack might manipulate the ML model in such a way that when it identifies a particular pattern of data (like a specific power output level, or a particular sequence of power changes), it triggers an unsafe action like overriding safety controls or initiating an unrequired shutdown of systems. This could be implemented by subtly altering the training data so that the ML model learns this malicious response as if it were a legitimate, safe action.

Potential Ramifications: The consequences of such attacks can be vast and catastrophic, especially in critical systems like nuclear reactors, where precision and reliability are paramount. The misbehaviors triggered by a trojan could range from inefficient operations to serious safety breaches, depending on the nature of the malicious instruction embedded during training.

7.1.2 Defensive Strategies

Defending against trojan attacks in ML models demands meticulous training data scrutiny, model monitoring during operations, and robust anomaly detection mechanisms.

- **Strategy 1: Rigorous Data Validation and Scrutiny** Implement stringent checks and validations for training data. Employ anomaly detection techniques to identify and scrutinize data patterns that could potentially be trojan triggers embedded within the dataset.
- **Strategy 2: Secure Model Training Environments** Ensure that the environment where models are trained is secured against unauthorized access and data manipulation, utilizing cybersecurity best practices to safeguard against infiltration.
- **Strategy 3: Real-time Model Monitoring and Anomaly Detection** Monitor the ML model's decisions and outputs in real-time during operation. Implement advanced anomaly detection that can identify and alert to unexpected or unexplained model behaviors, potentially highlighting the activation of a trojan.

By intertwining rigorous data validation, a secure training environment, and dynamic real-time monitoring, we form a multifaceted defensive strategy that fortifies ML models against the intricate threats posed by trojan attacks.

7.1.3 Cyber-Attacks Against ML Training Data

Trojan attacks against ML training data would typically involve altering the dataset to include inputs that, when recognized by the trained model, would trigger unwanted behaviors. The trigger might be a specific pattern or anomaly in the input data that would be benign or inconspicuous to humans but has been inserted into the training data enough times for the ML model to learn it as a valid input [32]. Here are some viable options:

a. **Poisoning Attacks:** In this context, the attacker injects malicious data into the training set, which influences the learning process to produce a malicious payload upon encountering specific triggers in real-time operation. Poisoning could manipulate the dataset in a way that when a particular pattern of data (the trigger) is encountered, the ML model responds in a predefined malicious way.
b. **Backdoor Attacks:** This involves modifying the algorithm's training data to include specific patterns that will later serve as a backdoor for attackers [33]. For instance, altering the training dataset of an AutoML digital twin to recognize a subtle but artificial pattern as normal, which could later be exploited to mislead the system during real-time operation without triggering anomaly detection systems.

Defensive Mechanisms

a. **Data Sanitization:** Regularly scrutinize and sanitize training data to eliminate the possibility of poisoned data or backdoor triggers.
b. **Anomaly Detection:** Employ auxiliary anomaly detection mechanisms to catch unexpected behaviors from the ML model and revert to safe states in autonomous systems.
c. **Secure Training Environments:** Ensuring that the training environment is securely isolated and well-protected against potential attackers.

7.1.4 Cyber-Attacks Against ML Real-Time Data

Attackers could leverage trojan triggers in real-time data to mislead operational ML models. These triggers, if well designed, might be challenging to detect using traditional cybersecurity mechanisms due to their benign appearance. Options include:

a) **Signal Manipulation:** Introducing a subtle trojan trigger in real-time input signals to ML models, to invoke malicious payloads that have been embedded during the training phase.
b) **Deceptive Attacks:** Such as attacks that make the model misinterpret the trojaned input, in the realm of false data injection, causing undesired outputs or system states.

Defending against these would involve meticulous analysis and validation of real-time data and implementing robust detection mechanisms.

Defensive Mechanisms:

a) **Input Verification:** Developing mechanisms to verify and validate real-time input data against expected patterns and using secure baselines.
b) **Real-time Monitoring:** Ensuring that ML model outputs are continuously monitored, and anomalous results are investigated promptly.
c) **Use of Certified Data:** Ensuring that real-time operational data is acquired from verified and secure sources to prevent possible contamination.

7.2 Adversarial Reprogramming Attack and Defense

Adversarial reprogramming involves manipulating machine learning (ML) models by reprogramming them to perform tasks they weren't originally designed for. This type of cyber-attack can repurpose ML models to act against their intended applications, effectively diverting them from their legitimate functionality and potentially causing adverse consequences in real-world operations.

7.2.1 Implementation of Adversarial Reprogramming

In the context of an ML-based Advanced Reactor System, adversaries could implement adversarial reprogramming by altering the model's decision-making parameters or injecting malicious payloads that exploit vulnerabilities in the ML model. This could be performed by exploiting the communication between the ML model and the system it controls, as demonstrated by attacking the ACS and PLC communication. The attackers might manipulate the input space of ML models to induce incorrect predictions or behavior.

Example Scenario: An attacker could manipulate the ML model that controls the reactor system, causing it to misinterpret normal operation data as indicative of a system failure. This could then prompt the model to unnecessarily shutdown systems or mismanage reactor cooling processes, potentially endangering equipment and personnel.

7.2.2 Defense Against Adversarial Reprogramming

To defend against adversarial reprogramming, secure coding practices, rigorous validation, and verification of ML models are paramount. Cybersecurity measures should be implemented to safeguard communications between ML models and controlled systems, utilizing encryption and secure channels.

- **Strategy 1:** Employ a zero-trust security model, where all communications and instructions, even from internal systems, are validated and authenticated.
- **Strategy 2:** Implement anomaly detection systems that can identify irregular instructions or anomalous behaviors stemming from the ML models, potentially highlighting adversarial reprogramming attempts.
- **Strategy 3:** Ensure that ML models are robust to adversarial inputs, meaning they can recognize and resist attempts to manipulate their operation.

7.3 Trojan Attacks (Future Research)

Given the complexity and severity of potential trojan attacks against ML models in critical infrastructure such as Advanced Reactor Systems, future research is vital to anticipate, understand, and mitigate such threats.

7.3.1 Identification and Understanding of New Threat Vectors

Continuous research to unearth new potential threat vectors is crucial. Understanding how trojan attacks could be implemented against ML models in novel and unanticipated ways will be central to developing resilient systems.

Aspect 1: Explore novel methods of injecting trojan payloads into ML models, particularly during the training phase.

Aspect 2: Investigate the potential for trojan attacks to manipulate ML model behavior during real-time operation, focusing on how subtle, gradual modifications over time (as opposed to overt, immediate changes) might be utilized by adversaries to avoid detection.

7.3.2 Development of Advanced Defense Mechanisms

Equally critical is the ongoing development and refinement of defense mechanisms against trojan attacks, ensuring they remain ahead of evolving threat methodologies.

Topic 1: Investigate the development of more advanced intrusion detection systems that can identify and neutralize trojan payloads in ML models.

Topic 2: Explore the application of ML to cybersecurity defense in this context, developing models capable of identifying and mitigating attempts to compromise their functionality or decision-making processes.

7.3.3 Policy, Regulation, and Standardization

Exploring policy and regulation dimensions to establish standards for the protection of ML models in critical systems against trojan attacks will form an integral part of future research.

Focus 1: Assess the current policy and regulatory landscape, identifying gaps and areas where new policies or regulations are needed to ensure the security of ML models in critical infrastructure.

Focus 2: Develop standards and best practices for safeguarding ML models against trojan attacks, providing guidelines for implementation and maintenance of cybersecurity defenses in the context of ML.

8. Exploring Adversarial Reprogramming of ML Models in Autonomous Systems

Adversarial reprogramming introduces nuanced challenges into the realm of Machine Learning-based Digital Twins, particularly examining its susceptibilities and potential impacts in the context of "Cyber Risk Analysis of Machine Learning (ML)-based Digital Twins (DTs)". Adversarial reprogramming involves manipulating an ML model’s input to compel it to perform tasks it was not originally intended to perform. Essentially, it harnesses the pre-trained model’s parameters without altering its weights, directing it towards alternate tasks, which in the context of DTs, could compromise the accuracy and reliability of simulations and predictions.

Three pertinent scenarios are conceptualized where adversaries might exploit adversarial reprogramming within ML-based DTs:

Misguiding Predictive Maintenance: Through a sophisticated and surreptitious manipulation of input data, an adversary could employ adversarial reprogramming to misdirect the Machine Learning (ML) model, leading it to furnish erroneous predictions regarding system failures or imperative maintenance schedules. This malevolent action could inevitably propel the system towards unnecessary, costly downtimes or, at the opposite end of the spectrum, cause it to ignore critical maintenance intervals, placing the system at risk of unanticipated failure. Both these outcomes not only present discernible operational challenges but also cast a looming shadow over the overarching safety and reliability of the entire operation, thereby escalating the intrinsic risks and potentially jeopardizing the operational efficacy of the plant.

Distorting Simulations: Leveraging the subversive technique of adversarial reprogramming to insidiously modify input variables, adversaries could manipulate the Digital Twin’s (DT’s) simulations, contriving a distorted, and thus unreliable, digital representation of the physical system. This distortion, albeit subtle, could systematically misinform decision-making processes and subsequently instigate misguided, and potentially detrimental, actions based upon untrustworthy and compromised data. Moreover, this alteration in the DT’s simulation outputs not only sows seeds of mistrust in the system’s reliability but also potentially jeopardizes the integral safety and operational stability, making the rectification of such deceptive alterations imperative to uphold system integrity.

Impairing Anomaly Detection: By skillfully manipulating the input data fed to the ML model, adversaries possess the capacity to mislead the DT into inaccurately classifying malicious or anomalous activities as benign, normal behaviors. This subtle yet destructive interference not only undermines prevailing security protocols but also potentially facilitates harmful actions to navigate through the system’s defenses, progressing undetected and unchallenged. Consequently, this impairment in the anomaly detection capabilities of the DT casts a pervasive vulnerability across the system, posing tangible threats to its stability, security, and operational continuity, while also necessitating a reevaluation of the system's vulnerability and threat mitigation strategies.

Considering the above, let's delve into a notional example. The Digital Twin, designed to optimize and monitor a nuclear reactor's performance, employs an ML model trained to predict maintenance needs based on various input parameters (e.g., temperature, pressure, and radiation levels). The adversary, leveraging adversarial reprogramming, manipulates the input parameters to mislead the model into inaccurately predicting the reactor’s maintenance needs. This can be done by adding perturbations to the input data, redirecting the model’s outputs towards deceitful predictions. Such manipulated forecasts might suggest that the reactor is operating optimally when critical maintenance is required, or vice versa, leading to unnecessary shutdowns.

Here is a notional code example where an adversary could manipulate input data to mislead the DT's predictive maintenance model:

```
import numpy as np
import joblib as jb

# Load the ML model
model = jb.load('predictive_maintenance_model.pkl')

# Adversarial reprogramming: introducing subtle, misleading alterations in the input data
def adversarial_reprogramming(original_data):
   perturbation = np.random.normal(loc=0, scale=0.01, size=original_data.shape)
   adversarial_data = original_data + perturbation
   return adversarial_data

# Original input data
original_data = np.array([...])  # array of original input data

# Generate adversarial data
adversarial_data = adversarial_reprogramming(original_data)

# Misleading the model with adversarial data
false_predictions = model.predict(adversarial_data)

# Save the adversarial data for future use or analysis
np.save('adversarial_data.npy', adversarial_data)
```

In this script, the adversary introduces subtle perturbations to the original input data, creating adversarial_data that is then fed into the predictive maintenance model, potentially leading to false predictions and suboptimal decision-making regarding the reactor’s maintenance needs.

Considering the discussed scenarios, securing DTs against adversarial reprogramming is critical. Employing robust adversarial training, incorporating redundant safety checks, and continuously monitoring and validating the DT’s outputs against expected or plausible outcomes emerge as pivotal strategies. This ensures that DTs within the nuclear sector, or any industry, remain resilient against such cyber-physical threats, safeguarding operational integrity, safety, and data veracity in the interconnected digital realm.

9. Cyber Risk Evaluation of ML Inference in DTs

In the nuclear sector, risk – typically articulated as the product of likelihood and impact and quantified in terms of failure rates – becomes particularly intricate when situated within the cyber domain. With the integration of Machine Learning (ML)-based Digital Twins (DTs) in Advanced Control Systems (ACS) in this sector, the analysis of cyber-risk acquires a qualitative and deeply contextual characteristic. Three levels, namely low, medium, and high, are utilized on the likelihood and impact axes to scrutinize the cyber-risk intrinsic to ML-based DTs.

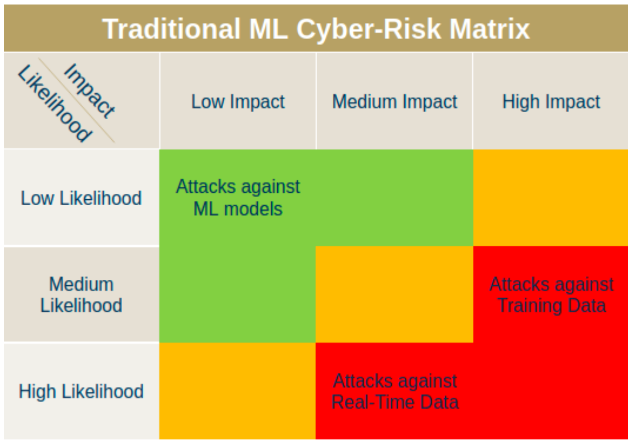
9.1 Analyzing the Cybersecurity Risk Spectrum

Considering simple cyber-attacks previously conducted and the resultant extrapolative consequences, a scoring system is proposed for attacks against training data, real-time data, and ML models, considering their likelihood and impact relatively. Our risk assessment accommodates a hypothetical scenario of facing highly adept threat actors, capable of orchestrating stealthy attacks with expertise spanning cybersecurity, nuclear engineering, and ML.

Attack Potential and Anticipated Risks

- **Likelihood:** Determined by the complexity and requisite insider knowledge to execute the attack.
- **Impact:** Assessed by the resultant alteration in the model's accuracy metrics post-attack.

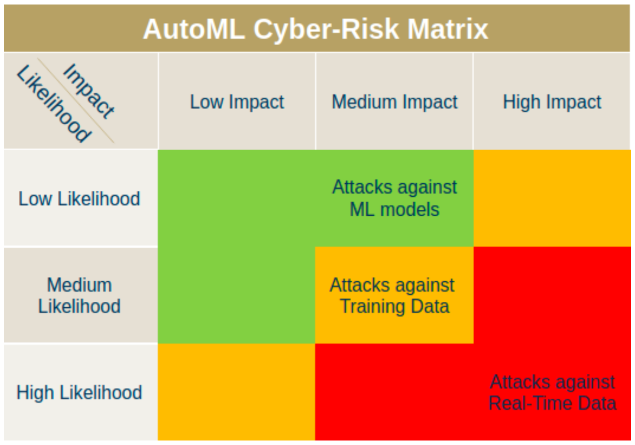
Figure 27 and Figure 26 respectively visualize the risk associated with different attacks on traditional ML and AutoML matrices – the likelihood remaining constant but the impact being variable across them.



Traditional ML Cyber-Risk Matrix

Likelihood \ Impact	Low Impact	Medium Impact	High Impact
Low Likelihood	🟩 Attacks against ML models	🟩	🟧
Medium Likelihood	🟩	🟧	🟥 Attacks against Training Data
High Likelihood	🟧	🟥 Attacks against Real-Time Data	🟥

Figure 27: The three cyber-attacks on a traditional ML cyber-risk matrix.



AutoML Cyber-Risk Matrix

Likelihood \ Impact	Low Impact	Medium Impact	High Impact
Low Likelihood	🟩	🟩 Attacks against ML models	🟧
Medium Likelihood	🟩	🟧 Attacks against Training Data	🟥
High Likelihood	🟧	🟥	🟥 Attacks against Real-Time Data

Figure 26: The three cyber-attacks on an AutoML cyber-risk matrix.

In Figure 26, attacks against training and real-time data on traditional ML models are allocated within the high-risk (red) zone, necessitating prioritized protective measures against these vectors. Conversely, the primary high-risk event in implementing AutoML model ACS revolves around attacks against real-time data.

9.2 Diving into Specific Attack Vectors and Their Impact

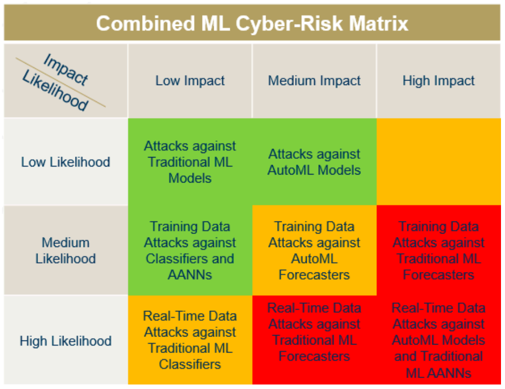
Training Data Attacks: Positioned in the high-impact and medium-likelihood zone, exemplified by SQL injections illustrated earlier in this report, are significant yet relatively straightforward to execute. Historical data, such as ML training data, is often housed within well-established database structures like MySQL, MongoDB, and InnoDB, are potentially vulnerable to exploits accessible from online repositories.

Real-time Data Attacks: Placed in the red zone, such attacks were seen to manipulate ML malfunction detection and forecasting models by, for instance, altering data injections to mimic a malfunction, potentially prompting the ACS to initiate an unneeded reactor SCRAM. With tools like Wireshark, Metasploit, or Scapy widely available, these attacks have a high likelihood due to the ease of execution.

Model Attacks: Attacks against ML models are considerably less impactful and are less likely due to the technical and specific knowledge needed and because they tend to be overwritten during training and are not heavily relied upon during real-time testing.

9.3 Comparative Risk Assessment for Traditional ML vs. AutoML

AutoML models, while sensitive to training data alterations, demonstrated an inherent safety mechanism, activating safety systems and inducing reactor SCRAM before consequential damage could occur. Attacks on AutoML models were relegated to the low-likelihood and medium-impact zone due to the requirement of substantial, specific knowledge about the AutoML package and implementation, and due to their comparatively minor impact.



Combined ML Cyber-Risk Matrix

Impact / Likelihood	Low Impact	Medium Impact	High Impact
Low Likelihood	Attacks against Traditional ML Models	Attacks against AutoML Models	
Medium Likelihood	Training Data Attacks against Classifiers and AANNs	Training Data Attacks against AutoML Forecasters	Training Data Attacks against Traditional ML Forecasters
High Likelihood	Real-Time Data Attacks against Traditional ML Classifiers	Real-Time Data Attacks against Traditional ML Forecasters	Real-Time Data Attacks against AutoML Models and Traditional ML AANNs

Figure 28: The three cyber-attacks on a combined ML cyber-risk matrix.

9.3.1 Emphasizing Protective Measures Across Various Cyber-Attack Categories

As depicted in Figure 28, both ML frameworks necessitate keen attention to protective measures against real-time data attacks, with training data protection following in priority, while ML model protection, given its comparatively lower risk of attack, is deemed less critical.

The study, titled "Cyber Risk Analysis of Machine Learning (ML)-based Digital Twins (DTs)," delves into the potential risks and consequences of adversarial and trojan attacks on the inference mechanisms of DTs, especially within the critical infrastructure of the nuclear sector. This encompasses not only the likely technological disruptions and damages but also extends to potentially catastrophic impacts, especially when one considers the sensitivity and potential repercussions associated with reactor operations.

9.4 Future Directions

This analysis reinforces the imperative to establish robust, multilayered cybersecurity strategies that safeguard against potential cyber-attacks – both common and sophisticated – ensuring the integrity and reliability of ML-based DTs in the nuclear sector, especially considering their pivotal role in managing and safeguarding critical operational data. This underscores the quintessential role of rigorous cyber-risk analysis in facilitating the holistic protection of ML-based DTs within intricate, high-stakes operational landscapes like the nuclear sector.

10. Strategies for Defending Against Inference, Trojan, and Adversarial Attacks

Given the integral role of Machine Learning (ML) models in Digital Twin (DT) technologies and the Autonomous Control System (ACS) within advanced reactor designs, securing them against cyber-attacks, particularly Inference, Trojan, and Adversarial attacks, is imperative. Herein, we delineate defensive measures that specifically target protecting inference mechanisms and preventing unauthorized reprogramming, whilst recalibrating the existing "Recommended Safeguards and Security Measures."

Fortifying Against Inference Attacks

To shield the ML models from inference attacks, which target the extraction of proprietary or sensitive information during the model's inference phase, the following strategies should be employed:

- **Implement Differential Privacy**: Ensure that query responses are perturbed in a manner that protects the training data from being reverse engineered, whilst maintaining model utility.
- **Deploy Homomorphic Encryption**: Enable the model to make predictions while the input data remains encrypted, thus safeguarding against unwanted information leaks during the inference process.
- **Utilize Secure Multi-party Computation**: Engage in a collaborative computational methodology wherein parties can jointly compute a function without revealing their inputs to each other.

Thwarting Trojan Attacks

Preventing Trojan attacks, which seek to manipulate the model by introducing malevolent data or triggering mechanisms, involves:

- **Conduct Rigorous Input Validation**: Ensure that all incoming data, especially those used for retraining, are thoroughly scrutinized and validated against expected patterns and baseline metrics.
- **Employ Outlier Detection**: Leverage anomaly detection mechanisms to identify and quarantine aberrant inputs that might be engineered to initiate a Trojan attack.
- **Continuous Model Monitoring**: Establish a regime of ongoing model behavior monitoring to detect any unexpected prediction patterns or deviations that could indicate the presence of a Trojan.

Mitigating Adversarial Attacks

For adversarial attacks, where attackers introduce subtly modified inputs designed to mislead the ML model into making incorrect predictions, the mitigation strategies include:

- **Employ Adversarial Training**: Integrate adversarially perturbed data into the training process to enhance the model's resilience against adversarial inputs.
- **Implement Robustness Checks**: Regularly subject the model to a range of perturbed inputs and monitor its responses, ensuring that it is not easily swayed by adversarial alterations.
- **Input Sanitization**: Apply filters or transformation functions on inputs to neutralize the effect of adversarial manipulations before they are fed into the model.

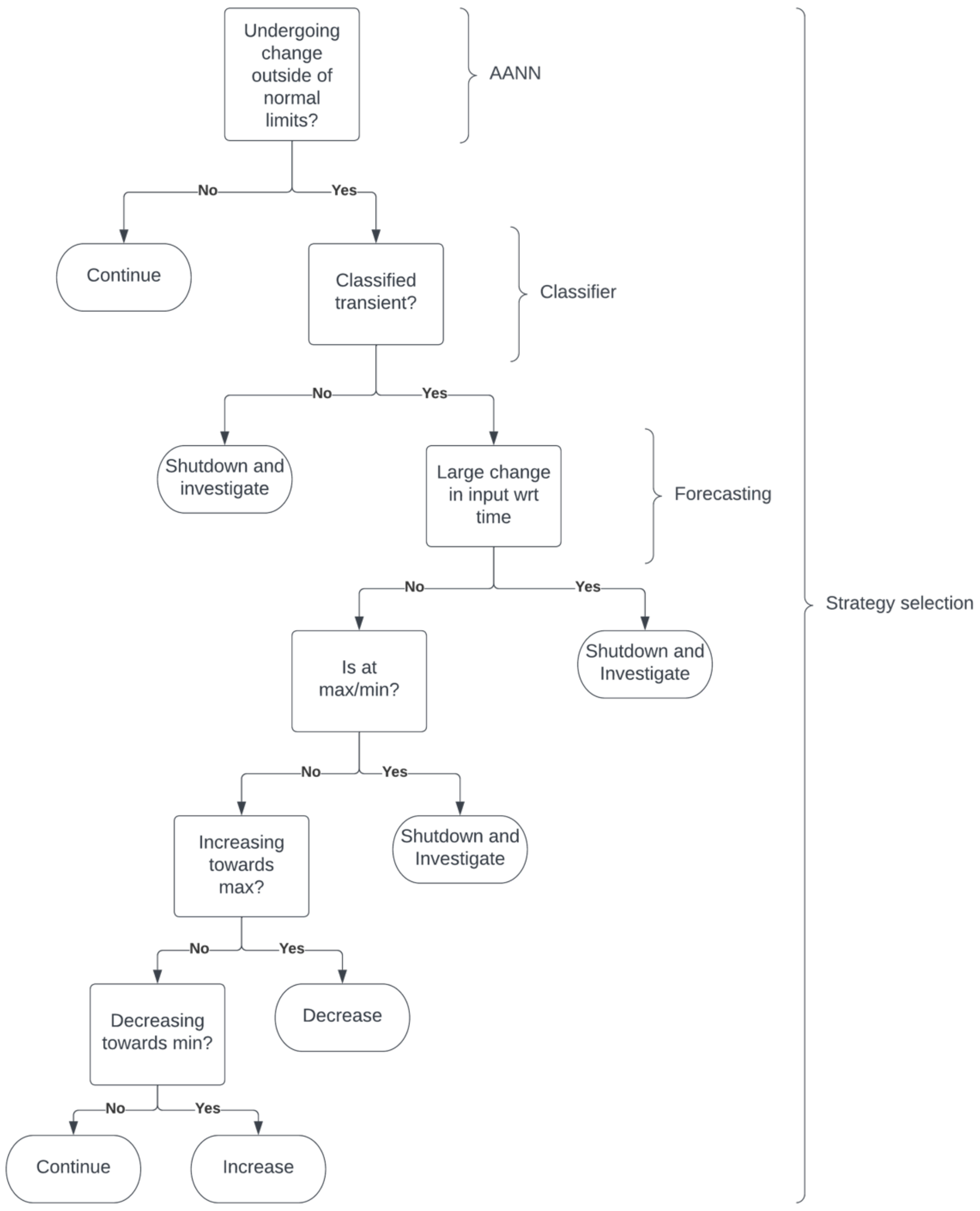
General Defensive Strategies

- **Maintain Offline Backups**: Ensure that ML model save files, including weights, hyperparameters, and training data, are securely stored offline, preventing unauthorized modifications.
- **Encrypt Model Data**: Apply rigorous encryption to model save files and parameters to secure them from potential intrusions.
- **Employ Multi-factor Authentication**: For systems managing training and real-time data, enforce strict access controls, including multi-factor authentication.
- **Engage in Regular Audits**: Implement a rigorous audit regimen, in conjunction with regulatory entities, ensuring that all defensive mechanisms are continuously operational and upgraded as per the evolving threat landscape.

In synthesizing these strategies, a multi-layered defense mechanism is orchestrated, ensuring that the ML models, integral to DT technologies and ACS implementations, are comprehensively shielded against the spectrum of cyber threats. Ensuring robust cybersecurity in this domain not only protects against immediate threats but also fortifies the infrastructure against evolving cyber challenges, thereby safeguarding critical systems in the realm of advanced reactor operation and beyond. Future work must persist in exploring, evaluating, and fortifying defenses against emergent cyber threats in ML and DT technologies, ensuring their secure and resilient operation in critical infrastructural applications.

Annex I: Full Size Images

Strategy selection decision tree plant-level Digital Twin



References

[1] R. T. Wood, B. R. Upadhyaya, D. C. Floyd, "An Autonomous Control Framework for Advanced Reactors," Nuclear Engineering and Technology, vol. 49, pp. 896–904, 2017.

[2] Department of Energy Office of Cyber Security, Energy Security, and Emergency Response, "Electric Disturbance Events Annual Summaries," 2023. [Online]. Available: https://www.oe.netl.doe.gov/OE417_annual_summary.aspx.

[3] G. Desarnaud, "Cyber Attacks and Energy Infrastructures," Études de l'Ifri, Institut français des relations internationales, 2017.

[4] M. Lehto, "Cyber-Attacks Against Critical Infrastructure," in Cyber Security, vol. 56, M. Lehto, P. Neittaanmäki, Eds. Cham: Springer International Publishing, 2022, pp. 3–42. doi: 10.1007/978-3-030-91293-2_1. [Online]. Available: https://link.springer.com/10.1007/978-3-030-91293-2_1.

[5] T. Wall, "Throwback Attack: German Nuclear Plant Cyberattack is a Wake-Up Call," 2023. [Online]. Available: https://www.industrialcybersecuritypulse.com/facilities/throwback-attack-german-nuclear-plant-cyberattack-is-a-wake-up-call/.

[6] R. Addison, "India Officials Confirm Cyberattack on Nuclear Power Plant Network," 2019. [Online]. Available: https://staging.industrialcyber.co/article/india-officials-confirm-cyber-attack-on-nuclear-power-plant-network/.

[7] C. M. Spirito et al., "Autonomous System Subversion Tactics: Prototypes and Recommended Countermeasures," Tech. Rep. INL/RPT-22-68871-Rev001, Idaho National Laboratory (INL), Idaho Falls, ID, USA, 2022. doi: 10.2172/1901802. [Online]. Available: https://www.osti.gov/biblio/1901802.

[8] M. J. Assante, R. M. Lee, "The Industrial Control System Cyber Kill Chain," SANS Institute, White Paper, 2021. [Online]. Available: https://sansorg.egnyte.com/dl/HHa9fCekmc.

[9] MITRE, "MITRE ATT&CK®," 2023. [Online]. Available: https://attack.mitre.org/.

[10] X. Chen, F. Zhang, "Development of a Hardware-in-the-Loop Testbed Using a Full-Scope Nuclear Power Plant Simulator for Instrumentation and Control and Cybersecurity Education, Training, and Research," in Innovative Use of Technology in Training: IV, Amelia Island, FL, 2023. [Online]. Available: https://www.ans.org/meetings/conte23/session/view-1567/.

[11] Western Services Corporation, "3KeyMaster Generic Pressurized Water Reactor Product Sheet," 2016.

[12] OPC Foundation, "OPC Data Access," 2017. [Online]. Available: https://opcfoundation.org/about/opc-technologies/opc-classic/.

[13] Allen-Bradley, "FactoryTalk Linx Gateway Software | FactoryTalk," 2023. [Online]. Available: https://www.rockwellautomation.com/en-us/products/software/factorytalk/operationsuite/communications/linx-gateway.html.

[14] ODVA Technologies, "Common Industrial Protocol (CIP™)," 2017. [Online]. Available: https://www.odva.org/technology-standards/key-technologies/common-industrial-protocol-cip/.

[15] Allen-Bradley, "Studio 5000 Logix Designer | FactoryTalk," 2022. [Online]. Available: https://www.rockwellautomation.com/en-us/products/software/factorytalk/designsuite/studio-5000/studio-5000-logix-designer.html.

[16] A. Skolnick et al., "Development of a Full Scope NPP Cybersecurity Hardware-in-the-Loop Testbed," in Cybersecurity for Nuclear Installations, Phoenix, AZ, 2022.

[17] Allen-Bradley, "1756-L81E ControlLogix," 2023. [Online]. Available: https://www.rockwellautomation.com/en-us/products/details.1756-L81E.html.

[18] D. Roeder, "dmroeder/pylogix: Read/Write Data from Allen Bradley Compact/Control Logix PLC's," 2022. [Online]. Available: https://github.com/dmroeder/pylogix/tree/master.

[19] M. Abadi et al., "TensorFlow: Large-Scale Machine Learning on Heterogeneous Distributed Systems," 2015.

[20] P. Yockey, K. Kelly, F. Zhang, "A Cyber Threat Methodology for Autonomous Control Systems for Advanced Reactors," in Cybersecurity for Nuclear Installations, Anaheim, CA, 2022.

[21] Oracle, "MySQL :: MySQL Documentation," 2023. [Online]. Available: https://dev.mysql.com/doc/.

[22] F. Chollet, "Keras," 2023.

[23] F. Pedregosa et al., "Scikit-learn: Machine Learning in Python," MACHINE LEARNING IN PYTHON, 2011.

[24] A. Moez, "PyCaret: An Open Source, Low-code Machine Learning Library in Python," 2020. [Online]. Available: https://pycaret.readthedocs.io/en/latest/index.html.

[25] D. R. Nair, M. G. Nair, T. Thakur, "A Smart Microgrid System with Artificial Intelligence for Power-Sharing and Power Quality Improvement," Energies, vol. 15, p. 5409, 2022.

[26] J. Roller, "MySQL Security Best Practices," 2022. [Online]. Available: https://www.computer.org/publications/tech-news/trends/mysql-security-best-practices/.

[27] Wireshark, "Documentation," 2023. [Online]. Available: [URL Needed].

[28] J. Steube, "hashcat - Advanced Password Recovery," 2023. [Online]. Available: https://hashcat.net/hashcat/.

[29] R. Automation, "Logix 5000 Controllers Data Access," 2022.

[30] OpenSSH Server, 2023. [Online]. Available: https://ubuntu.com/server/docs/service-openssh.

[31] Joblib Development Team, "Joblib: Running Python Functions as Pipeline Jobs," 2020. [Online]. Available: https://joblib.readthedocs.io/.

[32] Rapid7, "Metasploit," 2023. [Online]. Available: https://github.com/rapid7/metasploit-framework. Original date: *2011-08-30T06:13:20Z*

Page intentionally left blank.