CYBERSECURITY IN THE PUR-1 NUCLEAR REACTOR

by

Styliani Pantopoulou

A Thesis

Submitted to the Faculty of Purdue University

In Partial Fulfillment of the Requirements for the degree of

Master of Science in Nuclear Engineering

School of Nuclear Engineering

West Lafayette, Indiana

August 2021

THE PURDUE UNIVERSITY GRADUATE SCHOOL
STATEMENT OF COMMITTEE APPROVAL

Dr. Lefteri H. Tsoukalas, Chair

School of Nuclear Engineering

Dr. Chan K. Choi

School of Nuclear Engineering

Dr. Mary L. Comer

School of Electrical and Computer Engineering

Approved by:

Dr. Seungjin Kim

This is dedicated to the people I love.

ACKNOWLEDGMENTS

At first I would like to thank my thesis adviser, Professor Lefteri H. Tsoukalas, for offering me the opportunity to study at Purdue, as well as the member of my advisory committee, Professor Chan K. Choi, for his guidance and support. I also want to thank Professor Mary Comer for agreeing to be a member in my Master thesis committee. Without their help this thesis would not have been completed. During the second year of my Master's I was a visiting student in ANL, where I collaborated with Dr. Alexander Heifetz. I would like to thank Dr. Heifetz for his guidance and cooperation.

I would moreover like to thank and show my appreciation to the people of the Office of Naval Research (ONR) and MITRE, especially Dr. Michael Rivera and Matthew Mickelson for their help and comments during the research. This research has been partially supported by the ONR under Grant No N00014-18-1-2278.

Lastly, I would like to thank all my friends, for the good times we had during my stay in West Lafayette. Special thanks go to my family in Greece, for their love, continuous encouragement and moral support.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

ABSTRACT

Nuclear systems heavily depend on Instrumentation and Control (I&C) entities for their protection, monitoring and control processes, all of which play an important role for their safety and security. The obsolescence of analog I&C systems, along with the increased costs for their maintenance, has rendered the adoption of digital control systems inevitable. Digitization offers numerous advantages to systems, ranging from precision in measurements to reduction in equipment and costs. However, it also comes with a number of challenges, most of which are related to increased failure risk, either from human or control systems error, and vulnerability to attacks, which can be a major threat to non-proliferation. These characteristics point to the category of Cyber Physical Systems (CPSs), namely collections of computational components that receive physical inputs from sensors, and are connected to feedback loops in order to adapt to new circumstances. The ever growing use of CPSs may increase the risk for cyber attacks, that threaten a system's integrity and security. Plenty of research has been conducted on this topic. The focus of this work is to implement an architecture that can protect the system under review, namely Purdue University Reactor Number One (PUR-1), from these types of attacks. The reactor is physically modelled, through the use of point kinetics equations and reactivity calculations. Controllers existing in the plant are modelled and tuned for the purpose of controlling the reactor's power. Mitigation of the cyber attacks is later examined through fault tolerance. One of the main ways to achieve fault tolerance in systems of this type is through redundant components, the so-called replicas. Replicas are later used in a process of voting, in order to detect failures. According to the Byzantine Fault Tolerance (BFT) protocol, which is the most popular protocol for this purpose, a maximum number of t faults can be tolerated by the system, when there are in total $3t{+}1$ replicas in the system architecture. Redundancy, however, is not capable to keep a system safe by itself under all circumstances. For this purpose, software diversity is explored. According to this, software in the controllers gets diversified into distinct variants. Different software variants execute instructions, and other variants are expected to execute other actions. In the case where some tampered inputs crash (or deactivate) one of the variants, other variants take control and the system is tolerant

against failures. Lastly, CPS inertia is exploited along with rollback recovery methods for the rebooting of the system after a failure. The actual algorithm for the system studied in this work uses three redundant controllers and performs as follows; the error term from the subtraction of the output from the setpoint is fed as input to the first two controllers, as well as to the delay queue connected to the third controller. The outputs of the first two controllers are compared, and then there are two cases of operation. In the case of a good message in the input, the variants in the controllers do not crash, thus the signal from the top two controllers reaches the plant. In the case of a bad message, at least one of the two controllers crashes, because at least one of the code variants fails due to the diversity. This automatically triggers the comparator, which sends a signal so that the output of the isolated controller is used and propagates towards the plant. After implementing a Graphical User Interface (GUI), which acts as a simulator and visualizes the system's state, it is shown that PUR-1 is able to overcome bad messages regarding scram or control rod positions, when the protection architecture is activated. More specifically, when a bad message for scram is sent, the reactor manages to not drop its power level and continues to adjust the rod positions in order to achieve a specific power setpoint. Moreover, in the case of a bad message for the control rod positions, which means that the system is running open loop and thus is uncontrolled, the reactor manages to recover the rod positions and power level after some seconds. Conversely, when the protection system is deactivated, it is shown that bad messages regarding scram or rod positions are able to affect the reactor's state. In the case of the scram bad message, the reactor power drops immediately, while in the case of the rod position bad message, the power level changes uncontrollably.

1. INTRODUCTION

Nuclear systems depend heavily on Instrumentation and Control (I&C) entities for their protection, monitoring and control processes, all of which play an important role for their safety and security [1]. Although analog I&C systems have been widely utilized in the past, they have already started to become outdated. This obsolescence calls for increased Operation and Maintenance (O&M) costs, which ultimately prove to be ineffective with regards to overall costs and reliability. This consists one of the main reasons for updating or replacing analog legacy systems with digital or hybrid ones. Digitization has definitely brought many advantages to systems. For instance, digital technologies are able to customize certain facility functions, to control specific components, or to provide information about the system. Other benefits include, but are not limited to, precision in measurements, adaptability, reduction of equipment and cabling, ability to handle complex functions, reliability and availability, reduced costs, accurate monitoring, faster data processing, remote operation. Digital technologies are generally able to provide a wider spectrum of operation capabilities than analog ones.

However, there are a few challenges that need to be taken into account when embedding digital components into a system. First of all, the costs for the development of new systems can be increased. This can be partly attributed to the validation and verification (V&V) processes required after installing systems such as these. Moreover, the absence of suitable architectures in software or hardware components can result in increased risk of failures. The embedding of digital technologies in systems also comes with the need for training and involvement of the operations and maintenance personnel, although this does not consist a major issue. Another challenge is that in some cases, it may not be possible to replace all the components in the same time period. It should also be noted that analog and digital systems are not point-to-point identical, and this might require long procedures for adaptation of newer technologies to legacy equipment. This process can prove costly and highly time-consuming. Although digital I&C is software-based, the procedure of transitioning from analog to digital components is not as simple as modifying the software parts. Digital components can also introduce more failures, either from human or control systems error, and attack vectors for potential adversaries, which could project major threats to non-proliferation. This is associated with the demand for wireless remote access to systems and data. Various digital assets have to be categorized according to their level of sensitivity, with the purpose to apply security controls for protection of each one respectively. As was discussed here, systems can suffer from a plethora of challenges, ranging from compromised components to a variety of other attacks. The described complexity of digital I&C systems thus requires the development and implementation of methods to ensure safety and security. Table 1.1 lists the main advantages and disadvantages of digital I&C discussed in these paragraphs.

Table 1.1. The main advantages and disadvantages of digital I&C in systems.

Advantages	**Disadvantages**
precision in measurements	development and V&V cost
equipment and cabling reduction	lack of suitable architecture
reliability	failure risk
faster data processing	vulnerability against attacks
wireless remote operation	adaptation and training of personnel

The characteristics of the aforementioned systems classify them into the category of Cyber Physical Systems (CPSs), namely, systems which are comprised of both physical and computational components [2], that act in a dependent way, but also interact with each other. They utilize computational logic to monitor and control the dynamics of physical systems [3]. This means that computing, which is characterized by an inherent precision, meets with the uncertainty and imprecision of the physical world. CPSs can offer a wide variety of capabilities and ease the human-machine interaction through control and communication. In a nutshell, CPSs can be thought of as a collection of computers that control certain actuators, which receive sensor inputs. The combination of these creates a feedback control loop that is capable of adapting to the current system circumstances and providing efficient function to the CPS [4]. While these descriptions may seem abstract and not specific, there are certain architectures that have been proposed, in order to describe these systems. According to some work [5], the first step in developing a CPS is the ability to obtain reliable data from the system. This is a result of the quality of sensors and controllers used in the system, but also of the ability to hierarchize, classify, categorize and transfer the data. However, the data acquired from the system should be able to lead to useful information. The conversion from raw data to information can be achieved with algorithms, which bring consciousness to machines. The next step is the transfer of information from the system to other systems connected to it. This consists the cyber level of a CPS, where the performance of individual components can be compared to the rest of the parts in the system. Historical information can be further utilized in order to predict future states. Implementation of CPSs also comprises of proper decision making from the available information, through task prioritization and transferring of system knowledge to the users. Lastly, the stage of feedback from the computational to the physical space acts as a control level, so that the system obtains an adaptive and updating behavior. The discussed steps can be seen in summary in Figure 1.1. Throughout the years, relevant research has been focused on developing methods towards accomplishing system control, as well as on examining novel computational techniques or embedded software approaches; all with the ultimate goal of producing new supporting technologies for CPSs. Industrial CPSs have followed the procedure of decoupling the computational from the control aspects. Once the control system is implemented and tested, fine tuning helps to overcome any modeling faults. This can prove to be a challenging task, especially because the final objective is to keep the system functioning properly. All the information from the computational and physical components can be exploited efficiently towards the system's adaptation, robustness and reliability [6]. Apart from the mentioned aspects, CPSs behave according to laws of nature and are characterized by an inherited inertia [7]. This means that they need time to react to external stimulations. Inertia is of high importance to systems such as these, for the reason that they can continue operating even after they have been subjected to a design flaw or a fault. These properties are a step towards building safe systems, as they can provide improvements to traditional analog systems.

The nature of CPSs provides increased functionality and creates reliable and efficient systems. However, it can introduce a variety of vulnerabilities. These vulnerabilities are a result of a lot of factors, including but not limited to, the nature of the system's digital

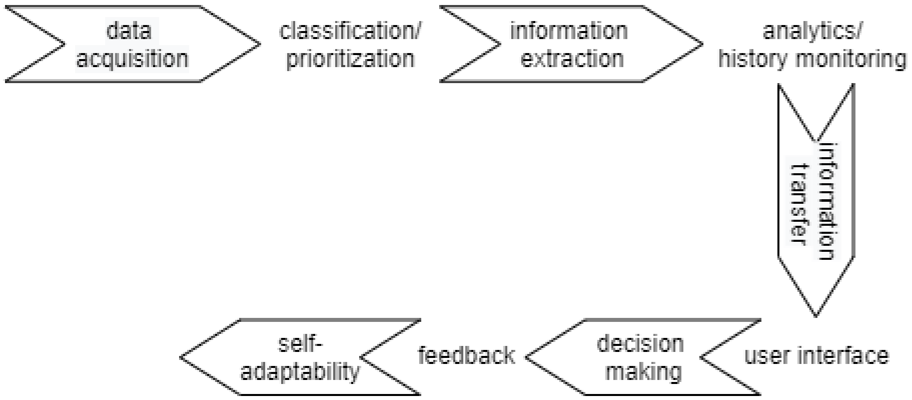


Figure 1.1. Steps for the implementation of a cyber physical system.

assets and the ever-increasing complexity of malicious intruders [8]. Generally, cyber attacks have a negative impact on confidentiality, integrity and availability, which are the three major objectives of the concept of "security" [9]. Confidentiality is related to the access of system assets only by authorized entities. However, it is considered hard to establish and even harder to maintain. Integrity pertains to the prevention of a system's alteration by an unauthorized party. This ensures that data and processes remain accurate, unmodified, and complete. Lastly, availability relates to the uninterrupted access to a system by its users; it should be noted that threats against availability are sometimes non-malicious instances. During the past years, the research community has studied a variety of methods with the purpose to achieve one or more of these security objectives. It has been argued that general control systems should be treated in a different way compared to CPSs [10]. It is considered crucial to detect an attack, as well as to prepare the system to respond towards that. These can be achieved through close monitoring of the system or even proper operators' training. The attacks that a CPS can face vary across a wide range of possibilities, such as physical exploitation, state estimation, introduction of tampered data, or connectivity to the internet [11]. One major example that can depict the severity of attacks against a system is definitely the cyber-attack on the Ukrainian power grid in 2015 [12]. In that case, the intruders were able to monitor the system for a long period of time. As a result, they got advantage of the system's vulnerabilities and executed the attack. Major weaknesses of that system were the large amount of information stored online, and the absence of reliable security protocols.

The two basic hijacking techniques used by the attackers, according to [12] are depicted in Figure 1.2. These included remote Supervisory Control and Data Acquisition (SCADA) software, which enabled the attackers to gain access to the information technology (IT) networks of electricity companies, as well as remote tools, that granted them access to the human-machine interface (HMI). The physical nature of CPSs can be one of the ways to deal with attacks [13]. Since physical laws are generally followed during system operation, one can discover possible attacks by monitoring the system's behavior. This can also be achieved by deploying prediction mechanisms. Other methods are focused on a system's architecture. For example, they can be realized by imitating possible threats or even by intentionally "attacking" the system [14]. Such approaches tend to be successful to some extent, by establishing certain points in the system where security can be checked. However, it is not guaranteed that a specific methodology of this kind will be efficient in every type of system without some additional modifications. Other developed methods focus on a system's modeling and architecture, in order to achieve security. This is achieved by using variations of a system's modeling [15], [16], by taking into consideration the system architecture [17], by introducing patterns [18] or by using specific agents (software) that make decisions upon triggering from the system's environment.

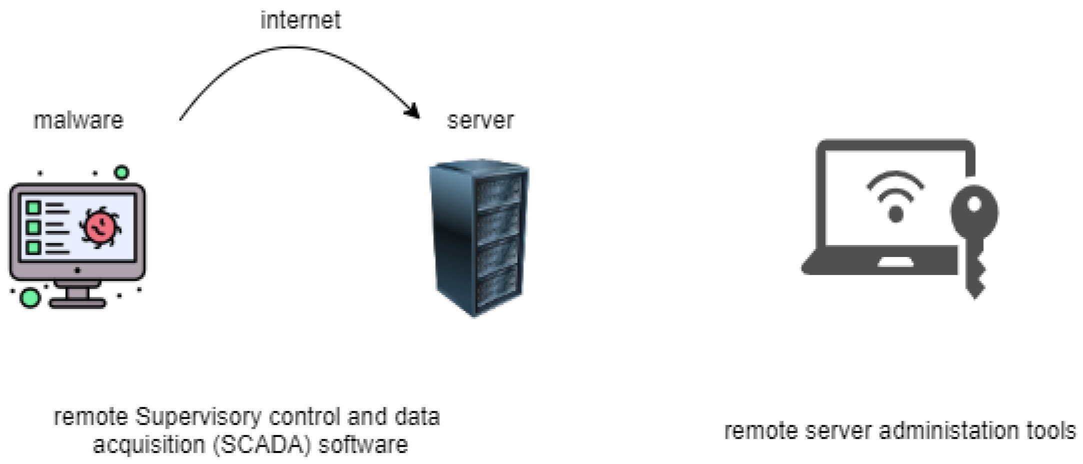

internet
malware
server
remote Supervisory control and data acquisition (SCADA) software
remote server administation tools

Figure 1.2. SCADA hijack approaches used during the cyber attack on the Ukrainian power grid.

A lot of attention should thus be paid to the availability and recovery of systems after a failure, as the aspect of fault tolerance is of major importance. Fault tolerance in a system includes detection of an intruder, recovery after the error is presented, as well as any additional actions in order to successfully mitigate the attack. There is a number of already studied techniques towards achieving fault tolerance. Many systems actually rely on component redundancy. Researchers have shown that utilizing the system's redundancy can ensure defense against attacks. This can be translated to redundancy in the transmitted information, in case some of it is tampered with. That is accomplished through error correcting codes in bits, or replicas [19]. Another approach is through redundancy in the time permitted to fulfill a process, or redundancy in the number of trials for the execution of specific computations. Finally, a type of physical redundancy can be accomplished through multiple identical system components (replicas). The redundant replicas account for some of them that will malfunction during a failure, through the concept of consensus or voting between them. A different feature of CPSs that can prove useful for achieving fault tolerance – and consequently cybersecurity – is the concept of diversity. It has been observed that a great amount of vulnerabilities in systems emerge from faults in the software part. The idea of diversity in systems is based on changing characteristics of the software. This way, any possible attackers have to face and get past the complex gates of a system [20]. Diversity also helps systems to appear and behave in distinct ways, depending on the inputs they are presented [21]. This can be achieved, for example, with network protocols' diversity or with binary transformations [22].

Nuclear reactors can also be considered as CPSs. The I&C systems of a nuclear reactor are comprised of several control systems that work together towards achieving the control of its vital processes. The trend to replace analog components of existing plants with digital ones is crucial towards the system's protection, monitoring, and control. However, there are issues arising from a reactor's 'modernization' process. For example, there is a chance that the design basis of the reactor has to be reestablished, that the requirements have to be modified, or that it needs to become more fortified against adversarial attacks. Figure 1.3 is an abstract representation of the protection layers of a nuclear reactor against threats. Since the actual reactor needs to be protected, an abundance of systems and instances that surround it need to create a protective layer around it. This ranges from operators' suitable training in order to take actions in case of a threat, to fortification of digital I&C systems so that they can detect and/or mitigate attacks. Certain compromises may have to be made in order for the system to be able to achieve reliability, availability, and security. The techniques described in the previous paragraphs can be thought of as only a part of the whole strategy towards achieving security, as they can be combined with control methods. For example, researchers have proposed fuzzy control systems, which connect the reactor parameters, in order to create a Proportional – Integral – Derivative (PID) controller for the plant [23]. Other studies have designed state space models that represent the system, which later were used to uncover certain relationships between the reactor's variables [24]. Even statistical methods have proved to be useful for controlling and monitoring, specifically through transient states [25].

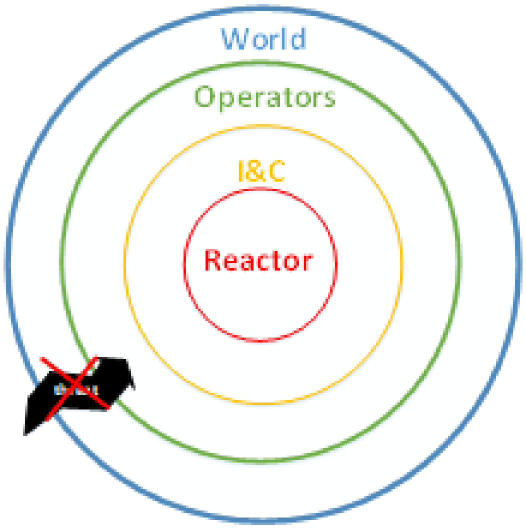


Figure 1.3. Layers for protection of a cyber physical system, such as a nuclear power plant.

This thesis explores the establishment of cybersecurity in a nuclear reactor, specifically in the Purdue University Reactor Number One (PUR-1), which comprises the first nuclear reactor in the United States with all-digital control systems. In order to develop a method for securing the reactor, it has to be modeled physically. This means that the governing equations of nuclear power plants (NPPs) are used. The control schemes are identified, in combination with the control variables that describe physical processes. Moreover, a protection architecture is implemented, that secures the system against cyber threats. This architecture is based on the Byzantine Fault Tolerance (BFT) algorithm, and comprises a modified model of already existing protection algorithms of this kind. Finally, the system is tested under a variety of circumstances and attacks, in order to show the effects of the protection infrastructure. This is done through the implementation of a graphical simulator, for the simple reason that, testing a reactor with a variety of attacks scenarios can lead to an extremely dangerous environment and create problems with licensing procedures. This thesis is organized as follows; Chapter 2 refers to the physics modeling of the reactor. Chapter 3 pertains to the detection and mitigation of cyber attacks in systems through fault tolerance and artificial diversity, and gives a description of the algorithm followed for the cybersecurity of the reactor. Chapter 4 presents the obtained results, and finally Chapter 5 concludes the thesis, while providing suggestions for future work on this matter.

2. REACTOR PHYSICS MODELING

2.1 Facility

The facility under review in this work is the pool-type Purdue University Reactor Number One (PUR-1) [26], which was built by Lockheed Nuclear Corporation in 1962. It is a nuclear reactor that runs at 1 kWth of power, but has been licensed for a level up to 10 kWth. Its core has a volume of about 0.06 cubic meters, and it is located on the bottom of a pool. The pool is about 5.2 meters deep, has a diameter of 2.4 meters, and is surrounded by 45 centimeters of concrete. The core is fueled by 190, 19.75% enriched, U_2Si_3-Al fuel plates clad in Al-6061. Sixteen fuel assemblies create the active core region, and each of the assemblies contains up to fourteen fuel plates. Also, they are surrounded by twenty reflector assemblies made of graphite. The reactor is filled with light water, which helps for neutron moderation and cooling at the same time. The reactor power is controlled by two borated stainless steel control rods, called SS1 and SS2, and one air-filled regulating rod, called RR. The core and pool can be observed in Figure 2.1.

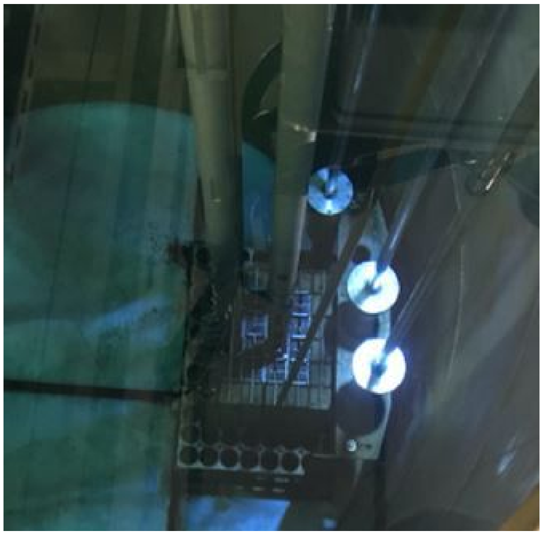


Figure 2.1. Core and coolant pool of the PUR-1 reactor.

The reactor is comprised of a variety of sensors that monitor several parameters, such as neutron flux, radiation level, air pressure and water chemistry. Signals coming from sensors can be evaluated by a Programmable Logic Controller (PLC). This controller has a time resolution of about ten milliseconds (ms), which means it can process signals every ten milliseconds. Measurements by the sensors are usually communicated as 4 – 20 milliamperes (mA) or 0 – 5 Volts (V) signals from the facility sensors to the central control. However, the signals are later digitized at the central control, in order to be able to get processed by the controllers and the rest of the system. Generally, when the reactor operates in steady state, small deviations may occur as noise. However, these stay below the threshold of the alarm in the reactor. As a change is present in the system, the values of measured quantities also change. A slow rate of change indicates a natural and expected operation. This is because the PUR-1 is a physical system. That mentioned, it is characterized by laws of nature that control its operational variables. As all physical systems, this too is defined by inertia, which means that any slow changes are completely normal. Something that would indicate nefarious operations would be a fast change in some of the crucial reactor parameters.

2.2 Point Kinetics Equations

The physics modeling of any system begins by the mathematical representation of its state for every operation cycle. In a nuclear power plant, this is performed by using the Point Kinetics Equations (PKEs) [24], that describe how various parameters of the reactor change with time. The simplest form of these equations is as follows:

$$\frac{dn(t)}{dt} = \frac{\rho(t) - \beta}{\Lambda} \cdot n(t) + \lambda \cdot c(t) \tag{2.1}$$

$$\frac{dc(t)}{dt} = \frac{\beta}{\Lambda} \cdot n(t) - \lambda \cdot c(t) \tag{2.2}$$

In these equations, n represents the neutron density in the reactor, c is the delayed neutron precursor density, ρ is the reactivity. Moreover, β is the delayed neutron fraction, Λ is the mean neutron lifetime in the reactor core, and λ is the mean neutron precursor lifetime. These equations can help towards designing a control scheme for the reactor.

If we have knowledge of the operation cycle duration $\mathrm{d}t$, the point kinetics equations can be solved as:

$$n(t+1) = n(t) + (\frac{\rho(t) - \beta}{\Lambda} \cdot n(t) + \lambda \cdot c(t)) \cdot \mathrm{d}t \tag{2.3}$$

$$c(t+1) = c(t) + (\frac{\beta}{\Lambda} \cdot n(t) - \lambda \cdot c(t)) \cdot \mathrm{d}t \tag{2.4}$$

The main measurable quantity in a reactor is the neutron flux Φ. This represents the number of neutrons going through a square centimeter per second. This quantity is connected with neutron density with the following formula:

$$\Phi = n \cdot \bar{\nu} \tag{2.5}$$

where $\bar{\nu}$ denotes the average neutron speed, which is constant, and n is the neutron density. Thus, knowing the neutron flux value in a nuclear power plant is equivalent to knowing the neutron density. The reactor power can then be easily calculated, as it is proportional to the neutron density:

$$P = V_r \cdot V_{fuel} \cdot n \cdot \bar{\nu} \cdot \Sigma_f \cdot E_f \tag{2.6}$$

In the above equation, V_r is the volume of the reactor, V_{fuel} is the percent amount of fuel volume (compared to the coolant volume), n is the neutron density, $\bar{\nu}$ represents the mean neutron velocity in the reactor, Σ_f is the macroscopic fission cross section, and E_f is the energy released per fission event.

2.3 Reactivity Calculations

Reactivity is a reactor parameter that defines how far from unity is the value of the effective multiplication factor k_{eff}. This factor is defined as the ratio of neutrons in the current generation to the number of neutrons of the preceding generation or as follows:

$$k_{eff} = \frac{number \quad of \quad produced \quad neutrons}{number \quad of \quad absorbed \quad neutrons + number \quad of \quad leaked \quad neutrons} \tag{2.7}$$

When this factor reaches unity, the reaction is called critical, which means that we have a self-sustaining fission chain reaction. Values smaller than unity render the reactor subcritical, while values of k_{eff} greater than unity render the reactor supercritical. Reactivity is calculated as:

$$\rho = \frac{k_{eff} - 1}{k_{eff}} = \frac{\Delta k}{k} \tag{2.8}$$

The factor k_{eff} (and subsequently the reactivity ρ) depends on the heights of the reactor control rods. Control rods are inserted into or withdrawn from the reactor core with the ultimate purpose of controlling the reactor's output power. The distance to which they are withdrawn affects k_{eff}. When the rods are fully inserted into the core, k_{eff} becomes very small; as a result, the reactor slows down and stops. When the rods are fully withdrawn, the reactor power rises quickly. The usual goal is to maintain the power level of the reactor constant. This means that the rods have to be kept in heights such that k_{eff} is close to unity.

In order to establish a relationship for the reactivity, we measure k_{eff} according to the rod heights and measure the reactivity from the above formula. The results are plotted below, in Figure 2.2. In the legend beside the graph, rod1 refers to rod SS1, rod2 refers to SS2, and regrod refers to RR, as these were discussed earlier.

In order to automate the procedure for updating the reactivity value for each rod height value, two methods are explored. First, linear interpolation is examined. Since we have measurements for specific points (shown in figure), the reactivity for all the in-between

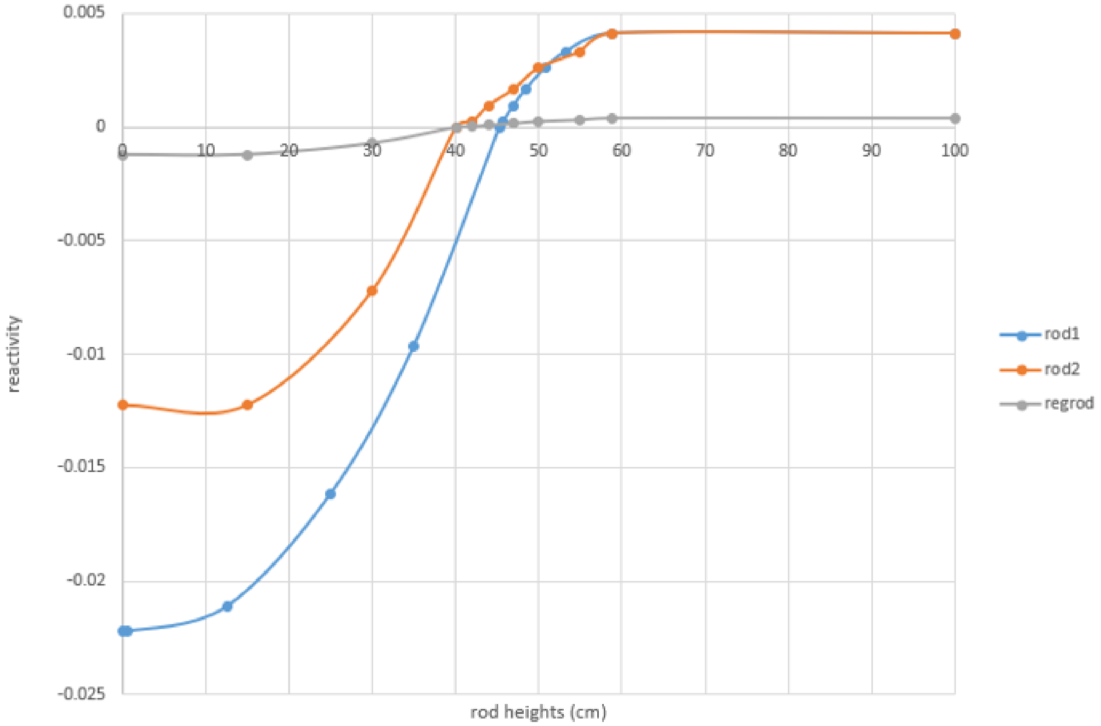


Figure 2.2. Reactivity as a function of rod heights (rod1 refers to rod SS1, rod2 refers to SS2, and regrod refers to RR) in the PUR-1 reactor.

values has to be calculated. If there is a point with coordinates (x_0, y_0) and a point with coordinates (x_1, y_1), and we need to find the coordinates of a point (x, y) in between, this is performed as follows.

$$k_{eff} = \frac{y - y_0}{x - x_0} = \frac{y_1 - y_0}{x_1 - x_0} \tag{2.9}$$

This is a result of the fact that we assume a straight line going through these three points.

The second method involves the fitting of polynomial lines in the measurements. This is done by fitting fourth degree polynomials to the reactivity curves. This degree is chosen because it is shown to have better fit on the measured curves. Specifically, shown in Figure 2.3 are the integral control rod worth fitted curves.

The three integral control rod worth curves are found to be the following, for rods SS1, SS2, and RR respectively. In these equations, h_1, h_2, and h_3 pertain to the heights of each

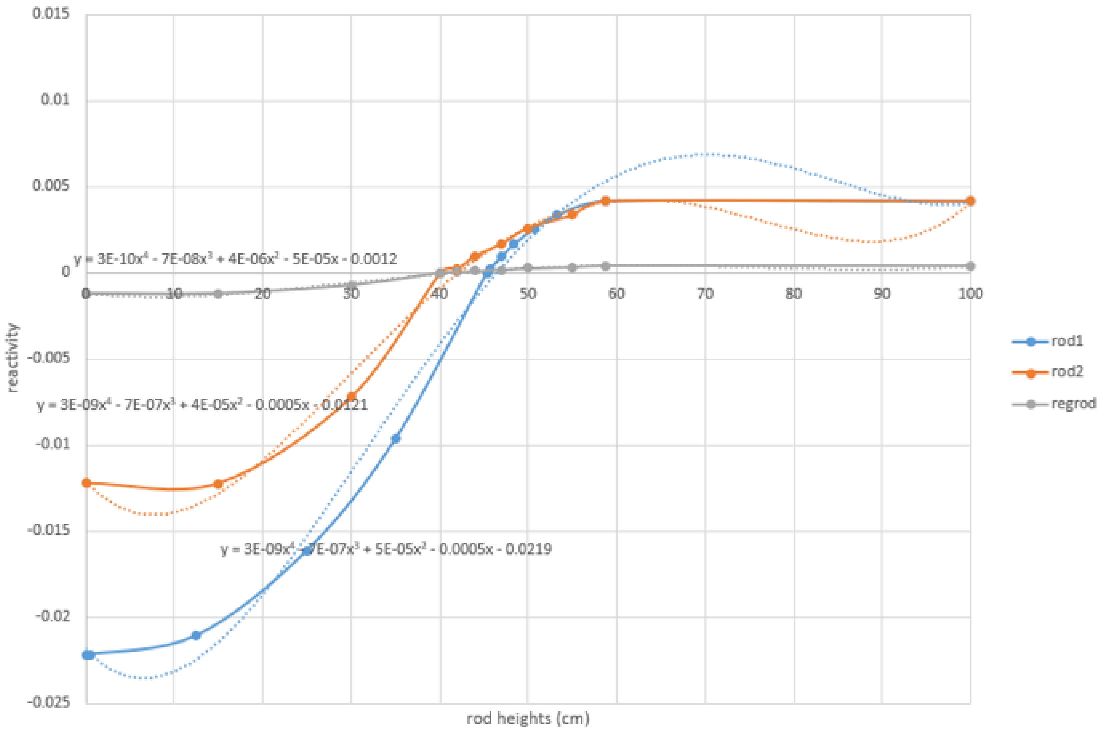


Figure 2.3. Curve fitting with fourth degree polynomials. Rod1 refers to rod SS1, rod2 refers to SS2, and regrod refers to RR.

rod, and ρ_1, ρ_2, ρ_3 are the reactivities produced from each rod, which are then added to get the total reactivity:

$$\rho_1 = 3 \cdot 10^{-9} \cdot h_1^4 - 7 \cdot 10^{-7} \cdot h_1^3 + 5 \cdot 10^{-5} \cdot h_1^2 - 5 \cdot 10^{-4} \cdot h_1 - 0.0219 \tag{2.10}$$

$$\rho_2 = 3 \cdot 10^{-9} \cdot h_2^4 - 7 \cdot 10^{-7} \cdot h_2^3 + 4 \cdot 10^{-5} \cdot h_2^2 - 5 \cdot 10^{-4} \cdot h_2 - 0.0121 \tag{2.11}$$

$$\rho_3 = 3 \cdot 10^{-10} \cdot h_3^4 - 7 \cdot 10^{-8} \cdot h_3^3 + 4 \cdot 10^{-6} \cdot h_3^2 - 5 \cdot 10^{-5} \cdot h_3 - 0.0012 \tag{2.12}$$

The rods in the reactor move with time according to the following equations. For rods SS1 and SS2 we have:

$$h = \begin{cases} h_0 & t \leq 0.1sec \\ \frac{1}{2} \cdot g \cdot t^2 & t \geq 0.1sec \end{cases} \tag{2.13}$$

where g is the acceleration of gravity, and h_0 the initial height of the rod. For rod RR it holds that:

$$h_{RR} = v \cdot t \tag{2.14}$$

where v is the speed of the rod.

2.4 Controllers

Controllers are basically used to control a system's state, or in other words bring the system's output in a steady state. The most universal and most used controllers in various industry applications are the Proportional – Integral – Derivative controllers, or simply PID. The basic equation for a controller such as this is the following:

$$u(t) = k_p \cdot \mathrm{e}(t) + k_\mathrm{i} \cdot \int \mathrm{e}(t)\,\mathrm{d}t + k_d \cdot \frac{d\mathrm{e}}{\mathrm{d}t} \tag{2.15}$$

where $u(t)$ is the variable of a system that needs to be controlled, k_p is the proportional controller gain, k_i is the integral gain, k_d is the derivative gain, and $\mathrm{e}(t)$ is the error term that is a result of subtracting the system output $y(t)$ from a specific setpoint $r(t)$, as seen in the following equation. As can be seen from the equation above, k_p gets multiplied to the error of the same cycle, k_i gets multiplied to the sum or history of errors, while k_d gets multiplied to the differential error, which is basically the derivative of the error.

$$\mathrm{e}(t) = r(t) - y(t) \tag{2.16}$$

The usual connection between a system and a PID controller is as shown in Figure 2.4.

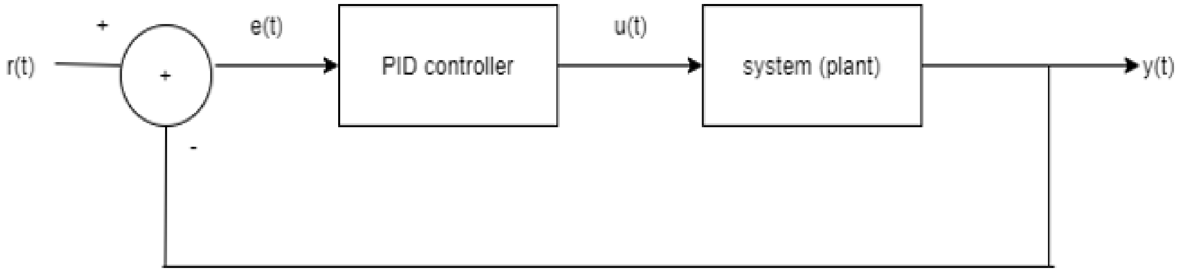


Figure 2.4. Connection of a Proportional - Integral - Derivative (PID) controller to a system.

If we try to express the transfer function of the PID controller in the Laplace Transform space, instead of the time space, we would get the following:

$$G_{PID}(s) = k_p + k_\mathrm{i} \cdot \frac{1}{s} + k_d \cdot s \tag{2.17}$$

which could be expressed also as:

$$G_{PID}(s) = k_p \cdot (1 + \frac{1}{T_\mathrm{i} \cdot s} + T_d \cdot s) \tag{2.18}$$

where the parameters T_i and T_d are other expressions for the controller parameters, namely:

$$k_\mathrm{i} = \frac{k_p}{T_\mathrm{i}} \tag{2.19}$$

$$k_d = k_p \cdot T_d \tag{2.20}$$

Expressing the function with these parameters is helpful for what will be described later. Generally, each of the three parameters has its own role towards defining the system output. More specifically, the proportional gain k_p has to do with the rise time of the system, namely how quick is the rise of the system's output towards reaching the desired value. The integral gain, k_i, is used to reduce the steady state error. The system might not be able to completely nullify the error from the desired setpoint in the steady state, but at least this error should be minimal. Lastly, the parameter of differential gain, k_d is used in order to reduce the settling time of the system, namely the time needed for the system to reach steady state. Moreover, this parameter helps reduce the overshoot of a system. In other words, it can reduce oscillations of the system until it reaches the setpoint. Figure 2.5 below shows what happens to a system when only the proportional gain k_p is used, if we assume that we set the setpoint to a value of 100. As can be seen, the system experiences oscillations of huge magnitude and never settles into the steady state. This type of controller is known as Proportional or simply P controller.

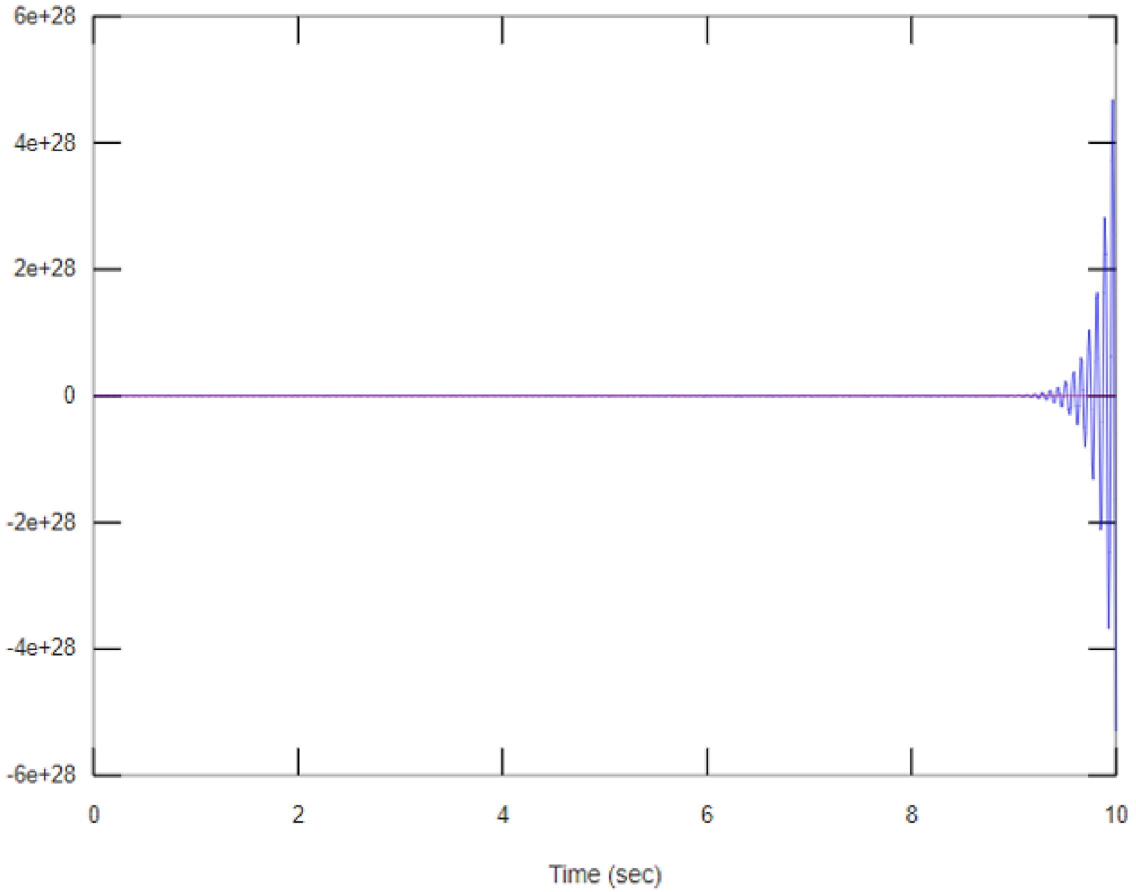


Figure 2.5. The effect of proportional control on an unstable system.

In Figure 2.6, it can be seen how adding the integral controller gain k_i changes the system behavior. The system manages to not oscillate to very large amplitudes, as well as to settle nearly close to the setpoint after some time. This type of controller is known as Proportional – Integral or simply PI.

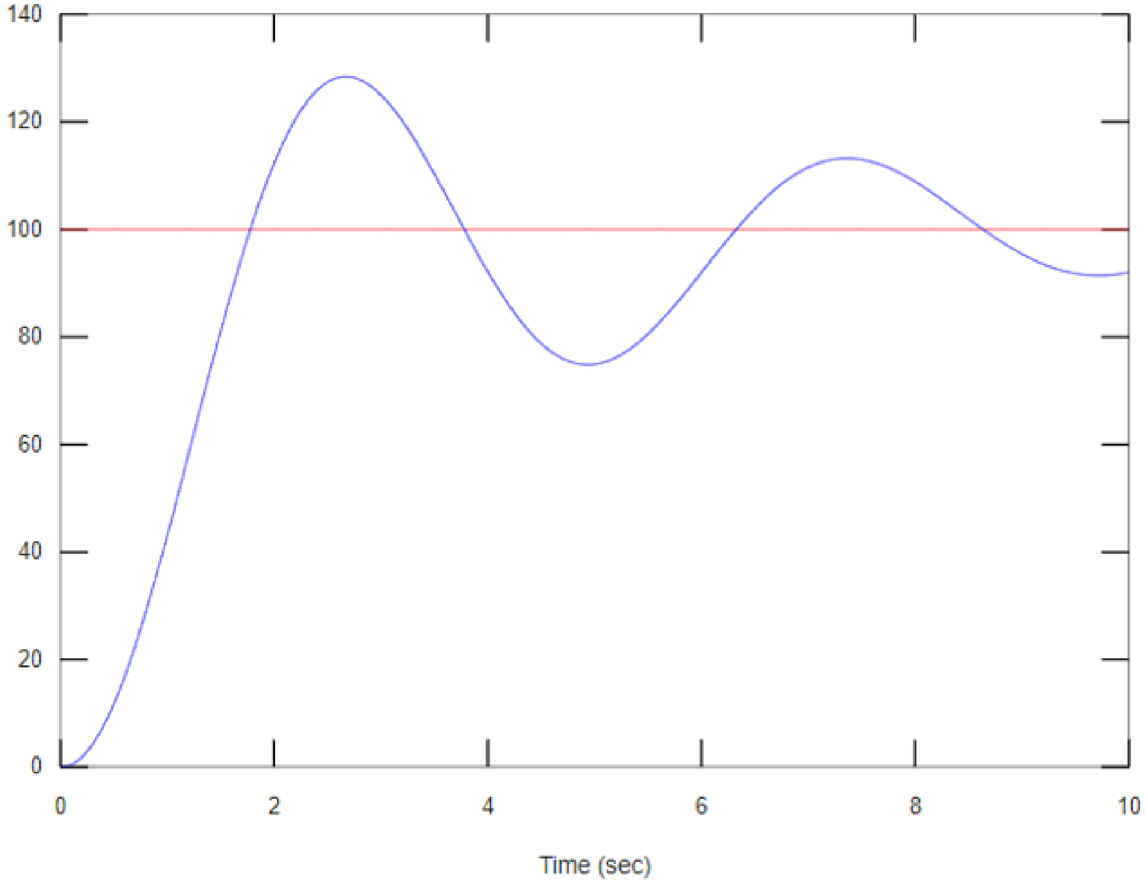


Figure 2.6. The effect of proportional-integral control on an unstable system.

Finally, when the derivative gain k_d is also used, the system manages to avoid a lot of oscillations and sets to the steady state value rather quickly, as seen in Figure 2.7. This type of controller is known as Proportional – Integral – Derivative or simply PID.

At this point, the question of how these controller gains are defined arises. There are many methods which are used to tackle this issue. One of the most common is the Ziegler-Nichols method. Even for this method, there are a lot of approaches. One of them is the following; first, a proportional gain called k_{cr} (critical) is used. This is a parameter that should make the system oscillate around the steady state with stable magnitude and period

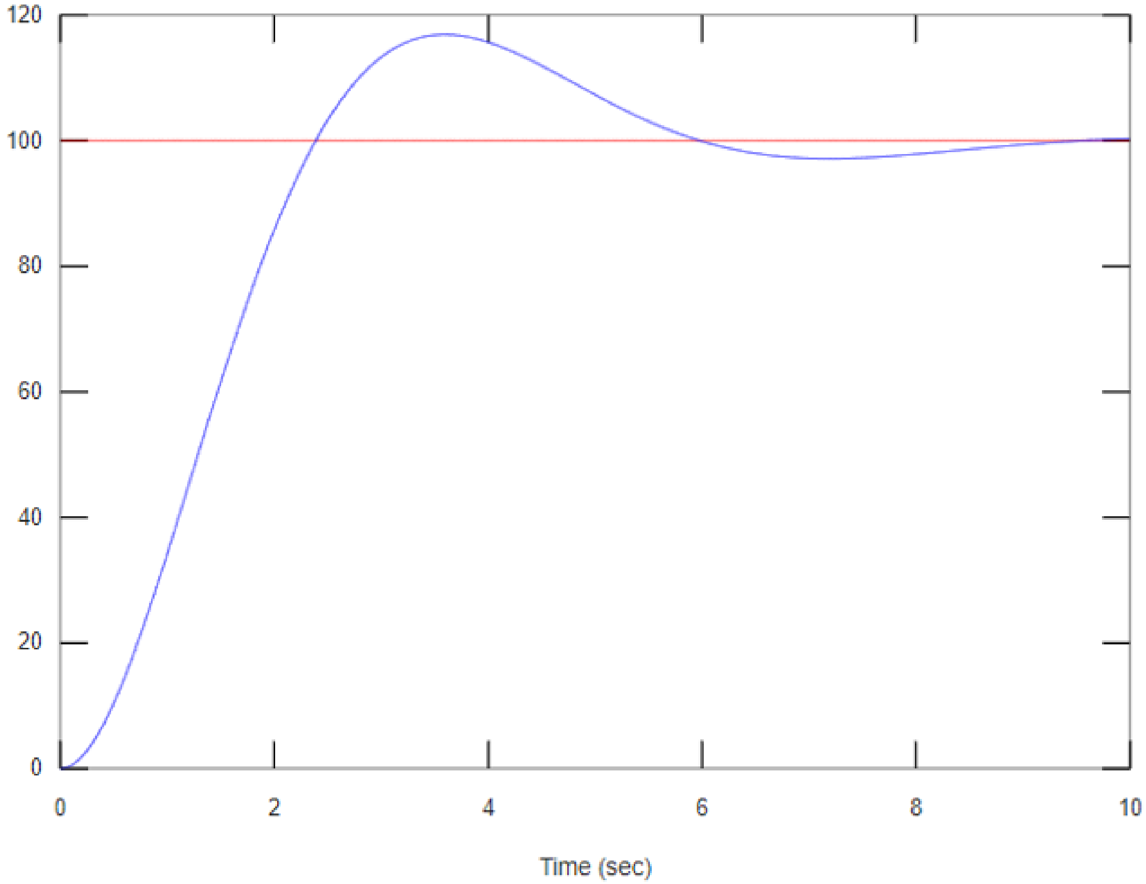


Figure 2.7. The effect of proportional-integral-derivative control on an unstable system.

T_{cr} (measured in seconds), like in the Figure 2.8. Then, the PID controller gains can be defined as follows:

$$k_p = 0.6 \cdot k_{cr} \tag{2.21}$$

$$k_\mathrm{i} = 0.5 \cdot T_{cr} \tag{2.22}$$

$$k_d = 0.125 \cdot T_{cr} \tag{2.23}$$

In the system under review in this work, the control rod heights are considered as the system's input, while the neutron density in the reactor (and subsequently the power) is

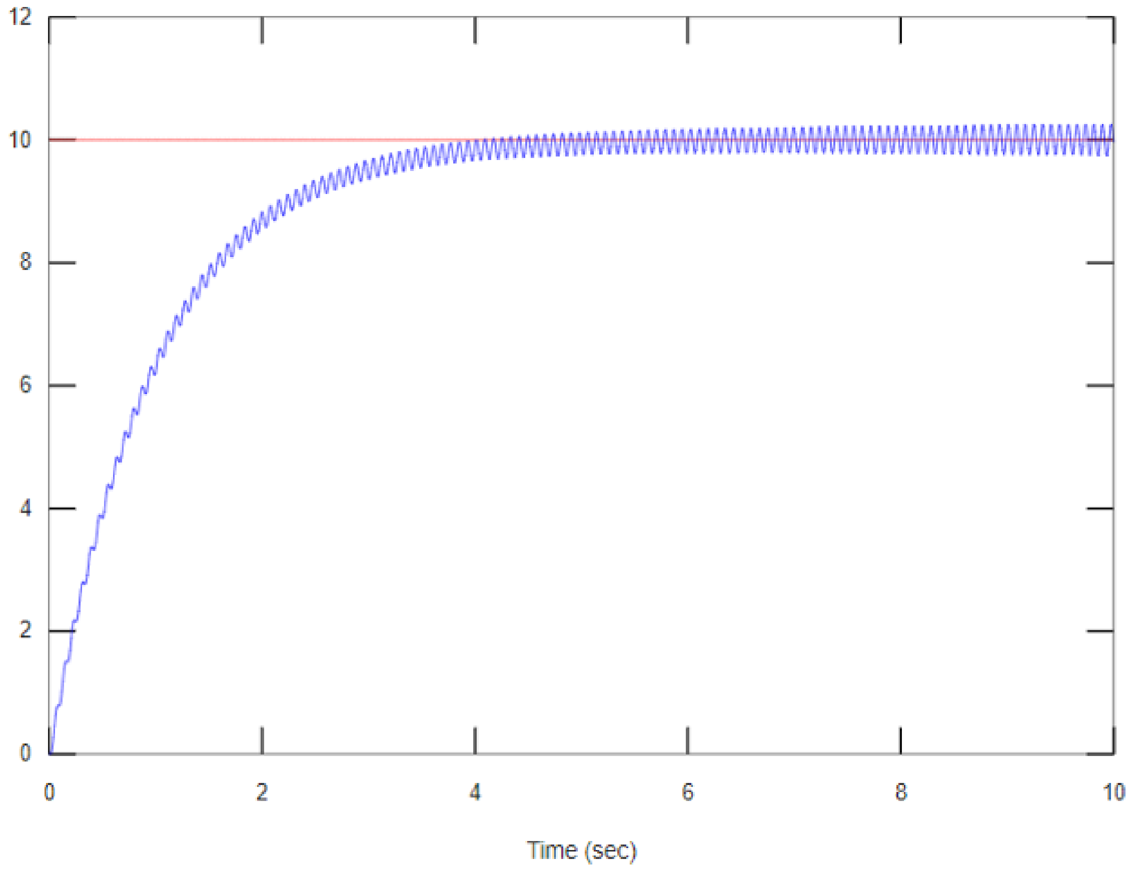


Figure 2.8. System response to be achieved as a part of the Ziegler-Nichols method.

considered as the system's output. This means that the rod heights need to be controlled, in order to achieve a specific reactor power. Figure 2.9 depicts the input-output relationship, in accordance with the plant and the controller.

2.5 Sensitivity Analysis

In order to better understand the relationship between various reactor parameters, a procedure called sensitivity analysis can be used. This analysis helps define the relationship between the inputs and outputs of a system, and lets one understand which variables are most crucial towards the system output. For the system studied here, it is needed to examine the relationship between the system's inputs and outputs. In other words, to find out how control rod heights (and subsequently reactivity) and delayed neutron precursor densities are

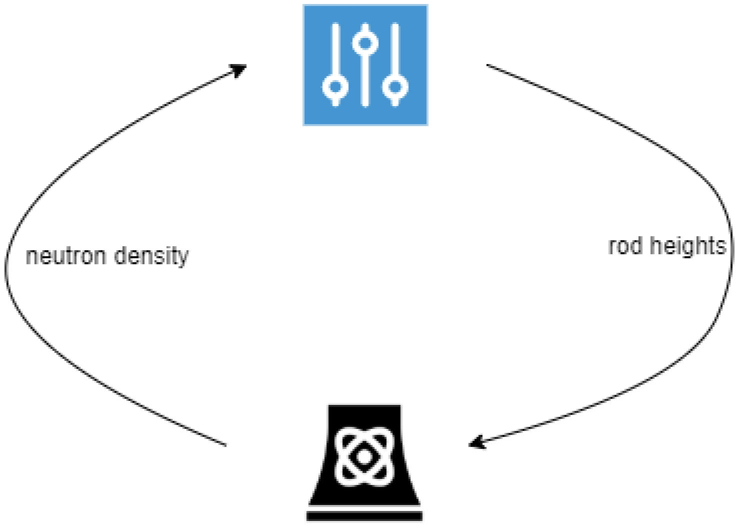


Figure 2.9. Relationship between control variables (rod heights) and output variables (neutron density) in the studied system.

connected with neutron density in the reactor. Knowing the neutron density at a specific time instant means that we obtain knowledge of the reactor power level at that point. More specifically, when performing sensitivity analysis on the variables, we are interested in quantifying the percent change in the outputs when one input parameter is perturbed. This is done with the help of the derivative. For example, for a system with an input named x_i and an output named y_j, one can calculate the sensitivity of the output with respect to the input as follows:

$$sensitivity = \frac{\partial y_\mathrm{j}}{\partial x_\mathrm{i}} \tag{2.24}$$

In Table 2.1, the values in the two-dimensional sensitivity matrix represent neutron density values n (measured in neutrons/cm^3), when rod heights (or subsequently reactivity ρ) and delayed neutron precursor density values c (measured in neutrons/cm^3) are perturbed. For example, the first line represents neutron density for c=900, and reactivity values of ρ=-4, ρ=-3, ρ=-0.2, ρ=-0.1 and ρ=-0.01. The second line represents neutron density for c=950, and reactivity values of ρ=-4, ρ=-3, ρ=-0.2, ρ=-0.1 and ρ=-0.01. Similar for the next lines. If seen from column perspective, the first column represents neutron density for ρ=-4, and delayed neutron precursor density values of c=900, c=950, c=1000, c=1050, c=1100, c=1150 and c=1200. Similar for the next columns. It can be observed that the variation of reactivity while the delayed neutron precursor density is stable has a greater impact on neutron density, than for the case when delayed neutron precursor density is varied for a stable reactivity value. Thus, it can be argued that reactivity plays a more important role towards defining the output value rather than delayed neutron precursor density.

Table 2.1. Sensitivity analysis for the reactor inputs and outputs.

parameters	ρ=-4	ρ=-3	ρ=-0.2	ρ=-0.1	ρ=-0.01
c=900 cm^{-3}	74.25869	99.25869	169.2587	171.7587	174.0087
c=950 cm^{-3}	74.26619	99.26619	169.2662	171.7662	174.0162
c=1000 cm^{-3}	74.27369	99.27369	169.2737	171.7737	174.0237
c=1050 cm^{-3}	74.28119	99.28119	169.2812	171.7812	174.0312
c=1100 cm^{-3}	74.28869	99.28869	169.2887	171.7887	174.0387
c=1150 cm^{-3}	74.29619	99.29619	169.2962	171.7962	174.0462
c=1200 cm^{-3}	74.30369	99.30369	169.3037	171.8037	174.0537

3. MITIGATION OF CYBER ATTACKS

3.1 Fault Tolerance

It is known that systems such as a nuclear reactor are considered to belong to the wider category of distributed systems, namely systems that are comprised of various separate and independent software and hardware components, which work together towards a specific goal. Distributed systems are different from the also well-known parallel systems, in the sense that parallel systems use multiple components that execute tasks in the same time, while distributed systems divide a task between the multiple components, all of which have a common purpose. It is worth mentioning here that these components communicate through network messages. One of the major advantages of distributed systems is their high fault tolerance. In a system, this pertains to detection and elimination of an attack, as well as recovery of the system afterwards. A crucial part towards achieving fault tolerance is the identification of the fault that caused the system to fail. This way, the system will be able to operate even after the detection of a fault. Faults can be classified according to their duration and extent [27]–[29]. There are cases of transient faults, which usually happen for short periods of time, intermittent faults, which have a periodic behavior and happen when a system is in an unstable state, and lastly permanent faults, which are a result of damage in the system or errors in the design phase of a system. Moreover, faults can affect a single or multiple components. A common concept used for managing failures like these is the introduction of redundancy in systems. Redundancy can occur in software or hardware components. This refers to redundancy in transmitted data, to minimize the chances of tampering. Another way to achieve redundancy is through multiplying components of the systems or, in other words through creating replicas. This strategy accounts for malfunctioning components in the case of a failure, through procedures of communication or voting between them. A type of failures that typically concern researchers are modeled as Byzantine. According to the protocol of Byzantine Fault Tolerance (BFT) [30], there are n identical components in the system, t of which have been compromised by an intruder. Taking into account that each component can communicate confidentially with another one, there is a maximum number of compromised components that can be tolerated, in order for the system to continue operating properly. There are two basic categories of Byzantine agreement (or BFT); consensus and broadcast [31]. To explain in further detail, we assume for both cases a number of n components, namely, c_1, c_2, ..., c_n. In consensus mode, each component c_i holds an input x_i, and decides on an output y_i. Consensus is achieved if (a) for all components c_i that hold the same input x_i, then all 'loyal' components also hold the same output y_i; and (b) all 'loyal' components decide on the same output y_i. Note that the word 'loyal' here is used with the meaning of uncompomised. In broadcast mode, one of the components (c_s) is the sender, and transmits a message x_s. Then every other component decides on an output y_i. Broadcast is achieved if (a) for a 'loyal' sender, the 'loyal' components decide on an output $y_\mathrm{i} = x_s$; and (b) all 'loyal' components decide on the same output y_i. Figure 3.1 illustrates how broadcast mode fails when at least one of circumstances (a) or (b) are not met. In the sketch on the left side, one of the components that receive the sender's message is 'disloyal', and broadcasts a different message to the other component. Not all components are 'loyal', thus not all of them decide on the same output, as suggested in (b) above. In the sketch on the right side, the sender is disloyal, and broadcasts two different messages to the two components. In this case, the components cannot decide on the same output, even if all of them are 'loyal', as suggested in (a) above. Leslie Lamport, Robert Shostak and Marshall Pease [32] have proved that t faults can be tolerated when there are $3t+1$ replicas in the system architecture [30], [33]. The protocol according to which the components communicate with each other is considered synchronous. Generally, the decision on how much replication is needed in a system could require an extensive study. A system is called k-fault tolerant if the maximum number of faulty components it can survive and operate normally is k. There are several variations of BFT protocols, each of which serves a different purpose and fits better to certain systems. Most of the research interest on this topic is focused on the robustness of these protocols, i.e., the good performance when a fault occurs [34]–[36]. However, some of them do not achieve a satisfactory degree of robustness. The primary reason for this is the fact that the voting process relies on a specific replica, which is used to order requests. Thus, it is very possible that this replica can be manipulated by attackers. A method to fix this issue would be to not rely on only one replica for ordering requests for the system [37]. In this case, all replicas have to verify that the have received a message from the others.

Then, the procedure of finding out which ones are correct or not introduces some delay, which decreases the system's performance. Another approach for implementing BFT calls for multiple procedures working in parallel [38]; upon a request order from the replicas, one of them in each procedure is working by itself, and then the time needed to execute the request is compared among all of them, to find out if one of these is malicious.

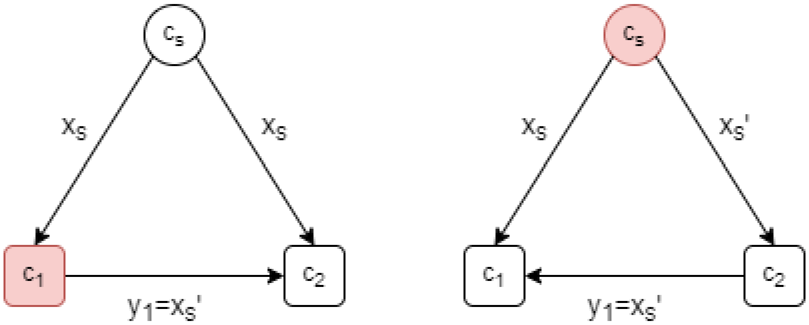


Figure 3.1. Broadcast mode failure with 'disloyal' component (left) and 'disloyal' sender (right).

Regarding the recovery of the system, a way has to be found so that the system regains full functionality after a failure or an attack. The most common method is rollback recovery, which means that the system needs to keep a specific state so it can return to that in the case of a failure [39]. In other words, the system goes back to a safe state in order to recover. This requires continuous recording of a system's state and can prove to be costly. Another method in bibliography is forward recovery, where the system keeps running, but instead of using data from the failed state, an interpolation scheme is implemented, based on the data not affected by the failure [40]. It is thus attempted to bring the system to a new state that is considered safe. This method is only helpful when there is knowledge of the possible errors. Lastly, other methods are based on restarting the system every time a fault is present.

3.2 Artificial Diversity

As discussed above, redundancy is crucial in order to achieve fault tolerance in a system. However, the identical features of the redundant components is not enough to reach the goal of fault tolerance. There should be some kind of diversity. If we assume that the controllers or actuators in a plant are redundant, there is a need to consider how to make them diverse. Artificial diversity basically refers to the process of creating several different variants of a program, with a purpose to achieve safety and security. Variants are essentially different versions of software, as shown in Figure 3.2. To understand this more clearly, one can think for example the case when different people choose to implement software for similar purposes. Multiple variants of a program can offer the same functionality through different implementations.

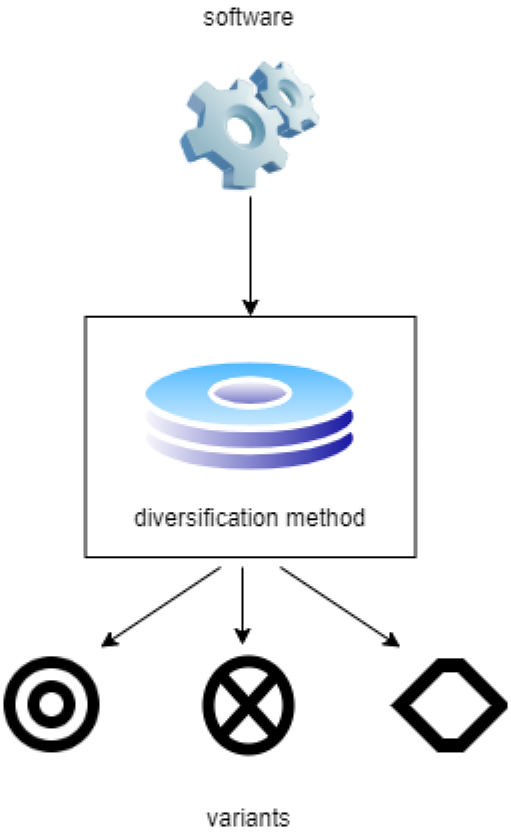


Figure 3.2. Different variants are created from the same software version.

Software and hardware in industrial applications are characterized by homogeneity. This simplifies several functions of a system, but in the same time renders it more vulnerable to attacks. Through software diversity, attackers cannot acquire enough knowledge to exploit certain variants. There are several types of diversity in the literature. Some approaches replace the order of instructions in the software [41], others change the order of blocks [42], move pieces of code to other functions , or encode the program in different ways [43]–[45]. There are also methods where variants are created by using different programming languages and different compilers, or by modifying the memory allocation for the code [46]. Other procedures include garbage code insertion, which means that unnecessary parts of code are added to create more variants, and randomization of data through padding [47].

Generally, in most industrial systems, including nuclear reactors, the controllers used are Programmable Logic Controllers (PLCs). As seen in the Figure 3.3, PLCs are comprised of three main components; the input, their CPU, and the output. PLC inputs consist of signals they receive from sensors or other physical entities. Examples of outputs can be various parts of a system that are controlled by the PLC, such as valves or motors. The CPU is the main part of the PLC, which acts in the same way the brain acts in the human body. It gives directions to the controller to perform operations. The code exists in the memory, which remains as is even after the power is cut off. The operating cycle of the CPU consists of the steps of scanning the input, executing the program, and updating the output.

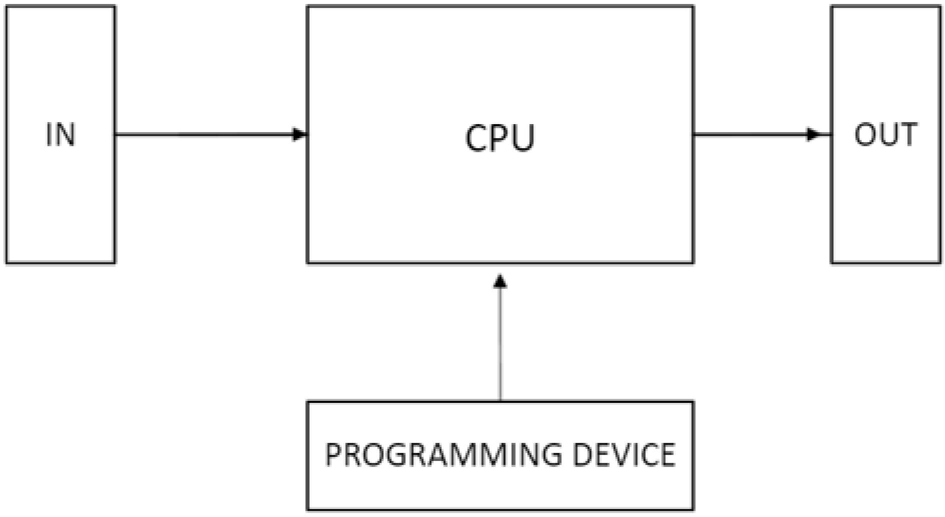


Figure 3.3. The structure of a Programmable Logic Controller (PLC).

According to what was previously described, redundancy should exploit one of these features of the controllers' instruction set architecture (ISA). Usually, diversity in PLCs is accomplished by modifying the software of the controllers. As a result, the executable code has variants that perform the same operations, with the difference that the locations for the inputs are in other places in the memory. This projects difficulties to cyber attackers who want to inject malformed data in the system. This implementation would behave as follows [48]; different software variants execute instructions, and other variants are expected to execute other actions. In the case where some tampered inputs crash (or deactivate) one of the variants, other variants take control and the system is tolerant against failures.

Another feature that should be considered is the inertia present in CPSs. This simply means that the physical nature of these systems makes them have a tendency to remain in the same state for some time. The changes in these systems are not abrupt; it takes some time for the physical system to evolve to a different state after a stimulation. Inertia can thus also provide fault tolerance, in the sense that the system can avoid failure because it can tolerate bad operation for a few cycles. This characteristic is crucial, as it gives an opportunity to the controllers to recover and reboot in case of a failure.

3.3 Problem Approach

The main approach towards achieving the cybersecurity of a nuclear reactor is based on an already existing algorithm developed by ONR [48], called RHIMES. Many different versions of this algorithm have been studied. The one analyzed in this work makes use of redundant components in a system, while in the same time keeps the system operating despite any threats. This modified version of the original RHIMES algorithm is presented in Figure 3.4 [48]. The main components of the architecture are the following; three (redundant) controllers shown in blue color, a delay queue (shown on the left of the bottom controller), a comparator (on the right of the top two controllers), and a multiplexer, which finally is connected to the actual plant. The redundancy in the controllers in combination with artificial diversity has been already discussed in the sections above. By executing diversified variants with the same inputs, the controllers are capable of detecting when an attack happens, if one of the variants fails. For this implementation, and for the reason that here the controllers are simulated, the diversification is expressed in an alternative way. The signal that is used to denote the failure of at least one of the top two controllers is the time by which they have to produce a result. In this implementation, it is considered that the controllers are allowed a specific time period to generate a result, and a specific time period to reboot after they have failed. In the case of a bad message, it is considered that the controllers are not able to produce a result within this specified time period. This is when they undergo a failure (crash) status. If at least one of them is in a crash status, the comparator detects it. The delay queue is connected to the input of the bottom controller, to keep it in an isolated, protected state (as a rollback recovery method). This happens because any data that will cause the top controllers to crash will just stay in the queue and will not affect the third controller. In case at least one of the top controllers crash, the queue gets emptied, so that the system disposes of the malicious inputs. The multiplexer selects between the output of the top controllers, and the output of the bottom controller. The control signal that indicates which one to choose comes from the comparator.

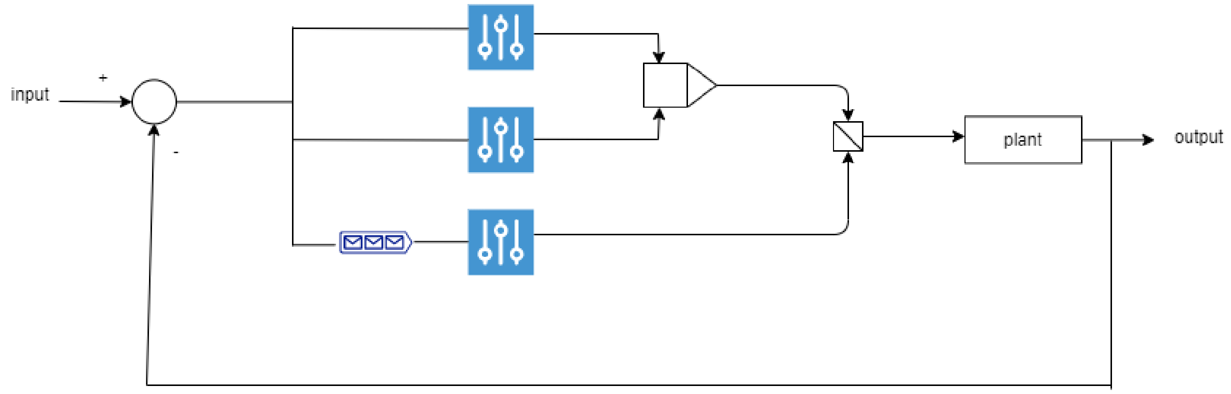


Figure 3.4. Modified version of the original RHIMES algorithm.

The algorithm performs as follows; the error term from the subtraction of the output from the setpoint is fed as input to the first two controllers, as well as to the delay queue. The outputs of the first two controllers are compared, and then there are two cases of operation. In the first case, which can be seen in Figure 3.5, let it be assumed that there is no bad message in the input. The error term is first distributed to the top two controllers and to the delay queue. Later, the controllers process this input and produce their outputs. In this case, both of them generate results within a specific amount of time, which means that no controller has crashed. Meanwhile, the input progresses further in the queue. The comparator does not detect a crash, thus the signal from the top two controllers reaches the multiplexer and finally the plant. Any data in the bottom controller remain unused.

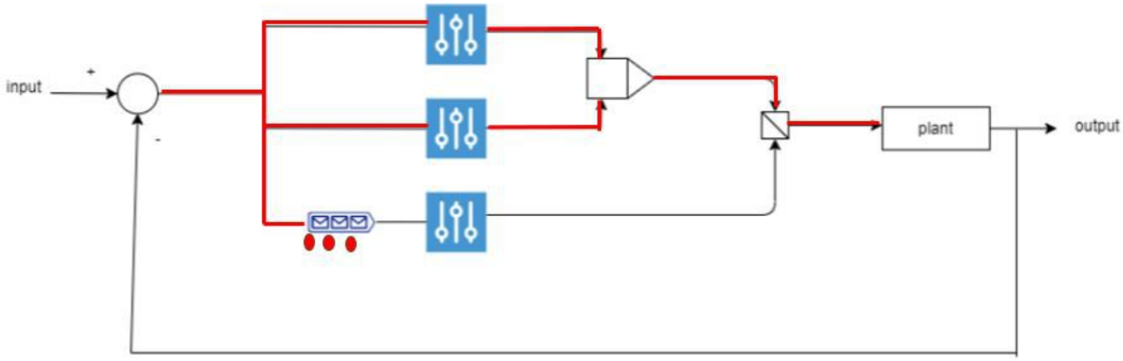


Figure 3.5. Information flow in the system in case of a good message.

In the case of a bad message, shown in Figure 3.6, at least one of the controllers crashes, because (for example) the top controller (red color) is not able to produce a result by a specific time. This automatically triggers the comparator, which sends a signal to the multiplexer, so that the output of the isolated (bottom) controller is used and propagates towards the plant. This controller contains safe data, as it still holds information from previous operating cycles. In the same time, the contents of the delay queue are emptied, so that the bad message never reaches the isolated safe controller. Until the plant processes the controller output and creates a new output, the crashed components have rebooted and the system resumes normal operation.

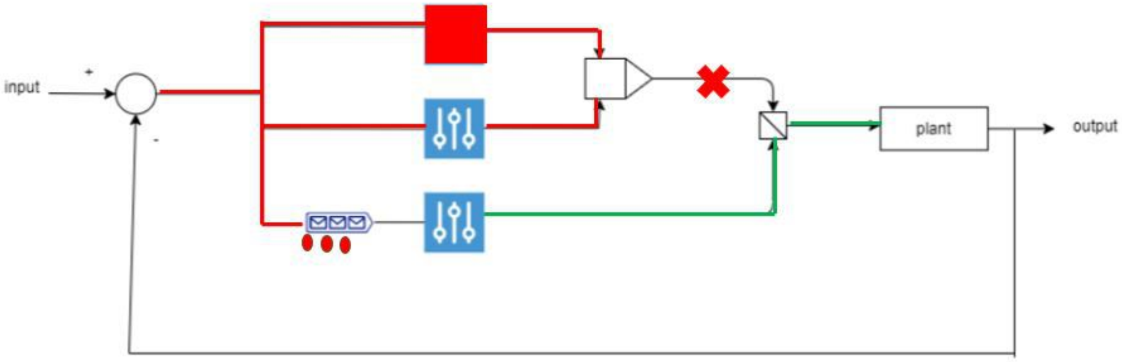


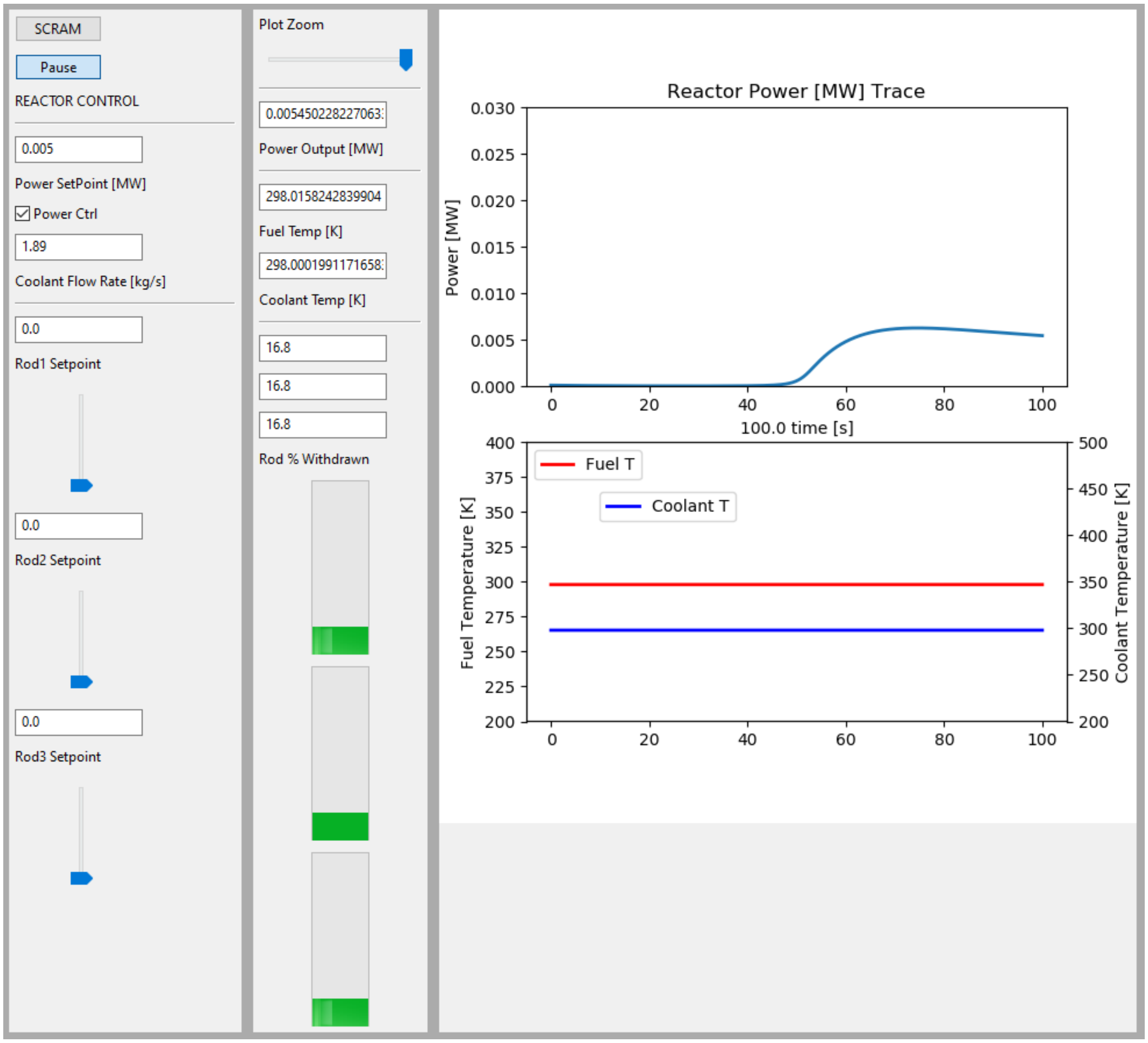
Figure 3.6. Information flow in the system in case of a bad message.

4. RESULTS

4.1 GUI Initial Implementation and Development

In order to test and establish the protection architecture, a Graphical User Interface (GUI) is a nice environment for the user to interact with the code. Of course, testing an actual research reactor under several different attack scenarios would not be possible at this point. It would be an extremely dangerous procedure to go through, while it would also create serious problems with the reactor's licensing and operation. For that reason, a simulator that controls and monitors several reactor parameters is implemented in Python, and the GUI is just a convenient environment for the users. As a starter, to help with the implementation, we use a generic GUI for reactors which has been created by William Gurecky for educational purposes [49]. Modifications on that code have helped bring the system in a state similar with that of PUR-1. All the reactor parameters are adjusted to account for the Purdue Reactor, such as the reactor volume, the coolant flow rate, the constants for the point kinetics equations, the fuel and coolant temperatures, and others. Furthermore, the GUI is modified in order to include three control rods, as these exist in PUR-1. Rod1 in the sliders refers to SS1, Rod2 refers to SS2 and lastly Rod3 corresponds to RR. The calibration of the rods is added, in order to calculate the reactivity properly, while the PID controller is tuned accordingly to Ziegler-Nichols method, as presented in Chapter 2, followed by some fine-tuning to get the best response. The graphical user interface in its initial form is shown in Figure 4.1.

As can be seen, the GUI is able to control the reactor power, when the user inputs a specific setpoint in the respective box. This function is activated when the power control checkbox is selected. This is accomplished through the PID controller, as this was explained in Chapter 2. In addition, the user is able to input a specific amount of coolant flow rate, although at first it is just fixed in a specific recommended value. All three rods can be individually raised or lowered in the reactor. The GUI also monitors and plots the temperatures of the coolant and the fuel, as this can prove important for a reactor's operation. Finally, the output power is plotted and the user can monitor the rods, as these are lowered or raised by themselves, in order to achieve the specific output assigned by the user. When



Power SetPoint [MW]: 0.005

☑ Power Ctrl

Coolant Flow Rate [kg/s]: 1.89

Rod1 Setpoint: 0.0

Rod2 Setpoint: 0.0

Rod3 Setpoint: 0.0

Power Output [MW]: 0.00545022822706З

Fuel Temp [K]: 298.0158242839904

Coolant Temp [K]: 298.0001991171658

16.8

16.8

16.8

Figure 4.1. Control of reactor output power by specifying a setpoint.

the power control checkbox is not selected, the reactor tries to settle the rods in the position assigned by the user in the left column. The scram button can be used in any time to scram the reactor.

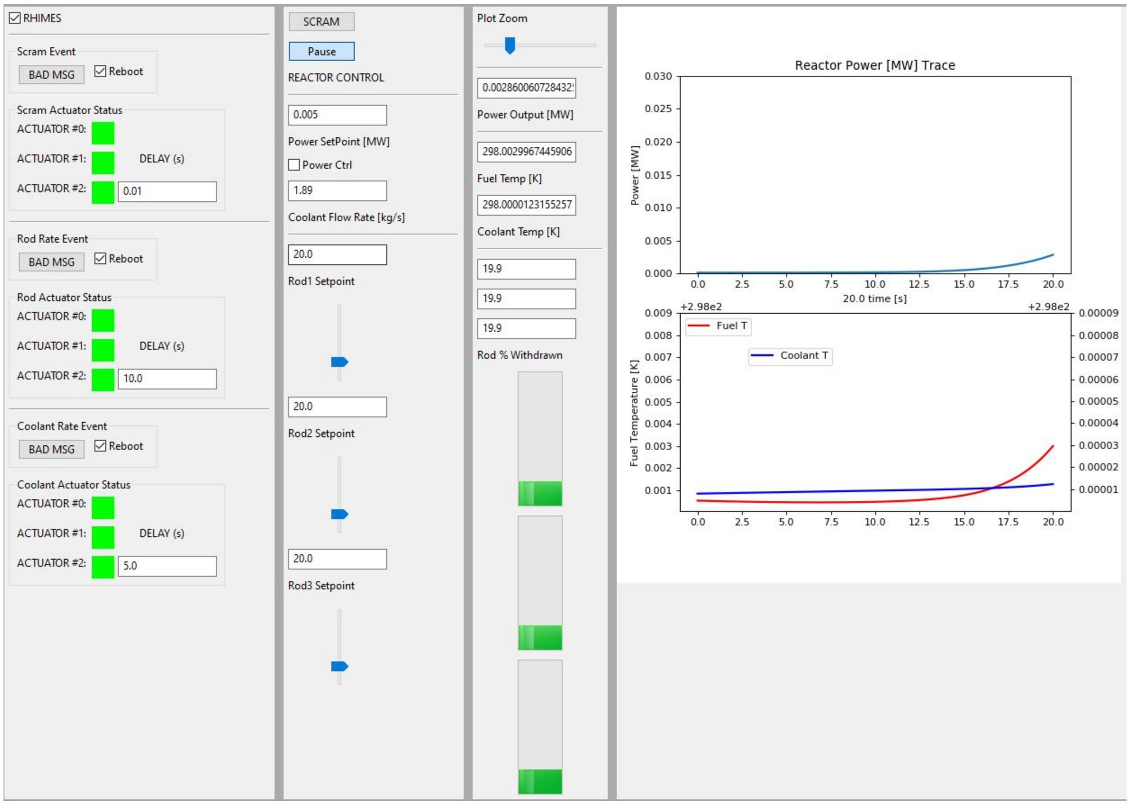
4.2 PUR-1 Cybersecurity

The following figures have the purpose of showing how the algorithm works. The modified RHIMES algorithm was implemented and added in the GUI with the collaboration of the

Office of Naval Research (ONR), as well as the MITRE team. In Figure 4.2, the GUI shows the system's state at a random time. As can be seen on the left panel there are the actuators that pertain to different types of attacks on the system. For example, the top one refers to malicious inputs coming to the system, asking the reactor to scram. The other is for attacks related to rod positions, and the last one refers to attacks against the coolant rate. The delay for every attack type, shown in white boxes in the left column of the GUI, refers to the time period between the time of the attack and the recovery of the system. The specified seconds in the boxes represent the optimal time period that the system can be under the influence of a malicious input, and still be able to recover afterwards. After several trials with the GUI, the time periods shown in Figure 4.2 are specified. In the same figure, it can be seen that the power setpoint has been set to 5 kW, and the system is working to reach that point and stabilize. Coolant temperature is stable, while fuel temperature rises slightly as power rises as well. If for some reason the protective algorithm is deactivated by the user, what happens is shown in Figure 4.3. Since the reactor got a message for scram, the rods are pushed in (% withdrawn in the third column is now zero) and the power level drops almost instantly to zero. Moreover, the coolant temperature slightly rises, and the fuel temperature decreases.

In another scenario, in Figure 4.4, the reactor is on the process of reaching a setpoint of 9 kW. In this case, the protection algorithm is activated and its results are shown in Figure 4.5. Here the safety system does not let the rods (and subsequently the power level) to fall, thus protecting the plant from a malicious and unwanted scram. The rod levels are still in the positions specified by the user, which will then change as the system continues to reach the requested power level of 9 kW. Also, the fuel temperature continues to rise with power.

In order to also show how the algorithm works for bad messages regarding rod positions, Figures 4.6, 4.7 and 4.8 present a related example. Initially, as seen in Figure 4.6, the withdrawn percentage for rods SS2 and RR has been selected as 14%, and for rod SS1 as 20%. The output power is set for 1 kW. When a bad message is sent to the system, it allows the SS1 rod motion to run open loop, meaning that it no longer belongs to the closed loop system controlled by the PID controller. In other words, when a bad message enters the system, a random withdrawn length is assigned to the rod, so it runs unchecked. As can be seen in Figure 4.7, the SS1 rod setpoint of 20% gets exceeded (see 20.3% in the third



Scram Event – DELAY (s): 0.01

Rod Rate Event – DELAY (s): 10.0

Coolant Rate Event – DELAY (s): 5.0

Power SetPoint [MW]: 0.005

Coolant Flow Rate [kg/s]: 1.89

Rod1 Setpoint: 20.0

Rod2 Setpoint: 20.0

Rod3 Setpoint: 20.0

Power Output [MW]: 0.002860060728432

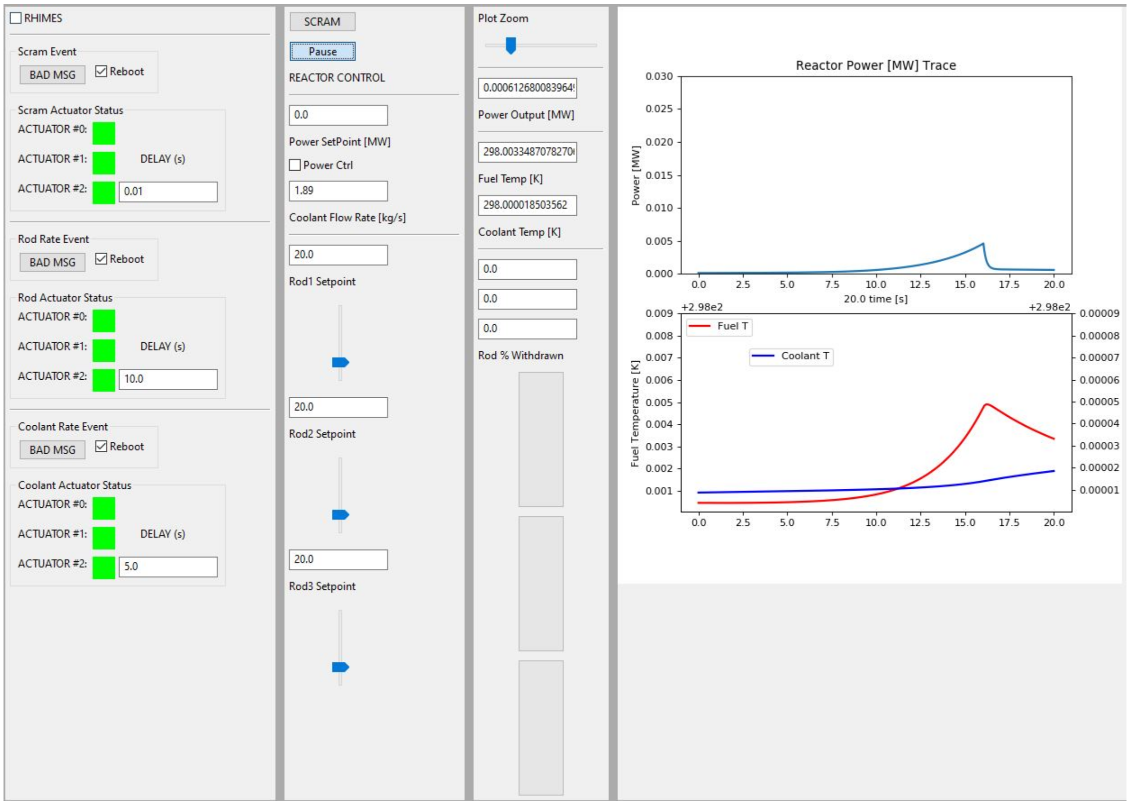
Fuel Temp [K]: 298.0029967445906

Coolant Temp [K]: 298.0000123155257

Rod % Withdrawn: 19.9, 19.9, 19.9

Figure 4.2. State of reactor before malicious SCRAM attempt for a power setpoint of 5 kW.

column). During the delay time, the first two actuators are being rebooted (see yellow color in the first column in Figure 4.7), while in the same time the third actuator takes control, gets activated and flushes the delay queue. As can later be seen in Figure 4.8, the system recovers after some time. There is a small time period when the power level slightly exceeds the setpoint. After recovery, the system tries to settle back to that specified power level. The SS1 rod position returns to the specified setpoint of 20%, and operation becomes normal again.



Scram Actuator Status
ACTUATOR #0: 🟩
ACTUATOR #1: 🟩 DELAY (s)
ACTUATOR #2: 🟩 0.01

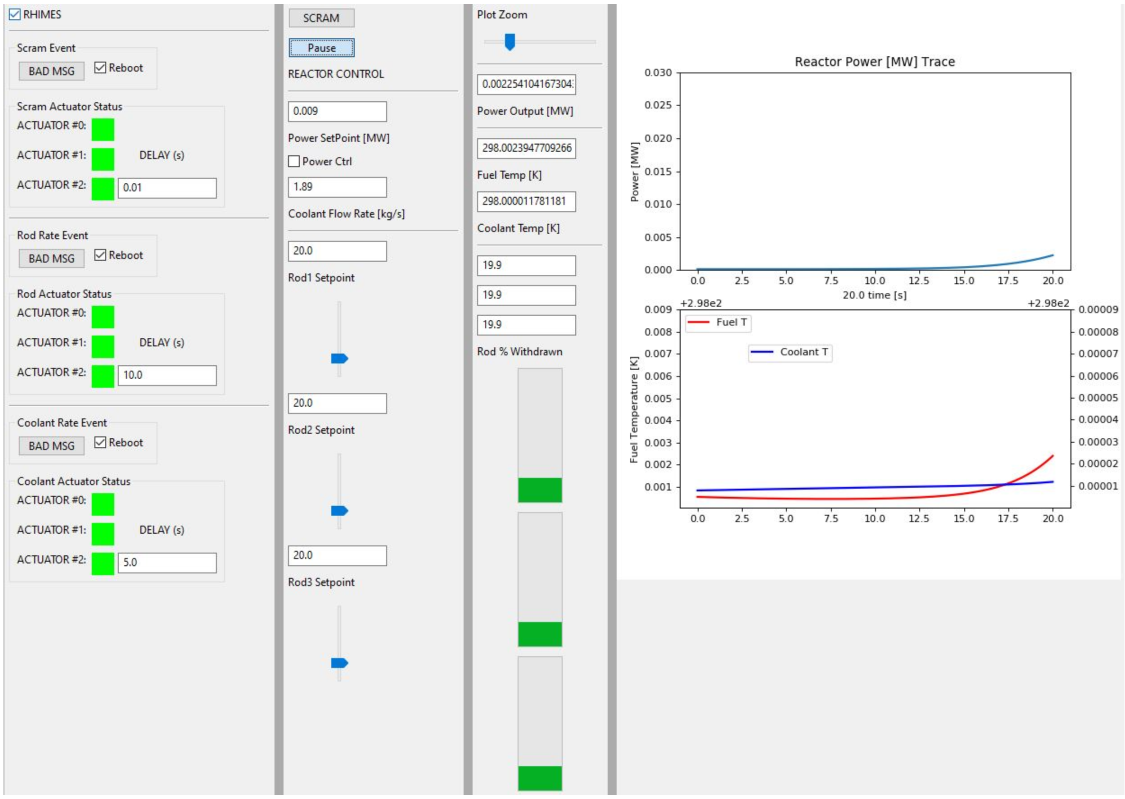
Rod Actuator Status
ACTUATOR #0: 🟩
ACTUATOR #1: 🟩 DELAY (s)
ACTUATOR #2: 🟩 10.0

Coolant Actuator Status
ACTUATOR #0: 🟩
ACTUATOR #1: 🟩 DELAY (s)
ACTUATOR #2: 🟩 5.0

0.0
Power SetPoint [MW]
☐ Power Ctrl
1.89
Coolant Flow Rate [kg/s]
20.0
Rod1 Setpoint
20.0
Rod2 Setpoint
20.0
Rod3 Setpoint

0.00061268008396
Power Output [MW]
298.003348707827
Fuel Temp [K]
298.000018503562
Coolant Temp [K]
0.0
0.0
0.0
Rod % Withdrawn

Figure 4.3. State of reactor after malicious SCRAM attempt for a power setpoint of 5 kW with safety system deactivated.



☑ RHIMES

Scram Event
☑ Reboot

Scram Actuator Status
ACTUATOR #0: 🟩
ACTUATOR #1: 🟩
ACTUATOR #2: 🟩
DELAY (s): 0.01

Rod Rate Event
☑ Reboot

Rod Actuator Status
ACTUATOR #0: 🟩
ACTUATOR #1: 🟩
ACTUATOR #2: 🟩
DELAY (s): 10.0

Coolant Rate Event
☑ Reboot

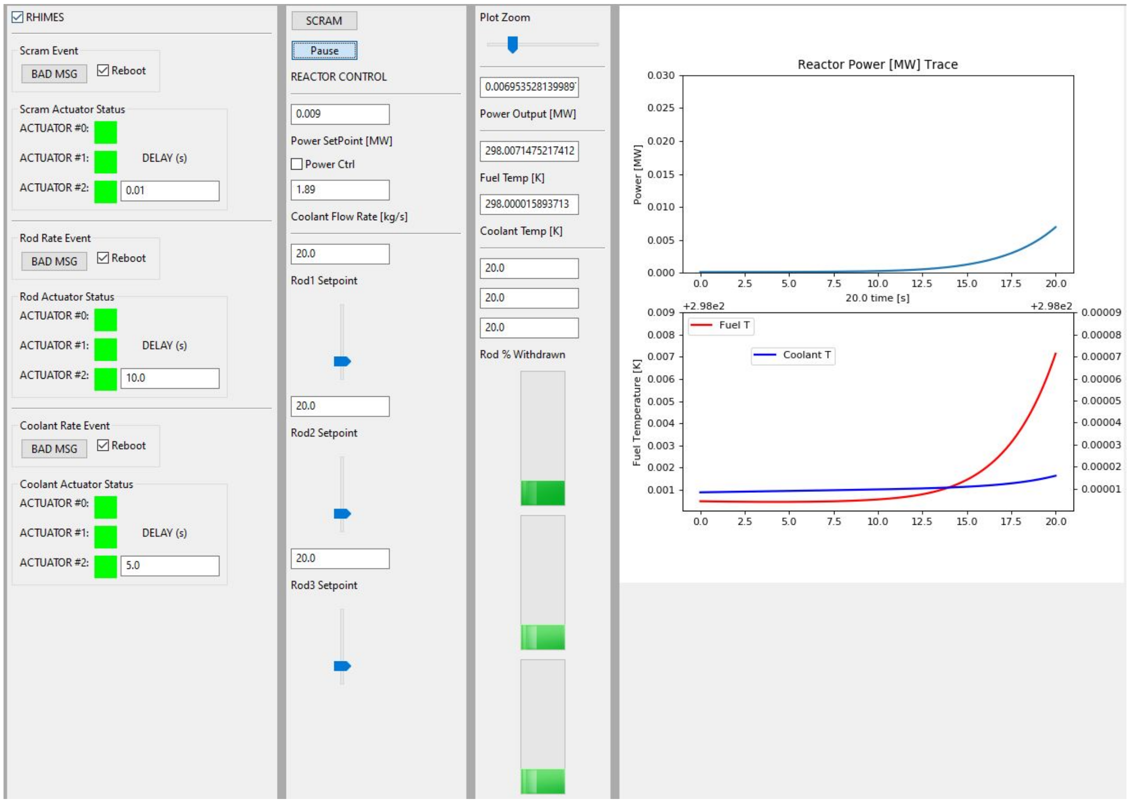
Coolant Actuator Status
ACTUATOR #0: 🟩
ACTUATOR #1: 🟩
ACTUATOR #2: 🟩
DELAY (s): 5.0

REACTOR CONTROL

Power SetPoint [MW]: 0.009
☐ Power Ctrl
Coolant Flow Rate [kg/s]: 1.89
Rod1 Setpoint: 20.0
Rod2 Setpoint: 20.0
Rod3 Setpoint: 20.0

Power Output [MW]: 0.002254104167304
Fuel Temp [K]: 298.0023947709266
Coolant Temp [K]: 298.000011781181
Rod % Withdrawn: 19.9
19.9
19.9

Figure 4.4. State of reactor before malicious SCRAM attempt for a power setpoint of 9 kW.



☑ RHIMES

Scram Event

BAD MSG ☑ Reboot

Scram Actuator Status

ACTUATOR #0: 🟩

ACTUATOR #1: 🟩 DELAY (s)

ACTUATOR #2: 🟩 0.01

Rod Rate Event

BAD MSG ☑ Reboot

Rod Actuator Status

ACTUATOR #0: 🟩

ACTUATOR #1: 🟩 DELAY (s)

ACTUATOR #2: 🟩 10.0

Coolant Rate Event

BAD MSG ☑ Reboot

Coolant Actuator Status

ACTUATOR #0: 🟩

ACTUATOR #1: 🟩 DELAY (s)

ACTUATOR #2: 🟩 5.0

REACTOR CONTROL

0.009

Power SetPoint [MW]

☐ Power Ctrl

1.89

Coolant Flow Rate [kg/s]

20.0

Rod1 Setpoint

20.0

Rod2 Setpoint

20.0

Rod3 Setpoint

0.006953528139989

Power Output [MW]

298.0071475217412

Fuel Temp [K]

298.000015893713

Coolant Temp [K]

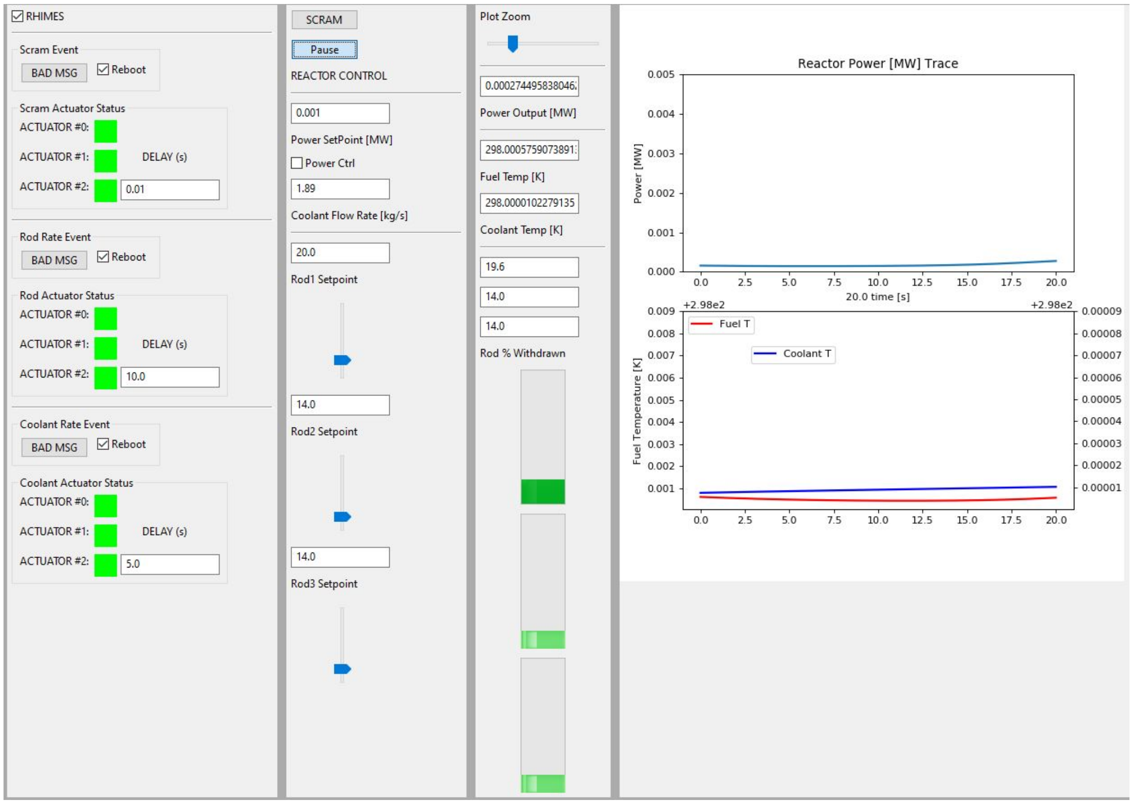
20.0

20.0

20.0

Rod % Withdrawn

Figure 4.5. State of reactor after malicious SCRAM attempt for a power setpoint of 9 kW with safety system activated.



Scram Actuator Status DELAY (s): 0.01

Rod Actuator Status DELAY (s): 10.0

Coolant Actuator Status DELAY (s): 5.0

Power SetPoint [MW]: 0.001

Coolant Flow Rate [kg/s]: 1.89

Rod1 Setpoint: 20.0

Rod2 Setpoint: 14.0

Rod3 Setpoint: 14.0

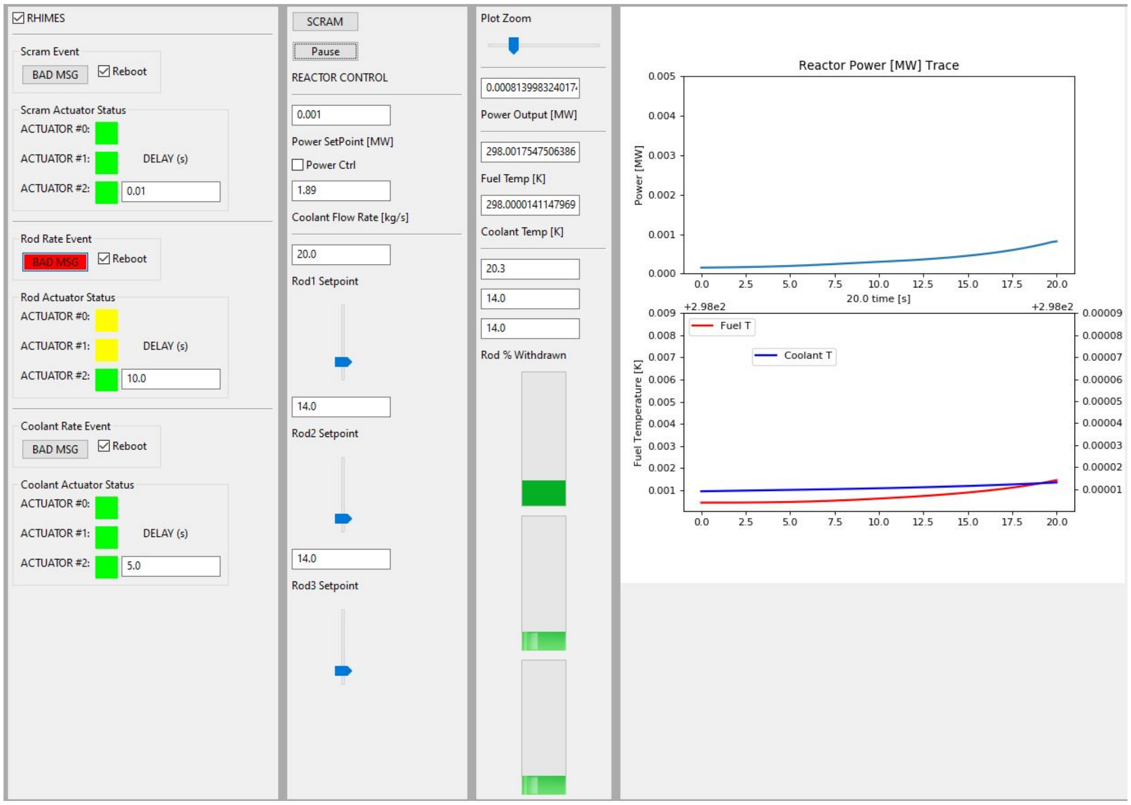
Power Output [MW]: 0.000274495838046

Fuel Temp [K]: 298.0005759073891

Coolant Temp [K]: 298.0000102279135

Rod % Withdrawn: 19.6, 14.0, 14.0

Figure 4.6. State of reactor before malicious rod message.



☑ RHIMES

Scram Event: ☑ Reboot

Scram Actuator Status: DELAY (s) 0.01

Rod Rate Event: ☑ Reboot

Rod Actuator Status: DELAY (s) 10.0

Coolant Rate Event: ☑ Reboot

Coolant Actuator Status: DELAY (s) 5.0

0.001
Power SetPoint [MW]

☐ Power Ctrl

1.89
Coolant Flow Rate [kg/s]

20.0
Rod1 Setpoint

14.0
Rod2 Setpoint

14.0
Rod3 Setpoint

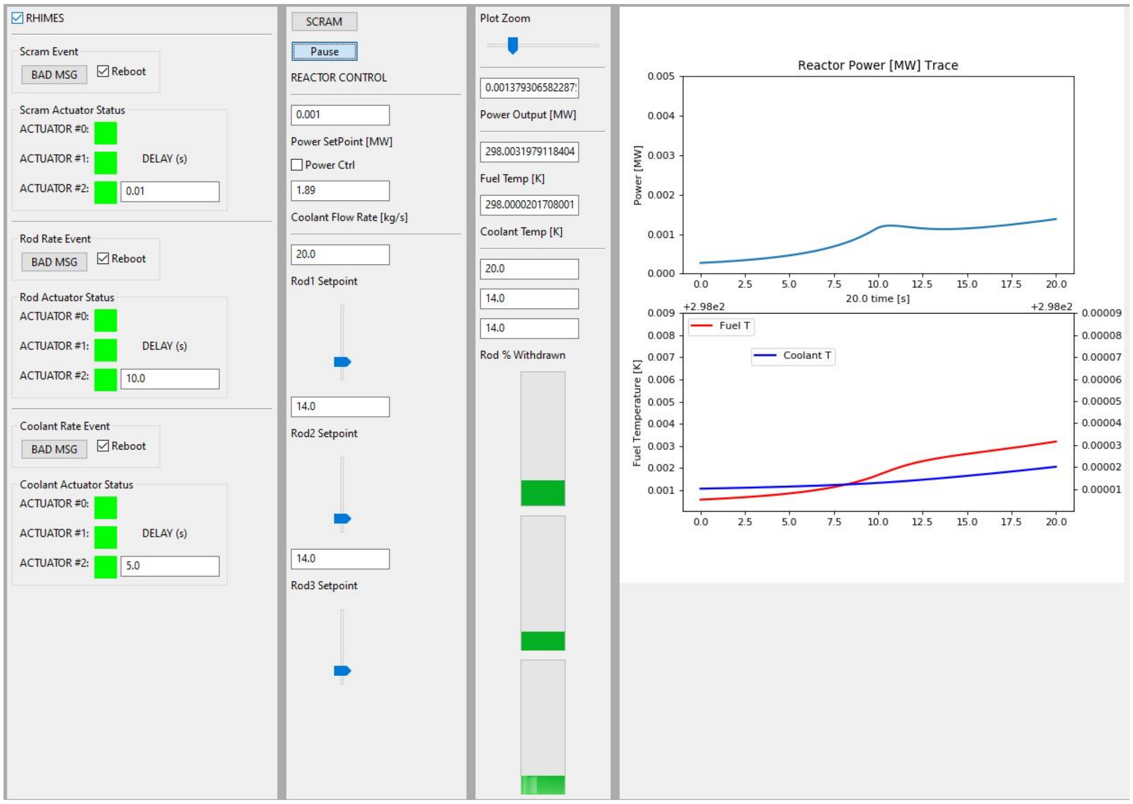
0.000813998324017
Power Output [MW]

298.0017547506386
Fuel Temp [K]

298.0000141147969
Coolant Temp [K]

20.3
14.0
14.0
Rod % Withdrawn

Figure 4.7. State of reactor during malicious rod message.



☑ RHIMES

Scram Event
☑ Reboot

Scram Actuator Status
ACTUATOR #0: 🟩
ACTUATOR #1: 🟩
ACTUATOR #2: 🟩
DELAY (s) 0.01

Rod Rate Event
☑ Reboot

Rod Actuator Status
ACTUATOR #0: 🟩
ACTUATOR #1: 🟩
ACTUATOR #2: 🟩
DELAY (s) 10.0

Coolant Rate Event
☑ Reboot

Coolant Actuator Status
ACTUATOR #0: 🟩
ACTUATOR #1: 🟩
ACTUATOR #2: 🟩
DELAY (s) 5.0

REACTOR CONTROL
0.001
Power SetPoint [MW]
☐ Power Ctrl
1.89
Coolant Flow Rate [kg/s]
20.0
Rod1 Setpoint
14.0
Rod2 Setpoint
14.0
Rod3 Setpoint

0.001379306582287
Power Output [MW]
298.0031979118404
Fuel Temp [K]
298.0000201708001
Coolant Temp [K]
20.0
14.0
14.0
Rod % Withdrawn

Figure 4.8. State of reactor after recovery from malicious rod message.

5. CONCLUSIONS AND FUTURE WORK

5.1 Conclusions

Nuclear energy systems are comprised of Instrumentation and Control (I&C) systems, which are responsible for the plants' control, monitoring and protection. Lately, analog I&C systems have become obsolete, and are observed to have increased operation and maintenance (O&M) costs. This is the main reason why power plants have started to adopt digital control systems. Digital I&C has proved to bring many advantages to systems' operation, overall costs, and equipment. However, there are several drawbacks, mostly related to the safety and security of systems. On the same note, the ever growing popularity of Cyber Physical Systems (CPSs) further increases the risks for cyber attacks, vulnerabilities, and failures. This thesis has the purpose to implement an architecture that can successfully protect a nuclear reactor, and more specifically the Purdue University Reactor Number One (PUR-1), from these types of attacks. The first step followed in this effort is the physics modelling of the reactor. This is done through the use of reactors' governing point kinetics equations, reactivity calculations, as well as controller tuning and modelling for the purpose of controlling the reactor's output power. In order to face a variety of attacks, an architecture for fault tolerance is studied. The most popular strategy towards achieving fault tolerance is through the use of identical redundant components (replicas), which undergo a voting process to reveal possible failures. The most used protocol for this case is the Byzantine Fault Tolerance (BFT) algorithm. However, since redundancy by itself is not capable of achieving a high degree of fault tolerance, artificial diversity is additionally explored. This creates variants in the redundant components. As a result, some variants execute specific instructions, and other variants are expected to execute other actions. In the case where some tampered inputs crash (or deactivate) one of the variants, other variants take control and the system is tolerant against failures.

The actual algorithm for the system studied in this work uses three redundant controllers and performs as follows; the error term from the subtraction of the output from the setpoint is fed as input to the first two controllers, as well as to the delay queue connected to the third controller. The outputs of the first two controllers are compared, and then there are two cases of operation. In the case of a good message in the input, the variants in the controllers do not crash, thus the signal from the top two controllers reaches the plant. In the case of a bad message, at least one of the two controllers crashes, because at least one of the code variants fails due to the diversity. This automatically triggers the comparator, which sends a signal so that the output of the isolated controller is used and propagates towards the plant. After implementing a Graphical User Interface (GUI) to simulate and visualize the system's state, it is shown that PUR-1 is able to overcome bad messages regarding scram or control rod positions, when the protection architecture is activated.

5.2 Future Work

The increased vulnerabilities and failure risks accompanying the digital I&C in CPSs – including nuclear reactors – could also be tackled with the integration of advanced informatics into the reactor monitoring system. More specifically, Artificial Intelligence (AI) and Machine Learning (ML) could be deployed towards a system's on-line monitoring [50]–[54], as has also been shown in our recent work [55], [56]. When one considers the ever-growing data volumes produced in a system, the Dynamic Data Driven Application Systems (DDDAS) paradigm is of great value in prioritizing and categorizing data in accordance with system dynamics. DDDAS is a framework based on systems that include physical models, measurements, and computational parts, as seen in Figure 5.1. It can provide suitable solutions to risks arising from connectivity of CPSs through big data and ML. A physical system, such as a nuclear reactor in this case, can be modelled through a computational simulation. DDDAS and CPSs can connect the physical with the cyber world, since measurements and computations produce patterns amenable to approaches involving AI methods for decision making. This is highly illustrated in some of our recent work [57], [58]. Frederica Darema, who pioneered the DDDAS paradigm, states the following [59]:

> "in DDDAS instrumentation data and executing application models of these systems become a dynamic feedback control loop, whereby measurement data are dynamically incorporated into an executing model of the system in order to improve the accuracy of the model (or simulation), or to speed-up the simulation,

> and in reverse the executing application model controls the instrumentation process to guide the measurement process. DDDAS presents opportunities to create new capabilities through more accurate understanding, analysis, and prediction of the behavior of complex systems, be they natural, engineered, or societal, and to create decision support methods which can have the accuracy of full-scale simulations, as well as to create more efficient and effective instrumentation methods, such as intelligent management of Big Data, and dynamic and adaptive management of networked collections of heterogeneous sensors and controllers. DDDAS is a unifying paradigm, bringing together computational and instrumentation aspects of an application system, which extends the notion of Big Computing to span from the high-end to the real-time data acquisition and control, and it's a key methodology in managing and intelligently exploiting Big Data."

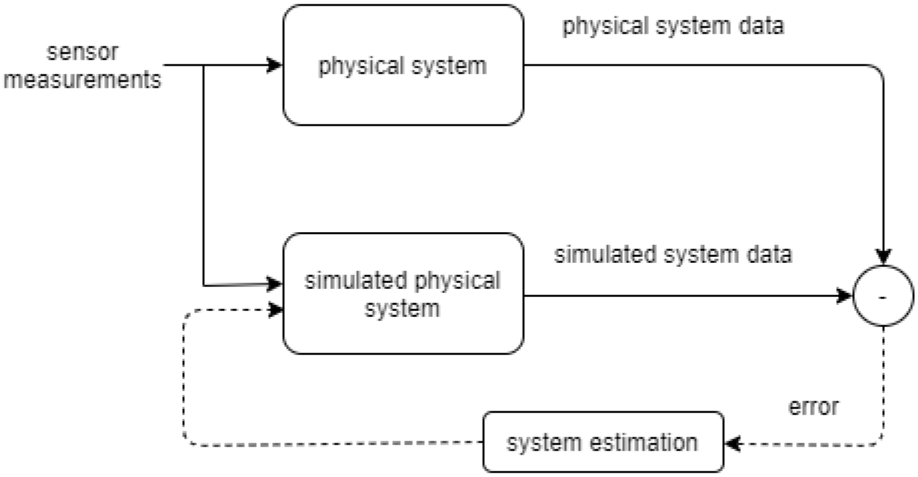


Figure 5.1. Dynamic Data Driven Application Systems (DDDAS) framework.

In the past, one might have argued that the large amount of data produced by CPSs is not completely useful, as according to Occam's razor (or the law of parsimony), simpler explanations of entities are preferred, and abstraction is the understanding of a real entity [60]. Modern researchers are trying to reveal a more formalized explanation for Occam's razor, and are finding promising results in Bayesian inference.

Bayesian inference is a statistical procedure that refers to updating beliefs in the best possible way when making new observations [61]. For a system, this would mean to update its state when receiving new inputs from sensors. Bayes' theorem is essentially an algorithm that combines prior experience and current evidence and is expressed as follows:

$$p(b|o,m) = \frac{p(o|b,m) \cdot p(b|m)}{p(o|m)} \tag{5.1}$$

The term $p(b|m)$ is the probability of a belief (b), given a specific modelling (m) of a system. The term $p(o|b,m)$ means the probability within a specific modelling that an observation (o) would be obtained, given a belief. The term $p(o|m)$ is the probability to get an observation, given a specific modelling for the system. Finally, the term $p(b|o,m)$ encodes what would be the optimal belief after making a new observation. In other words, in order to update a belief, first a system has to take the initial belief, then combine it with the likelihood (i.e., how logical a new observation is with several possible beliefs), and lastly consider to what degree this observation matches any possible beliefs in the model.

In this context, a nuclear system in particular would be able to use past and current observations in order to make decisions about an updated, optimal state. The knowledge of current and past states can detect excitations in the system, which in turn can produce activation patterns.

Another approach involves the Markov Decision Process (MDP), where the current state of a system depends only on its immediate previous step. Respectively, its next state depends only on the current state. This simply means that, when making decisions, all knowledge about older past states is included in the beliefs about the current state. Figure 5.2 depicts how a system can update its state this way. Basically, there is a decision maker that interacts with the system. The system provides the rewards, as well as the updated states, based on the actions of the decision maker. This consists the basis of reinforcement learning.

This simply means that, for example, a nuclear system can receive actions from the actuators, and then calculate an updated state and a reward. Rewards cannot be changed by the decision system; however, the latter has some knowledge on how rewards are calculated by the system, since they are based on the actions.

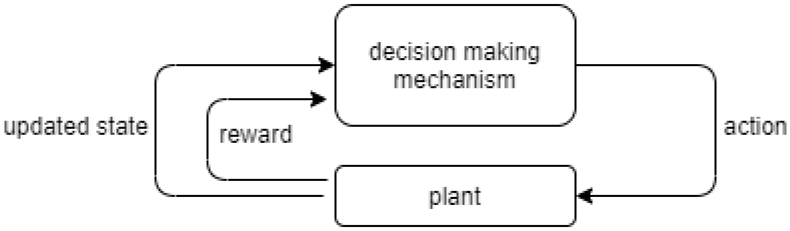


Figure 5.2. Cycle of reinforcement learning.

REFERENCES

[1] *Digital Instrumentation and Control Systems for New and Existing Research Reactors*, ser. Nuclear Energy Series NR-G-5.1. Vienna: INTERNATIONAL ATOMIC ENERGY AGENCY, 2021, ISBN: 978-92-0-118320-0. [Online]. Available: https://www.iaea.org/publications/13565/digital-instrumentation-and-control-systems-for-new-and-existing-research-reactors.

[2] R. Baheti and H. Gill, "Cyber-physical systems," *The impact of control technology*, vol. 12, no. 1, pp. 161–166, 2011.

[3] R. Rajkumar, I. Lee, L. Sha, and J. Stankovic, "Cyber-physical systems: The next computing revolution," in *Design Automation Conference*, 2010, pp. 731–736. DOI: 10.1145/1837274.1837461.

[4] S. Zanero, "Cyber-physical systems," *Computer*, vol. 50, no. 4, pp. 14–16, 2017. DOI: 10.1109/MC.2017.105.

[5] J. Lee, B. Bagheri, and H.-A. Kao, "A cyber-physical systems architecture for industry 4.0-based manufacturing systems," *Manufacturing letters*, vol. 3, pp. 18–23, 2015.

[6] W. Wang, F. Di Maio, and E. Zio, "A non-parametric cumulative sum approach for online diagnostics of cyber attacks to nuclear power plants," in *Resilience of Cyber-Physical Systems*, Springer, 2019, pp. 195–228.

[7] M. A. Arroyo, M. T. I. Ziad, H. Kobayashi, J. Yang, and S. Sethumadhavan, "Yolo: Frequently resetting cyber-physical systems for security," in *Autonomous Systems: Sensors, Processing, and Security for Vehicles and Infrastructure 2019*, International Society for Optics and Photonics, vol. 11009, 2019, 110090P.

[8] S. L. Eggers, "The nuclear digital i&c system supply chain cyber-attack surface," Jun. 2020. [Online]. Available: https://www.osti.gov/biblio/1634821.

[9] S. Samonas and D. Coss, "The cia strikes back: Redefining confidentiality, integrity and availability in security.," *Journal of Information System Security*, vol. 10, no. 3, 2014.

[10] A. Cardenas, S. Amin, B. Sinopoli, A. Giani, A. Perrig, S. Sastry, *et al.*, "Challenges for securing cyber physical systems," in *Workshop on future directions in cyber-physical systems security*, Citeseer, vol. 5, 2009.

[11] I. Ahmad, M. K. Zarrar, T. Saeed, and S. Rehman, "Security aspects of cyber physical systems," in *2018 1st International Conference on Computer Applications & Information Security (ICCAIS)*, IEEE, 2018, pp. 1–6.

[12] L. Robert, J. Michael, and C. Tim, “Analysis of the cyber attack on the ukrainian power grid,” *USA: Electricity Information Sharing and Analysis Centre (E-ISAC)*, 2016.

[13] J. Giraldo, D. Urbina, A. Cardenas, J. Valente, M. Faisal, J. Ruths, N. O. Tippenhauer, H. Sandberg, and R. Candell, “A survey of physics-based attack detection in cyber-physical systems,” *ACM Computing Surveys (CSUR)*, vol. 51, no. 4, pp. 1–36, 2018.

[14] M. Bishop, “About penetration testing,” *IEEE Security & Privacy*, vol. 5, no. 6, pp. 84–87, 2007.

[15] I. Schieferdecker, J. Grossmann, and M. Schneider, “Model-based security testing,” *arXiv preprint arXiv:1202.6118*, 2012.

[16] A. V. Uzunov, E. B. Fernandez, and K. Falkner, “Engineering security into distributed systems: A survey of methodologies.,” *J. UCS*, vol. 18, no. 20, pp. 2920–3006, 2012.

[17] J.-G. Song, J.-W. Lee, C.-K. Lee, K.-C. Kwon, and D.-Y. Lee, “A cyber security risk assessment for the design of i&c systems in nuclear power plants,” *Nuclear engineering and technology*, vol. 44, no. 8, pp. 919–928, 2012.

[18] A. V. Uzunov, E. B. Fernandez, and K. Falkner, “Securing distributed systems using patterns: A survey,” *Computers & Security*, vol. 31, no. 5, pp. 681–703, 2012.

[19] M. Van Steen and A. S. Tanenbaum, *Distributed systems*. Maarten van Steen Leiden, The Netherlands, 2017.

[20] J. Knight, J. Davidson, A. Nguyen-Tuong, J. Hiser, and M. Co, “Diversity in cybersecurity,” *Computer*, vol. 49, no. 4, pp. 94–98, 2016.

[21] B. Cox, D. Evans, A. Filipi, J. Rowanhill, W. Hu, J. Davidson, J. Knight, A. Nguyen-Tuong, and J. Hiser, “N-variant systems: A secretless framework for security through diversity.,” in *USENIX Security Symposium*, 2006, pp. 105–120.

[22] J. Rowe, K. N. Levitt, T. Demir, and R. Erbacher, “Artificial diversity as maneuvers in a control theoretic moving target defense,” in *National Symposium on Moving Target Research*, 2012.

[23] X. Luan, J. Zhou, and Y. Zhai, “Takagi-sugeno fuzzy load-following control of nuclear reactors based on reactor point kinetics equations,” in *International Conference on Nuclear Engineering*, American Society of Mechanical Engineers, vol. 45967, 2014, V006T13A011.

[24] A. Ashaari, T. Ahmad, M. Shamsuddin, and M. A. Abdullah, "State space modeling of reactor core in a pressurized water reactor," in *AIP Conference Proceedings*, American Institute of Physics, vol. 1605, 2014, pp. 494–499.

[25] X. Wang, L. H. Tsoukalas, T. Y. Wei, and J. Reifman, "An innovative fuzzy-logic-based methodology for trend identification," *Nuclear technology*, vol. 135, no. 1, pp. 67–84, 2001.

[26] C. Townsend, "Licensable power capacity of the pur-1 research reactor," Master's thesis, Purdue University Graduate School, 2019.

[27] V. Nelson, "Fault-tolerant computing: Fundamental concepts," *Computer*, vol. 23, no. 7, pp. 19–25, 1990. DOI: 10.1109/2.56849.

[28] N. Campregher, P. Y. Cheung, G. A. Constantinides, and M. Vasilko, "Reconfiguration and fine-grained redundancy for fault tolerance in fpgas," in *2006 International Conference on Field Programmable Logic and Applications*, IEEE, 2006, pp. 1–6.

[29] S. Anwar and L. Chen, "An analytical redundancy-based fault detection and isolation algorithm for a road-wheel control subsystem in a steer-by-wire system," *IEEE Transactions on Vehicular Technology*, vol. 56, no. 5, pp. 2859–2869, 2007. DOI: 10.1109/TVT.2007.900515.

[30] R. Anderson, *Security engineering: a guide to building dependable distributed systems.* John Wiley & Sons, 2020.

[31] M. Fitzi, "Generalized communication and security models in byzantine agreement," PhD thesis, ETH Zurich, 2002.

[32] L. Lamport, R. Shostak, and M. Pease, "The byzantine generals problem," in *Concurrency: the Works of Leslie Lamport*, 2019, pp. 203–226.

[33] L. Perronne and S. Bouchenak, "Towards efficient and robust bft protocols," in *Fast Abstract in the 46th Annual IEEE/IFIP International Conference on Dependable Systems and Networks*, 2016.

[34] Y. Amir, B. Coan, J. Kirsch, and J. Lane, "Byzantine replication under attack," in *2008 IEEE International Conference on Dependable Systems and Networks With FTCS and DCC (DSN)*, IEEE, 2008, pp. 197–206.

[35] A. Clement, E. L. Wong, L. Alvisi, M. Dahlin, and M. Marchetti, "Making byzantine fault tolerant systems tolerate byzantine faults.," in *NSDI*, vol. 9, 2009, pp. 153–168.

[36] G. S. Veronese, M. Correia, A. N. Bessani, and L. C. Lung, “Spin one’s wheels? byzantine fault tolerance with a spinning primary,” in *2009 28th IEEE International Symposium on Reliable Distributed Systems*, IEEE, 2009, pp. 135–144.

[37] F. Borran and A. Schiper, “Brief announcement: A leader-free byzantine consensus algorithm,” in *International Symposium on Distributed Computing*, Springer, 2009, pp. 479–480.

[38] P.-L. Aublin, S. B. Mokhtar, and V. Quéma, “Rbft: Redundant byzantine fault tolerance,” in *2013 IEEE 33rd International Conference on Distributed Computing Systems*, IEEE, 2013, pp. 297–306.

[39] M. Treaster, “A survey of fault-tolerance and fault-recovery techniques in parallel systems,” *arXiv preprint cs/0501002*, 2005.

[40] L. Jaulmes, M. Casas, M. Moretó, E. Ayguadé, J. Labarta, and M. Valero, “Exploiting asynchrony from exact forward recovery for due in iterative solvers,” in *SC ’15: Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis*, 2015, pp. 1–12. DOI: 10.1145/2807591.2807599.

[41] G. R. Lundquist, V. Mohan, and K. W. Hamlen, “Searching for software diversity: Attaining artificial diversity through program synthesis,” in *Proceedings of the 2016 New Security Paradigms Workshop*, 2016, pp. 80–91.

[42] R. Wartell, V. Mohan, K. W. Hamlen, and Z. Lin, “Binary stirring: Self-randomizing instruction addresses of legacy x86 binary code,” in *Proceedings of the 2012 ACM conference on Computer and communications security*, 2012, pp. 157–168.

[43] J. Hiser, A. Nguyen-Tuong, M. Co, M. Hall, and J. W. Davidson, “Ilr: Where’d my gadgets go?” In *2012 IEEE Symposium on Security and Privacy*, IEEE, 2012, pp. 571–585.

[44] G. S. Kc, A. D. Keromytis, and V. Prevelakis, “Countering code-injection attacks with instruction-set randomization,” in *Proceedings of the 10th ACM conference on Computer and communications security*, 2003, pp. 272–280.

[45] E. Shioji, Y. Kawakoya, M. Iwamura, and T. Hariu, “Code shredding: Byte-granular randomization of program layout for detecting code-reuse attacks,” in *Proceedings of the 28th annual computer security applications conference*, 2012, pp. 309–318.

[46] L. V. Davi, A. Dmitrienko, S. Nürnberger, and A.-R. Sadeghi, “Gadge me if you can: Secure and efficient ad-hoc instruction-level randomization for x86 and arm,” in *Proceedings of the 8th ACM SIGSAC symposium on Information, computer and communications security*, 2013, pp. 299–310.

[47] P. Larsen, A. Homescu, S. Brunthaler, and M. Franz, “Sok: Automated software diversity,” in *2014 IEEE Symposium on Security and Privacy*, IEEE, 2014, pp. 276–291.

[48] J. S. Mertoguno, R. M. Craven, M. S. Mickelson, and D. P. Koller, “A physics-based strategy for cyber resilience of cps,” in *Autonomous Systems: Sensors, Processing, and Security for Vehicles and Infrastructure 2019*, International Society for Optics and Photonics, vol. 11009, 2019, 110090E.

[49] W. Gurecky, *Pyreactor model*, 2018. [Online]. Available: https://github.com/wgurecky/pyReactor.git.

[50] R. Ayo-Imoru and A. Cilliers, “A survey of the state of condition-based maintenance (cbm) in the nuclear power industry,” *Annals of Nuclear Energy*, vol. 112, pp. 177–188, 2018.

[51] R. Ayo-Imoru and A. Cilliers, “Continuous machine learning for abnormality identification to aid condition-based maintenance in nuclear power plant,” *Annals of Nuclear Energy*, vol. 118, pp. 61–70, 2018.

[52] H. Hashemian, “Applying online monitoring for nuclear power plant instrumentation and control,” *IEEE Transactions on Nuclear Science*, vol. 57, no. 5, pp. 2872–2878, 2010.

[53] J. Coble, P. Ramuhalli, R. Meyer, and H. Hashemian, “Online sensor calibration assessment in nuclear power systems,” *IEEE Instrumentation & Measurement Magazine*, vol. 16, no. 3, pp. 32–37, 2013.

[54] P. Ramuhalli, J. Coble, and B. Shumaker, “Robust online monitoring technologies for nuclear power plant sensors,” *Transactions*, vol. 118, no. 1, pp. 280–283, 2018.

[55] V. Ankel, S. Pantopoulou, M. Weathered, D. Lisowski, A. Cilliers, and A. Heifetz, “Monitoring of thermal mixing tee sensors with lstm neural networks,” in *12th Nuclear Plant Instrumentation, Control and Human-Machine Interface Technologies (NPICHMIT 2021)*, ANS, 2021, pp. 313–323.

[56] V. Ankel, S. Pantopoulou, M. Weathered, D. Lisowski, A. Cilliers, and A. Heifetz, “One-step ahead prediction of thermal mixing tee sensors with long short term memory (lstm) neural networks,” Argonne National Lab.(ANL), Argonne, IL (United States), Tech. Rep., 2020.

[57] S. Pantopoulou, P. L. Lagari, C. H. Townsend, and L. H. Tsoukalas, “Data-based defense-in-depth of critical systems,” in *International Conference on Dynamic Data Driven Application Systems*, Springer, 2020, pp. 283–290.

[58] S. Pantopoulou, M. Pantopoulou, and L. H. Tsoukalas, "Secure decision making and inference in critical systems," in *2021 12th International Conference on Information, Intelligence, Systems and Applications (IISA)*, IEEE, 2021.

[59] F. Darema, "Grid computing and beyond: The context of dynamic data driven applications systems," *Proceedings of the IEEE*, vol. 93, no. 3, pp. 692–697, 2005.

[60] H. A. Van Den Berg, "Occam's razor: From ockham's via moderna to modern data science," *Science progress*, vol. 101, no. 3, pp. 261–272, 2018.

[61] R. Smith, K. Friston, and C. Whyte, "A step-by-step tutorial on active inference and its application to empirical data," 2021.

VITA

Styliani Pantopoulou was born on August 14^{th} 1994, in Athens, Greece. She obtained her Engineering Diploma from the department of "Electrical and Computer Engineering" of the University of Patras. She entered the School of Nuclear Engineering of Purdue University as a visiting scholar in September of 2018, and next year she proceeded to pursue a Master's degree. For the past year, she has been a visiting student in Argonne National Laboratory (ANL). In addition to Greek (native) and English, she speaks basic French.