Mirage Theory: A Deception Approach to Intrusion Detection in Process Control Networks

Julian L. Rrushi[1] and Kyoung-Don Kang
Department of Computer Science
State University of New York
P.O. Box 6000, Binghamton, NY 13902-6000
USA

jrrushi@cs.binghamton.edu / kang@cs.binghamton.edu

ABSTRACT

Process control systems interact with transducers and actuators via exchanges of electrical signals. This form of communication involves analog-to-digital and digital-to-analog conversion, since information is sent across two totally different spaces, namely discrete and continuous spaces. In this paper we propose mirage theory, which is a deception approach to intrusion detection in process control networks. Mirage theory is founded on two factors, namely real-time simulation of physical phenomena and exploitation of information conversion as a barrier to prevent an adversary from detecting deception. Mirage theory uses such concealed simulation to convey information and indicators to adversaries in order to pilot their target selection process towards simulated or emulated physical processes and equipment, thereby causing them to take specific actions that will contribute to the detection of their intrusion. In this paper we provide a thorough discussion of mirage theory, and analyze and quantify its deception effects via application of signal detection theory.

1.0 INTRODUCTION

Computer network attacks on process control systems are known to have potential for causing physical damage to the underlying physical infrastructures [15]. In this paper we address the problem of detecting known and unknown computer network intrusions in process control networks [26]. We provide a discussion of mirage theory, i.e. a novel deception approach that we have derived from military deception (MILDEC) [30] and its applications [32]. In [30], MILDEC is defined as those actions executed to deliberately mislead adversary decision makers as to friendly military capabilities, intentions, and operations, thereby causing the adversary to take specific actions or inactions that will contribute to the accomplishment of the friendly mission. Mirage theory is comprised of actions that are devised to deliberately mislead an adversary as to digitally controlled physical processes and equipment such as nuclear power plants and radar units, thereby causing the adversary to take specific actions that will contribute to the detection of his/her intrusion in process control networks.

Deception means in MILDEC are grouped into three categories, namely physical means, technical means, and administrative means. Examples of physical means include dummy and decoy equipment and devices, tactical actions, movement of military forces, etc. Examples of technical means include emission of chemical or biological odors, emission of radiation, reflection of energy, computers, etc., while examples of administrative means include techniques to convey or deny physical evidence. Mirage theory employs mainly technical deception means, namely emission of deceptive network traffic along with computers and computer networks. Mirage theory relies to a large degree on a MILDEC concept that is referred to as a display. Displays are simulation, disguising, and/or portrayal of friendly objects, units, or capabilities that may not exist, but are made to appear so. In this regard mirage theory employs computers or computer clusters to simulate or emulate the presence of physical processes and equipment.

[1] Julian L. Rrushi is a doctoral student of the Università degli Studi di Milano, Italy, and a research scholar at SUNY, USA.

The ultimate goal of mirage theory is to cause an adversary to target computer systems that monitor and control simulated or emulated physical processes via simulated or emulated sensors and actuators. Actions taken by a deceived adversary are thereafter leveraged to detect and characterize his/her malicious activity. Our research on mirage theory was inspired by a lesson that we drew from history, namely the Operation Fortitude South, which is a MILDEC application that was conducted in the second world war [32]. The allied invasion of German occupied territory of France was preceded by a strategic plan whose codename was Operation Fortitude South, which was made by the allies to deceive the command of German military into believing that the allies would attack from Pas de Calais rather than from Normandy. In addition to large intelligence operations such as espionage and controlled leaks of information through diplomatic channels, this plan included also the creation and deployment of a special electronic unit that was called the 5th wireless group.

This group used some newly developed transmitters to generate radio communications based on preprogrammed and especially written scripts. These radio communications contained conversations that are typical to military assault operations. As the German military in France had few aerial reconnaissance capabilities left, eavesdropping on radio communications was the principal mechanism that they could use to determine movements of allied troops. The Operation Fortitude South was highly successful to a degree that Adolf Hitler concentrated a large number of military units, including Panzer tank units, in Pas de Calais. Mirage theory exploits similar concepts, namely the adversary's reliance on analysis of intercepted network data to derive the presence and characteristics of physical targets, and the lack of means to verify that intercepted traffic is indeed generated by existing physical targets.

In this paper the terms physical system and continuous space are used interchangeably. The terms cyber-physical system and digitally controlled physical system are also used interchangeably. The terms communicating finite state machine and automaton are used interchangeably as well. In this paper the term network traffic is used to refer to process control protocol frames.

2.0 RELATED WORK

The application of deception techniques from conventional warfare, as detailed under the light of specific case studies of military conflicts drawn from history, for improving the security of computer systems and networks has been explored by Rowe and Rothstein in [22, 23]. Rowe and Rothstein analyze historical military operations like Operation Mincemeat [20], which took place during the second world war, to illustrate a set of principles and mechanisms that are used for an effective tactical deception in conventional warfare. The authors then evaluate the applicability of the said principles and mechanisms to the invention of defensive deceptive capabilities for computer systems and networks. Mirage theory moves along the line of Rowe and Rothstein's research as it is a direct application of MILDEC to the defense of process control systems and networks. Furthermore, as written in the previous section, while researching on mirage theory we carefully analyzed a historical conventional warfare operation, namely Operation Fortitude South.

Honeypots, i.e. closely monitored information system resources that serve as network decoys [25, 27], and mirage theory have a few features in common to some extent, including distraction of adversaries from valuable attack targets, and leverage of deception for intrusion detection [8, 14]. Nevertheless, mirage theory is fundamentally different. Honeypots are passive and just stand by to receiving network connections. Thus, they have no normal activity. Mirage theory is exactly the opposite, in the sense that its main strength lies in normal system and network activity. Honeypots usually simulate services that are more vulnerable than their production counterpart in order to lure attackers. Mirage theory fights for making deceptive process control networks, systems, and simulated or emulated physical processes or equipment as much undistinguishable from their production counterparts as possible.

The deception capabilities of honeypots are placed within the boundaries of a computer system or network, and hence fall within network access visibility. Mirage theory develops deceptive capabilities at a layer which is not reachable through a network access to a target process control network. Rowe and Rothstein in [22, 23] indicate that honeypots are not in line with an important principle of conventional warfare, namely that deception should be integrated with operations. Their thesis is that deceptive tactics are more effective on real systems. In fact Holz and Raynal show in [9] several techniques to detect honeypots by capturing technical details that are characteristic for virtual execution environments, and hence different than in real ones. In mirage theory the process control systems and networks are all genuine. Furthermore, they are deployed and configured in such a way that they can be used to smoothly monitor and control an existing physical system.

The research of Yuill et al. provided in [33] and mirage theory leverage concepts that are similar to some degree, namely honeyfiles and deceptive program variables in cyber-physical mappings, respectively. Honeyfiles are bait files intended for hackers to access. Honeyfiles reside in servers, which generate intrusion alerts if a honefile is accessed. Honeyfiles are intended to be no different than other normal files. Thus, for an adversary to detect a honeyfile, he/she has to open it, an action that results in the detection of the adversary's presence. A cyber-physical mapping is a one-to-one correspondence between program variables, which hold logical or continuous I/O values in the random access memory (RAM) of control systems, and physical process parameters or parameters that characterize the operation of physical equipment. While a file is mapped to regions of secondary storage by the operating system, a deceptive variable is mapped to a parameter related to a physical process or equipment, which in fact are all simulated or emulated.

Mirage theory employs deceptive protocol frames to make deceptive variables appear no different than their genuine counterpart. For an adversary to detect a deceptive variable, he/she has to access it either locally or over a process control network, an action that is used to detect the intrusion. In [21], Rowe attacks the problem of logical consistency in deception. He explores automated methods which track assertions that are made up to a certain point in time along with their effects, and thereafter identify the possible consistent deceptions that may be conducted next in order. In mirage theory we attack the same problem, but do so in a way that differs from the approach followed by Rowe in [21] due to our different levels of intervention. Row works mainly at the operating system level, while from the logical consistency perspective in mirage theory we focus mainly on physical system phenomena. We employ large systems of differential equations to feed an adversary with a consistent view of the internal dynamics of physical processes and equipment at any point in time.

The end objective is what mirage theory has in common with cognitive hacking [4], reflexive control theory [28], and perception management process [12]. Cognitive hacking is basically manipulation of the perception of technology users. Reflexive control is a warfare theory that has been studied in the former Soviet Union and later on in Russia for a very long time. Reflective control theory is comprised of methods for conveying to a subject especially prepared information in order to incline him/her to voluntarily make a predetermined decision. The US approximate counterpart of reflective control theory is the perception management process. Perception management is comprised of actions that convey and/or deny selected information and indicators to foreign audiences in order to influence them. Mirage theory seeks to exploit the adversary's mind, namely his/her perception of a defined cyber-physical system. Mirage theory actively conveys information and indicators to an adversary for the purpose of piloting his/her target selection process towards simulated or emulated physical processes and equipment.

3.0 MIRAGE THEORY

3.1 Conducting Defensive Deception for Intrusion Detection

As illustrated in the top part of Figure 1, the interactions between sensing or actuating devices and edge control systems take place via application of electrical signals with certain characteristics. In a typical sensing activity sensors, i.e. transducers, measure physical phenomena and report continuous values by generating analog values, i.e. voltages or currents. For instance, incore detectors in a nuclear reactor measure neutron flux. Incore detectors apply electrical signals that are proportional to neutron population in the reactor core. Neutron flux measurements that are conveyed by these electrical signals are processed by computer systems, which together form a neutron monitoring system. For measurement values to be processed by computer systems, the corresponding electrical signals are periodically sampled and converted into discrete numerical values via analog-to-digital conversion integrated circuits [7].

Edge control systems actuate physical equipment also by applying electrical signals. Discrete numerical values in a computer system are converted into analog values via digital-to-analog conversion integrated circuits. For instance, an edge control system may set the rotational speed of an AC induction motor by controlling the applied voltage frequency. In the actual context we see two spaces, namely one in which values are in a continuous form and another in which values are in a discrete form. In this work we refer to these spaces as the continuous space and the discrete space, respectively. As also depicted in Figure 1, analog-to-digital and digital-to-analog conversion integrated circuits may be thought of as a boundary between the continuous and discrete spaces. In fact it is in these integrated circuits that information changes form from continuous to discrete and vice versa.

The basis of mirage theory is formed by leverage of the boundary between continuous and discrete spaces, leverage of how the presence of a continuous space is reflected on a corresponding discrete space, and simulation or emulation of physical processes and physical equipment. A computer network attack provides an adversary with access that may extend to a whole discrete space. Nevertheless, due to physical limits there are no feasible ways for an adversary to gain visibility over a continuous space through a computer network attack. In other words, a computer network attack won't enable an adversary to virtually move beyond the analog-to-digital and digital-to-analog conversion integrated circuits. Consequently an adversary cannot verify whether input electrical signals are indeed applied by existing sensing devices, nor can he/she verify whether output electrical signals indeed reach an existing actuating device.

Referring to the bottom part of Figure 1, in mirage theory we generate measurement values in a digital form via computer simulation or emulation, and hence employ digital-to-analog converters to generate the electrical signals that correspond to these digital values. What edge control systems receive in input comprises a series of analog values as if such values were generated by existing sensing devices. Edge control systems then convert these logical values into digital values via analog-to-digital converters, and thereafter process them. Since it is after the conversion to a digital form that the values in question become accessible to an adversary, the previously described manipulation based on computer simulation or emulation is totally transparent. Similarly, we employ analog-to-digital converters to receive analog values from edge control systems. Resulting digital values thereafter may be fed to a computer system in order to simulate or emulate the effects that electrical signals that are applied by edge control systems would have had on existing physical processes or equipment.

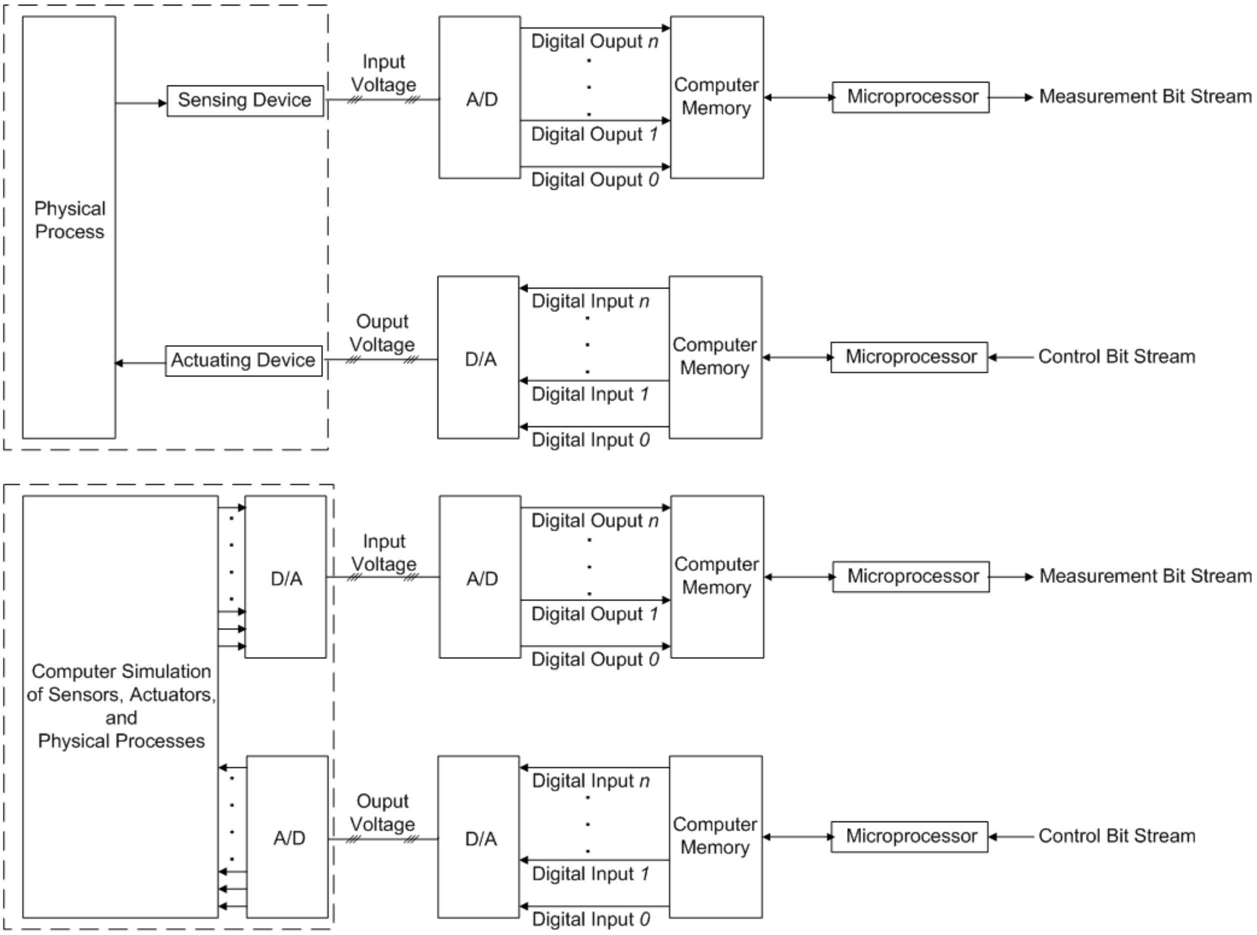

Figure 1: Boundary between continuous and discrete spaces exploited in mirage theory to camouflage computer simulation or emulation of sensors, actuators, and physical processes

When values that are generated by an edge control system are converted into an analog form, an adversary entirely loses visibility on them. As a consequence the previously described manipulation is, again, transparent. From a network access perspective the presence of physical processes and physical equipment is derived from protocol frames that flow over a process control network. For instance, in a nuclear power plant control network a protocol frame such as the one depicted in Figure 2 denotes the presence of an electric motor that generates rotational motion, a ball screw that translates this rotational motion into a linear motion, and a control rod that is inserted or withdrawn via the linear motion in question. The reconnaissance that normally precedes initiation of physical damage through a computer network attack over a target process control network includes analyses of protocol frames that flow over this network.

By employing techniques such as applied regression analysis [24] an adversary analyzes data that are conveyed by protocol frames, with the result being an identification of the configuration of a target cyber-physical system along with equipment and physical process specifications indicated in Figure 2. Not only is the information derived through these reconnaissance analyses indicative of the presence of physical processes and equipment, but it also details them. The interaction between a simulated or emulated continuous space and a genuine discrete space, as depicted in the bottom part of Figure 1, is characterized by network traffic that guides the reconnaissance analyses conducted by an adversary into identification of physical processes and equipment, which in reality are all simulated or emulated, along with the computer systems that control and monitor them.

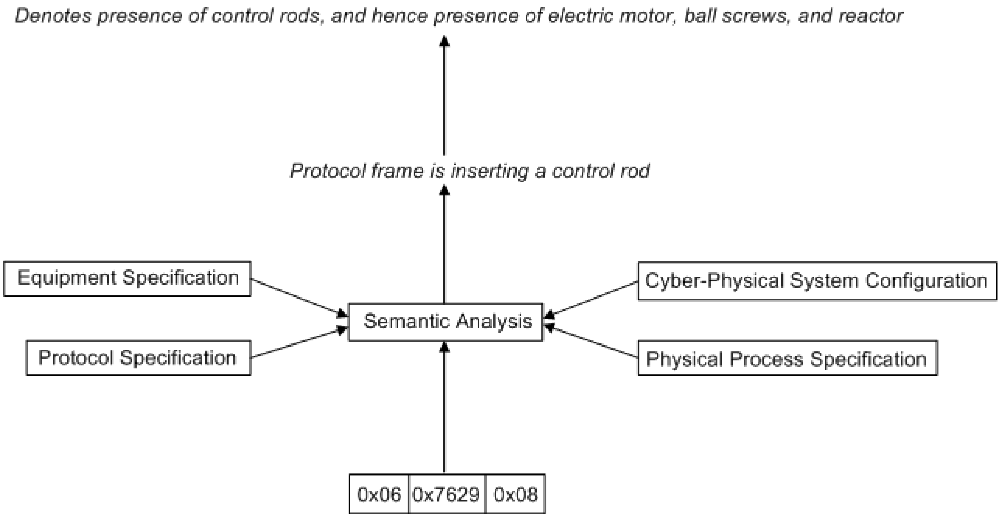
Denotes presence of control rods, and hence presence of electric motor, ball screws, and reactor
Protocol frame is inserting a control rod
Equipment Specification
Protocol Specification
Semantic Analysis
Cyber-Physical System Configuration
Physical Process Specification
0x06
0x7629
0x08

Figure 2: A protocol frame that is indicative of the presence of physical equipment and a physical process

As written previously, in mirage theory only continuous space is simulated or emulated. Discrete space is intentionally chosen to be genuine in its entirety. In mirage theory process control systems and networks are deployed and configured as if they were to monitor and control a real physical process through real sensors and actuators. The computer code that is to be run in these control systems also needs to have been written for control systems that are intended to monitor and control a real physical process through real sensors and actuators. A clean way of constructing the discrete space of an application of mirage theory is to deploy a replica of the discrete space of a real cyber-physical system in production. The ultimate goal of mirage theory is to deceive an adversary into targeting a simulated or emulated physical process and/or equipment by actively attacking a control system over a process control network in the discrete space of an application of mirage theory itself.

3.2 Real-Time Deceptive Event Generation

Taking into account that an adversary cannot cross the boundary between continuous and discrete spaces, a possible option that he/she may explore for detecting deception consists of analyses of protocol frames having as an objective to identify inconsistencies that may be caused by errors in the simulation or emulation of physical processes and equipment. Even tiny imperfections in the simulation or emulation of physical processes and equipment are reflected on network data in the discrete space, and hence have potential for enabling an adversary to catch inconsistencies by employing advanced analysis techniques such as those based on applied mathematics. In this section we discuss our research on faithfull simulation or emulation of physical processes and equipment, namely continuous simulation and traffic mirroring.

3.2.1 Continuous Simulation of Physical Processes and Equipment

In mirage theory we need to simulate a continuous space in real-time in order to faithfully mimic the appearance of existing digitally controlled physical systems such as nuclear power plants or radar units. We deem that a viable simulation technique for such purpose is continuous simulation [3] as, in addition to having the necessity to simulate continuous systems, we are also interested in modeling the internal dynamics within each one of these continuous systems in order to generate a faithful simulation. As a matter of fact physical processes and physical equipment such as, for instance, nuclear fission and AC induction motors, respectively, are continuous systems. Their state variables, which are modeled as functions, change continuously over time. Furthermore, the rates of change of these variables, which are modeled as derivatives, change continuously over time as well.

Continuous simulation models of a continuous space are comprised of ordinary or partial differential equations that characterize the behaviour, i.e. internal dynamics, of physical processes and equipment at any point in time. The continuous simulation itself is conducted by solving these differential equations via analytical methods, i.e. explicit formulas, or numerical analysis in computer clusters behind the boundary between continuous and discrete spaces, as indicated in the bottom part of Figure 1. With regard to detection of protocol frames that are part of a computer network attack, we organize this process as in anomaly intrusion detection, namely in a learning phase and a monitoring phase. We model each process control system that is involved in deceptive communications, i.e. protocol frames that are generated as a result of simulation of a continuous space, as a communicating finite state machine [29]. Both input that causes these automata to transition from one state to another and output that is generated when state transitions take place are protocol frames.

An example of the said communicating finite state machines is given in Figure 3. The purpose of a learning phase is to thoroughly contruct these automata, while in a successive monitoring phase these automata are used to distinguish between deceptive protocol frames and possible malicious protocol frames. In a learning phase we define a sequence of process set points, which when entered in a human machine interface (HMI) [26] will emulate the digitally controlled operation of a physical system. Recall that all or part of the continous space is simulated. Examples of set points in a nuclear power plant include discrete values of the level of water whithin a reactor pressure vessel, logical values, i.e. on or off, for starting or stopping a defined water pump, logical values for opening or closing a main steam isolation valve, logical values for opening or closing the circuit breaker of an electric generator, discrete values of the terminal speed of a turbine, etc.

The said set points are then issued from a computer system that would normally be used by a system operator, i.e. the computer system that is associated with a HMI and that is depicted in the upper left corner of Figure 4, with the result being a realistic operation of a digitally controlled physical system as if the continuous space was real rather than simulated. As the simulated continuous space is operated, we sniff all protocol frames that flow over the process control network. The protocols frames that a control system receives and transmits are modeled as input and output, respectively, in the associated communicating finite state machine. Each one of the protocol frames in input causes a state transition. Nevertheless, the automaton may transition from one state to another state even though it has received no protocol frames in input. Similarly, it may transition from one state to another state while producing no protocol frames in output.

Overall, a communicating finite state machine acts as a sequence detector with respect to the network traffic that is received from or transmitted by the control system that it models. Although we coded a few communicating finite state machines in the concurrent hierarchical state machine language system [17], and those mainly in the form of a proof of concept prototype, to our knowledge the activity of constructing these automaton-based sequence detectors can be automated. In a monitoring phase we employ a computer system, which would normally be used by a system operator, to issue the sequence of process set points that were defined and used in the learning phase. Here, again, the process of mimicking a system operator, i.e. issuing set points, can be easily fully automated. From a network access perspective, which is what an adversary sees and perceives, network traffic that flows over the process control network is indicative of the presence of a physical system that is being operated digitally by system operators.

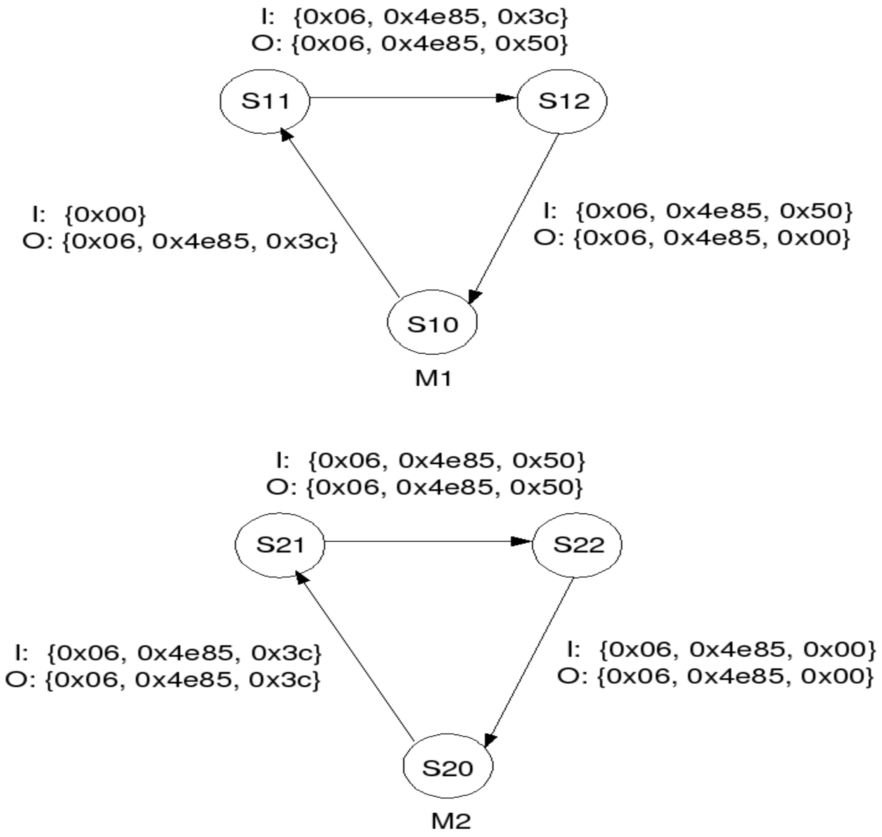

Figure 3: Excerpt from two communicating finite state machines that model two individual control systems in a process control network

Communicating finite state machines are embedded in a network protocol frame inspection tool. As protocol frames flow accross a process control network, the detection tool inspects them to determine the network identifiers of the source and destination nodes, i.e. for example IP addresses in the case the process control protocol runs over TCP/IP, and thereafter submits them individually to the corresponding automata. If these sequence detectors recognize the protocol frames submitted to them, then they simply transition to another state, and from there the inspection loop keeps running. We do allow for small variations of values in the data field of protocol frames in order to deal with various sources of small errors, such as for example jitter effects on analog to digital and digital to analog converters. If sequence detectors do not recognize a given protocol frame, then it means that they are processing a protocol frame that was not generated in the learning phase, therefore the inspection tool raises an intrusion alert. We deem that if learning is conducted correctly in the learning phase, then a computer network attack is manifested as one or more extra protocol frames, i.e. protocol frames that do not result to have been generated in the learning phase.

Later on in this paper we discuss an attack-defense model in which an adversary conducts a loss of cooling attack on a power plant. The attack is materialized by transmitting a protocol frame that writes a malicious value to a program variable, which is mapped to applied voltage frequency, in a target control system. According to our thesis the said malicious protocol frame is not generated in the learning phase, therefore sequence detectors in the inspection tool won't recognize it. We see a series of pros and cons in employing continuous simulation to simulate a continuous space. Continuous simulation allows for interactivity with an adversary, in the sense that an adversary can generate malicious frames and verify their negative impact on his/her targets. This is useful in the case security officers are interested in letting an adversary do some progress, in order to extract a better characterization of his/her attack. Furthermore, there are cases in which it may be necessary to allow for more than a single unseen protocol frame in order to have sufficient confidence that a computer network attack is actually taking place.

With continuous simulation we have no reliance on existing digitally controlled physical systems. We can independently deploy an entire process control network and associate it with a simulated continuous space. We can, for example, deploy a large number of dummy power plants that are based on continuous simulation, while not needing the presence of an existing power plant. On the other hand, it is challenging to conduct a continuous simulation of a nuclear power plant in real-time. The issue is that differential equations that model complex systems such as nuclear power plants are too complex to be solved analytically. Their solutions are to be obtained via numerical analysis, which requires considerable computing resources for making the internal dynamics within a simulated continous space appear as taking place in real-time. Furthermore, numerical analysis generally produces numerically approximated solutions of differential equations, a fact which in theory may open a window for an adversary to look for imperfections that indicate the simulation nature of a target continuous space.

In practice though we don't see approximate solutions of differential equations as an issue as long as they stay within an acceptable degree of accuracy range. The rationality behind our assumption lies in the fact that, to the best of our knowledge, analyses of network traffic that is induced by an existing continuous space do not reflect any absolute perfections. Thus, the challenge that an adversary has to face does not consist in how to differentiate between a perfect view and an imperfect view of a target continuous space. An example of a source of imperfection in our context is the conversion of information. There are no ideal analog to digital converters. Analog to digital conversion of information is characterized by unavoidable errors such as quantization errors, aperture errors, non-linearity errors when applicable, etc. Similarly, digital to analog conversion of information is not ideal either. Furthermore, the said errors are mostly random. Thus, the challenge that an adversary has to face consists in how to differentiate between randomly imperfect views of a target continuous space.

3.2.2 Deceptive Emulation via Network Traffic Mirroring

Traffic mirroring is a technique for emulating a continuous space. Thus, it is an alternative to continuous simulation. The main idea behind this emulation technique is to sniff network traffic in a process control network in production, i.e. a process control network that is used to operate an existing physical system, and thereafter replay the said traffic in a deceptive process control network, i.e. a process control network that is deployed as part of a mirage theory application. A schematic diagram of how traffic mirroring may be applied is depicted in Figure 4. As of the time that the research reported in this paper was conducted, we had no access for experimentation purposes to a real world power plant. Due do this limitation we couldn't implement network traffic mirroring and analyze it in practice, consequently we describe it in this paper as a theoretical approach to emulation of a continuous space. In addition, the concepts that accompany our proposal of the said emulation technique, including network sniffing systems, traffic mirroring network, and inspection tool, are purely hypothetical.

The deceptive process control network, which is depicted in the left part of Figure 4, is the replica of a process control network in production, which is depicted in the right part of Figure 4. Network sniffing systems are deployed in all network segments of the process control network in production. These systems are equipped with two network interfaces, namely one attached to a segment of the process control network in production, and another one attached to a network that is used to propagate protocol frames to the deceptive process control network. We refer to the latter as a traffic mirroring network. The network sniffing systems operate at the data link layer, therefore they are not assigned a network layer address on the network interface that is attached to a segment of the process control network in production.

Furthermore, for obvious reasons we need the network sniffing systems not to reveal their existence. For this purpose the network stack of a network sniffing system may be modified such that it doesn't respond to any queries, and hence evade network discovery tools.

The network sniffing systems are assigned a network layer address on the network interface that is attached to the traffic mirroring network. As protocol frames flow across the process control network in production, they are intercepted by the network sniffing systems, which in turn associate a timestamp with them and send them to the general purpose computers depicted in the right part of Figure 4. More precisely, protocol frames that convey set points and that are transmitted by the HMI system in the process control network in production are sent to the replica of a HMI system in the deceptive process control network. At emulation time the replica of a HMI system replays these protocol frames into the deceptive process control network according to their associated timestamps. Protocol frames that convey sensor data are sent to those computer systems that are equipped with digital to analog converters. In Figure 4, for example, to each one of the flowmeters corresponds a computer system, which at emulation time converts sensor data from digital to analog according to their timestamps. Thus, in general the said systems emulate transducers that are in the process control network in production.

Sniffing and replaying protocol frames that convey set points and sensor data is sufficient for emulating a continuous space. As set points are transmitted from the replica of a HMI system into the deceptive process control network, they trigger exchanges of protocol frames between control systems, with the result being a transmission of protocol frames that convey actuator control data. Upon reception of the said data, actuators in the deceptive process control network generate electrical signals, which are received from computer systems in the traffic mirroring network that are equipped with digital to analog converters. Once these computer systems derive the digital form of an analog value, they ignore it as their only objective is to mimic the reception of physical parameter changes by physical equipment. As computer systems that emulate transducers generate electrical signals, which correspond to the analog form of sensor data that are received from network sniffing systems, these signals are converted into digital sensor data and then are transmitted over the deceptive process control network.

The resulting protocol frames, which convey the said sensor data, trigger exchanges of protocol frames between control systems in the deceptive process control network. Some of these sensor data are also sent to the replica of a HMI system in the form of process status updates. Thus, overall, what we expect to achieve with synchronized replays of set points and sensor data is a logical mirror. The said mirror is intended to reflect protocol frames that flow across the process control network into the deceptive process control network, with a shift Δt with respect to time. With regard to detection of malicious protocol frames, we propose an adaptation of the approach used in the case continuous simulation is used to simulate a continuous space. More precisely, we propose to model each control system in the process control network in production as a communicating finite state machine like those depicted in Figure 3.

The emulation and intrusion detection activities are organized into intervals of time Δt. During each one of these intervals the network sniffing systems intercept protocol frames as they flow across the process control network in production. If we assume that the activity of constructing automata can be automated, then the said protocol frames are used in an automated construction of communicating finite state machines, which model control systems in the process control network in production. The interval of time Δt is chosen sufficiently long for the automated contruction of communicating finite state machines to be complete by the end of Δt itself. As in the case of continuous simulation, we may integrate these automata into a network protocol frame inspection tool. In the next interval of time Δt, as the protocol frames that convey set points and sensor data are replayed in the deceptive process control network, the inspection tool intercepts all protocol frames that flow accross the deceptive process control network.

The inspection tool then submits these protocol frames to the communicating finite state machines, which act as sequence detectors. As in the case of continuous simulation, we need to tolerate small variations of values in the data field of protocol frames in order to deal with small errors induced by factors such as jitter effects. Furthermore, considering that the control systems in the deceptive process control network are replicas of the control systems in the process control network in production, we expect the said sequence detectors to be valid for the former as well as for the latter. As in the case of continuous simulation, if the sequence detectors recognize the protocol frames that are submitted to them, then they transition to from ones state to another. If not, we deem that the network traffic flowing across the process control network in production is diverging from the network traffic flowing across the deceptive process control network.

We deem that one of the most relevant advantages of employing traffic mirroring to emulate a continuous space is its potential for producing a highly accurate copy of the network traffic that flows in a process control network in production. On the other hand, traffic mirroring does not allow for interactivity with an adversary. With traffic mirroring we have to rely on an existing digitally controlled physical system. That said, a logical connection with a process control network in production may be a disadvantage as much as an advantage. While we cannot independently deploy an entire process control network and associate it with an emulated continuous space, divergences captured by the sequence detectors may be an indication of intrusion in the process control network in production.

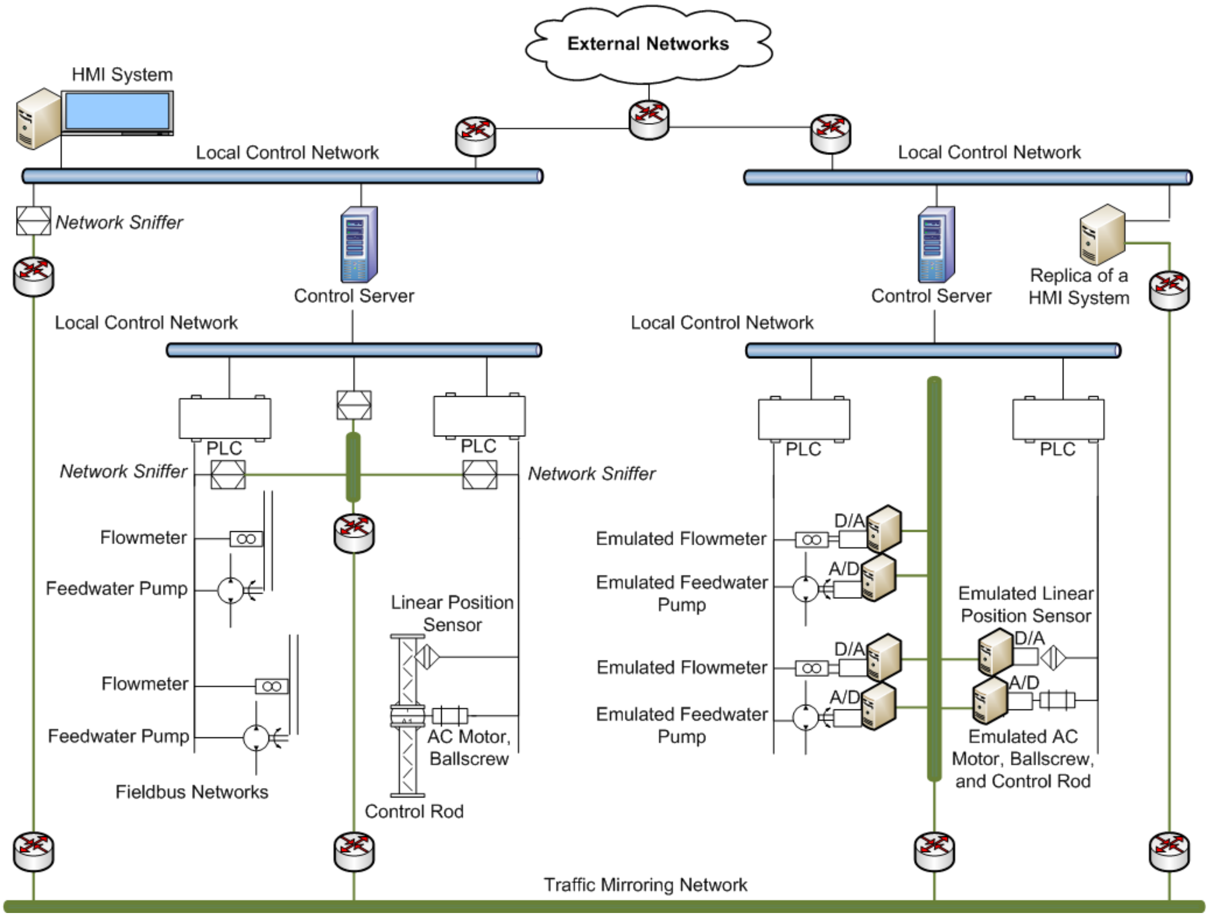

Figure 4: Schematic diagram of an emulation of a continuous space via traffic mirroring

3.3 Analysis of Deception Capabilities in Mirage Theory

We now provide a practical attack-defense model in order to analyze the deception capabilities of mirage theory. In this model an adversary attempts a loss of cooling attack on a digitally controlled power plant that is based on a boiling water reactor design [11]. In the said power plant a number of water pumps feed water into the reactor core. The injected water picks up the heat produced by nuclear fission, and thereafter is transformed into steam. Steam is then directed through pipes to spin the shaft of a turbine that is connected to an electric generator. In addition to being transformed into steam, the injected water serves to also cool nuclear fuel in the reactor. Let water be fed into a reactor core by motor-driven water pumps that are controlled by programmable logic controllers (PLCs) [5]. In this example attack-defense model the adversary conducts a computer network attack on these PLCs in order to cause physical damage to the motors that they control, and hence prevent the motor-driven water pumps from functioning.

Let the PLCs that control motor-driven water pumps in this model be part of a distributed control system (DCS) [26] whose communications are based on Modbus TCP protocol [19]. For an adversary to affect the operation of an electric motor, he/she should preliminarily identify that part of a cyber-physical mapping which relates program variables in a target PLC with physical parameters that characterize the operation of the associated electric motor. This is due to the fact that an adversary can affect these physical parameters only by modifying the corresponding program variables. We provide a statistical technique that may be used for such purpose. We thereafter show how mirage theory deceives an adversary by generating data that guide his/her data analysis, and hence quantify empirically the deception capabilities exhibited in this attack-defense model by analyzing under the light of signal detection theory [13, 18] the reaction of mirage theory.

3.3.1 Reconnaissance for a Computer Network Attack on an Electric Motor

We have found persistent statistical relations between certain program variables in RAM of PLCs, depending on what physical parameters they are mapped to. More precisely, we have developed a statistical technique which, under the condition that a linear relation between program variables of interest is in place, employs the degree of linear association as measured by a linear correlation coefficient to identify such variables of interest and hence reveal their Modbus addresses [24]. In our attack-defense model an adversary applies the said technique to conduct a network inertial attack on an alternating current (AC) induction motor [10], which in turn drives a water pump. An inertial attack is conducted by speeding up or slowing down heavy equipment at high rates. It is reported to have potential for forcing heavy equipment to fail as in general heavy equipment is not tolerant to abrupt speed changes [16].

For an adversary to be able to launch an inertial attack, he/she needs to find out what program variable in the RAM of a target PLC is mapped to applied voltage frequency, which is a physical parameter that is used to set the actual rotational speed of an AC induction motor controlled by the PLC in question. Let this PLC use a continuous sensor, namely a battery powered stroboscopic tachometer, to measure the rotational speed of the AC induction rotor. The adversary is assumed to have acquired access to a target process control network, and hence can reconstruct the content of variables in the RAM of a target PLC by sniffing protocol frames or by sending Modbus queries to the target PLC. The first step for the adversary is to find out whether a program variable of interest in a target PLC is linearly correlated with other program variables that are acquirable via network sniffing or scanning. Referring to our model, an adversary is interested in finding out whether a program variable that is mapped to applied voltage frequency in a target PLC, which controls an AC induction motor, can be linearly associated with a variable, say actual rotational speed, that is available in the discrete space.

Let γ, ω, τ, p, δ, l, and ν denote applied voltage frequency, actual rotational speed, synchronous speed, number of poles, magnetic slip, load, and nameplate speed at full load of an AC induction motor, respectively. In laboratory settings we study a PLC-controlled AC induction motor that is characterized by values of physical parameters p, δ, τ, ν, and l, chosen randomly among those available. These values are given in Table 1. Our thesis is that, although the internal architecture and configuration of a randomly chosen AC induction motor may be totally different than the internal architecture and configuration of an AC induction motor controlled by the target PLC, some program variables that are mapped to physical parameters such as γ and ω exhibit hidden but calculable statistical relations. Furthermore, these statistical relations are persistent among electric motors, even though their internal architectures and configurations may differ to large degrees. We show that the said thesis holds for a linear relation between γ and ω, a fact that is leveraged by the adversary in our attack-defense model to find out what variable in the target PLC is mapped to ω.

p	ω	τ	l	ν	δ	γ
4	1246.3	1884.0	0.9	1175.4	637.7	62.8
4	1255.6	1977.0	0.9	1175.4	721.4	65.9
4	1236.1	1782.0	0.9	1175.4	545.9	59.4
4	1218.7	1608.0	0.9	1175.4	389.3	53.6
4	1205.8	1479.0	0.9	1175.4	273.2	49.3
4	1178.8	1209.0	0.9	1175.4	30.2	40.3
4	1203.7	1458.0	0.9	1175.4	254.3	48.6
4	1222.6	1647.0	0.9	1175.4	424.4	54.9
4	1197.7	1398.0	0.9	1175.4	200.3	46.6
4	1186.0	1281.0	0.9	1175.4	95.0	42.7

Table 1: A sample of values of physical parameters that characterize the operation of an AC induction motor studied in laboratory settings

Taking into account that the actual rotational speed ω of the AC induction motor as reported by the tachometer is a continuous input value, by referring to the Modbus specification the adversary derives that the target PLC uses an input register variable to hold ω in its RAM. Furthermore, since the applied voltage frequency γ is a continuous output value, the adversary derives that the target PLC uses a holding register variable to hold γ. The reconnaissance analysis proceeds with assessing whether the program variable of interest and program variables that the program variable of interest may be potentially linearly correlated with, i.e. the holding register variable mapped to γ and the input register variable mapped to ω in our attack-defense model, follow a Gaussian distribution [6]. We use ModScan [31] to acquire values of the holding register variable mapped to γ and values of the input register variable mapped to ω over a defined period of time from the testing PLC that controls the AC induction motor in laboratory settings.

These values are given in Table 1 along with the values of physical parameters p, δ, τ, ν, and l mentioned previously. A series of Modbus scanner tools such as ModScan [2, 31] have been developed for security assessments, and some of them are publicly available. These tools enable a security analyst or adversary to acquire the values of discrete input variables, coil variables, input register variables, and holding register variables, which are stored in the RAM of a target PLC. Furthermore, most of these tools also enable a security analyst or adversary to send attack frames that attempt to write to logical or continuous variables in a target PLC once he/she has identified the cyber-physical mapping. Modbus variables in a target PLC may be scanned several times, a process that normally produces a large set of data. The challenge consists in analyzing these data to identify a cyber-physical mapping, and it is this challenge that is addressed by the proposed statistical technique.

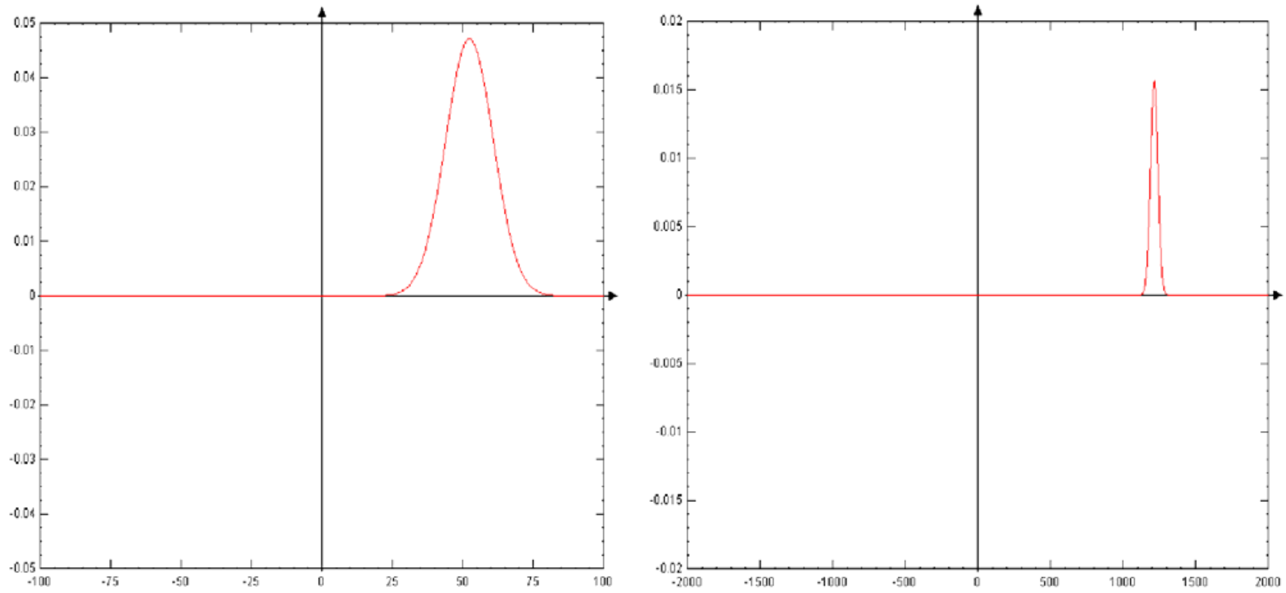

Figure 5: Normal density curves for applied voltage frequency γ and actual motor rotational speed ω, left and right respectively, in which the standard deviation of γ is 8.46751 and the standard deviation of ω is 25.40254

Let $\bar{\gamma}$ and $\bar{\omega}$ denote the mean average of γ and the mean average of ω, respectively. $\bar{\gamma}$ and $\bar{\omega}$ are estimated through the formulae shown below:

$$\bar{\gamma} = \left(\frac{\sum_{i=1}^{10} \gamma}{10}\right) = 52.41$$

and

$$\bar{\omega} = \left(\frac{\sum_{i=1}^{10} \omega}{10}\right) = 1215.13$$

The normal density curves for γ and ω are depicted in Figure 5.

Under the condition that the program variables that are being analyzed follow a Gaussian distribution, the adversary applies the least squares regression method [1] to estimate the regression line that characterizes the linear relationship between these program variables. More precisely, he/she estimates the intercept and slope of this regression line, and hence builds the regression line itself along with a scatter plot displaying values of the program variables under investigation. He/she then uses the regression line and the scatter plot to characterize the degree of linear association between these program variables, and hence estimates their linear regression coefficient. In our specific attack-defense model the adversary pilots his analysis towards quantification of a linear relation between the holding register variable that is mapped to γ and the input register variable that is mapped to ω. More precisely, he/she is interested in their degree of linear association as measured by a linear correlation coefficient that we denote with r.

In the analysis of the data that are shown in Table 1 we consider γ as a dependent variable, and ω as an independent variable. Note that we are not assuming causality between these two program variables in this order. Let a and b denote intercept and slope, respectively, in the linear relation between γ and ω. The linear relation between γ and ω is modeled by the equation below:

$$\gamma = a + b\,\omega$$

where a and b are estimated via least squares regression as shown in the following equations:

$$b = \left(\frac{\sum_{i=1}^{10}((\omega_i-\bar{\omega})(\gamma_i-\bar{\gamma}))}{\sum_{i=1}^{10}(\omega_i-\bar{\omega})^2}\right) = 0.33$$

$$\bar{\gamma} = a + b\,\bar{\omega} \Rightarrow a = \bar{\gamma} - b\,\bar{\omega} = -352.6$$

Thus, the linear relation between γ and ω is modeled as:

$$\gamma = (-352.6) + 0.33\,\omega$$

The scatter plot and linear regression line for this linear relation are depicted in Figure 6. The linear correlation coefficient r measuring the degree of linear association between γ and ω is estimated by the following equation:

$$r = \left(\sqrt{\frac{\sum_{i=1}^{10}(\hat{\gamma}_i-\bar{\gamma})^2}{\sum_{i=1}^{10}}\,(\gamma_i-\bar{\gamma})^2}\right) = 1$$

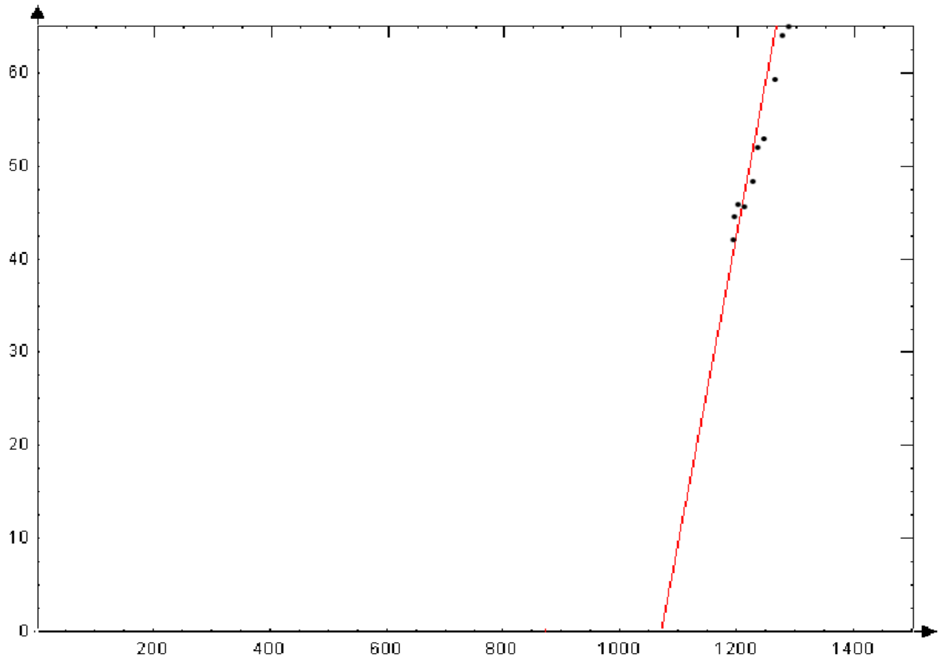

Figure 6: Scatter plot and linear regression line for the statistical relation between γ and ω

Armed with a quantification of the correlation coefficient r that measures the degree of linear association between γ and ω, an adversary reconstructs the content of program variables that are stored in the RAM of a target PLC. Table 2 presents a sample of these data. For the sake of clarity, *IR* and *HR* stand for input register variable and holding register variable, respectively. The Modbus address of each reconstructed variable is given in square brackets. Recall that the adversary has already derived that it is an input register variable and a continuous output value in the RAM of the target PLC that is mapped to ω and γ, respectively.

IR[16]	*IR[53]*	*IR[18]*	*IR[69]*	*HR[19685]*	*HR[20008]*	*HR[18610]*	*HR[65530]*
702.5	1884	1205.3	685.2	63.9	36.5	42.1	49.6
803.8	1977	903.9	679.2	55.4	39.2	41.6	52.4
901.8	1782	1306.9	722.4	55.8	38.3	45.2	62.3
904.1	1608	1004.8	763.2	67.3	45.8	48.6	60.1
1004.7	1884	1407.8	735.6	57.8	48.1	46.3	59.2
903.1	1977	1409.4	796.8	58.1	49.3	51.4	57.3
1004.9	1782	1408.3	868.8	61.8	51.5	57.4	57.9
809.6	1608	1598.3	817.2	48.9	58.3	53.1	61.4
1208.8	1782	1203.9	890.2	38.9	61.8	59.1	63.8
803.5	1608	957.5	945.6	48.6	47.5	63.8	65.0

Table 2: Excerpt from the data set acquired through ModScan from a target PLC

At this point the adversary estimates the degree of linear association between all program variables, which is, he/she estimates their linear regression coefficients. These estimations are given in Table 3. Let r_{lab} be the regression coefficient between a program variable that is mapped to a physical parameter of interest and a program variable that is mapped to another physical parameter, which in turn is linearly correlated with the physical parameter of interest, as preliminarily estimated in laboratory settings. The identification of a program variable of interest in a target PLC takes place when defined program variables are found to have a regression coefficient that is equal to r_{lab}. Thus, in our specific attack-defense model the holding register variable that is mapped to applied voltage frequency γ is the one whose linear association with an input register variable, which is presumably mapped to the actual rotational speed ω, has a degree that is equal to 1.

	IR[16]	*IR[53]*	*IR[18]*	*IR[69]*
HR[19685]	-0.41	0.16	-0.05	-0.54
HR[20008]	0.64	-0.36	0.43	0.71
HR[18610]	0.4	-0.54	0.05	0.99
HR[65530]	0.49	-0.66	0.1	0.72

Table 3: Measurements of the degree of linear association between holding register variables and input register variables that were scanned from the memory of a target PLC

From referring to Table 3 we see that the Modbus address of the holding register variable that is mapped to applied voltage frequency γ is 18610. Furthermore, the Modbus address of the input register variable that is mapped to actual rotational speed is 69. In fact the correlation coefficient of these two program variables is slightly less than 1, namely 0.99, since a series of roundings of numbers were performed during the mathematical estimations.

3.3.2 Empirical Quantification of Deception Effects

The reconnaissance for a loss of cooling attack produces information such as presence of motor-driven water pumps that are controlled over a reachable discrete space, presence of physical processes such as nuclear fission, evaporation, condensation, etc., in a target continuous space, address or name of a program variable in the RAM of a target PLC that is mapped to applied voltage frequency, the IP address of a target PLC, the TCP port used by a slave Modbus application in a target PLC to receive and send data, etc. This information has a direct influence on the adversary's decision making process with regard to target selection and attack engineering. Mirage theory intervenes during the reconnaissance process with the objective of interfering with the adversary's decision-making process. Such interference is conducted by manipulating the adversary's perception of a target continuous space through the lenses of an associated discrete space.

We analyze the effects of mirage theory on the adversary's decision-making process under the light of signal detection theory. Signal detection theory is a method to characterize and quantify the ability of a subject to discern between signal and noise. If we draw three parallel lines between signal detection theory and mirage theory, namely a parallel line between signal and presence of existing physical processes and equipment, another parallel line between noise and simulated or emulated physical processes and equipment, and another parallel line between subject and adversary, then we end up with applying signal detection theory to characterize and quantify the ability of an adversary to discern between an existing continuous space and a simulated or emulated continuous space.

The subjects in our analysis comprised a team of students who were taught the internals of process control networks, Modbus TCP, PLCs, and AC induction motors. Most importantly, the student team was taught how to apply the statistical technique discussed in the previous subsection for the purpose of identifying in a large set of data a holding register variable that is mapped to applied voltage frequency. Thus, the student team took the role of the adversaries. The complete protocol frames exchanged, i.e. network packets that comprise data link, IP, and TCP headers, and application data units [19], were sniffed and gathered in a data set, which was presented to members of the student team individually. The student team was told that only one of the holding register variables is mapped to an applied voltage frequency. Each member of the student team was asked to reconstruct the values of program variables from ordered series of protocol frames printed on paper, and from there identify an existing target of a loss of cooling attack by the means of estimation of degrees of linear association.

This test revealed that during the reconnaissance for a loss of cooling attack an adversary is subject to what in signal detection theory is defined as external noise. The most common form of external noise met during this test was that the degree of linear association between several program variables was estimated to be r_{ab}. In other words, more than two program variables were found to be linearly associated to the same degree. External noise was found also during the application of other optional or complementary reconnaissance techniques. For instance, an adversary may identify variables of interest by comparing the values of reconstructed program variables to typical values of parameters related to physical processes or equipment. Thus, when conducting reconnaissance for a loss of cooling attack an adversary may assess whether values of each holding register variable are typical for a parameter such as applied voltage frequency.

The external noise in this case is that values of several holding register variables may be typical for applied voltage frequency. Despite the affects of external noise, the said test showed that the adversary's decision-making process with regard to target selection is subject to a relatively low uncertainty. Figure 7 depicts the internal response probability of occurrence (POC) curves that characterize the said uncertainty.

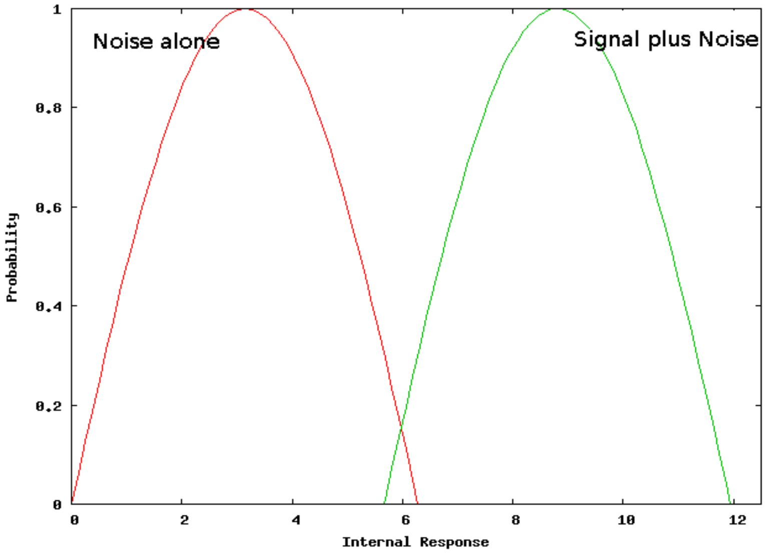

Figure 7: POC curves for a PLC that controls a motor-driven water pump, as estimated during a simulated loss of cooling attack

The horizontal axis in Figure 7 represents information that motivates an adversary to take the decision that the signal is present, while the vertical axis represents the frequency of the occurrence of a defined amount of such information. The POC curves in Figure 7 show that the signal strength is high and the amount of noise, both external and internal, is low. Consequently the overlap of these two POC curves is small, while their spread is reduced. The discriminability index derived from the separation and spread of the two POC curves in Figure 7 has a value around $d' = 5.6$. Thus, with such a high discriminability index an existing continuous space is considerably discriminable from a simulated or emulated continuous space. These estimations overall indicate that adversaries who have expertise in process control can identify correctly the target of a loss of cooling attack with a high rate of correct selections and a low rate of wrong selections.

The receiver operating characteristic (ROC) curve, which is plotted with hit rate on the vertical axis and false alarms rate on the horizontal axis, for a discriminability index $d' = 5.6$ goes up to the upper left corner converging with a straight line that intersects the vertical axis at a value of 100% and is parallel with the horizontal axis. This ROC curve indicates that an adversary's decision making process with regard to target selection is characterized by a large number of hits and just a few false alarms. The previous test was conducted again, but this time the student team was made subject to deception effects that were induced by an application of mirage theory. The target process control network included a number of PLCs that controlled emulated motor-driven water pumps. As in the former test, in the latter test we sniffed a set of complete protocol frames that were sent over the process control network, and thereafter presented them to each member of the student team individually.

Emulation of motor-driven water pumps resulted in generation of deceptive protocol frames that drastically incremented the uncertainty to which the adversary's decision-making process is exposed with regard to target selection. These deceptive communications acted as what in signal detection theory is defined as internal noise. As in signal detection theory, a subject, i.e. an adversary, has little or no control over the internal noise that is emitted during a decision making process. The POC curves that characterize the uncertainty that was induced by deceptive protocol frames are depicted in Figure 8.

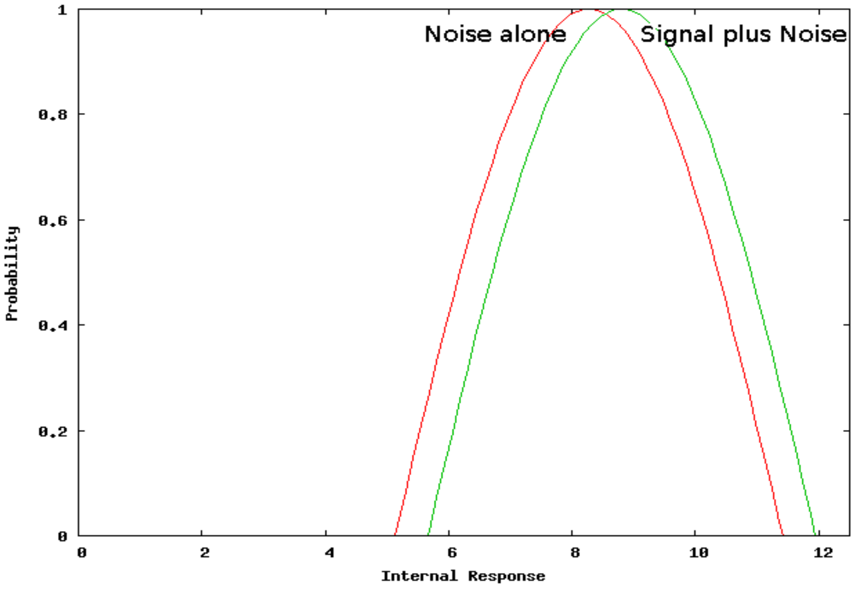

Figure 8: POC curves that characterize the uncertainty under which adversaries identify the target of a loss of cooling attack in the attack-defense model given in this section

The internal noise makes an existing continuous space hardly discriminable from a simulated or emulated continuous space. Under the effects of mirage theory the strength of the internal response is lowered. More precisely, mirage theory lowers the discriminability index of the reconnaissance for a loss of cooling attack from $d' = 5.6$ to $d' = 0.45$. The effects of mirage theory in terms of hit rates and false alarm rates are provided by the ROC curve depicted in Figure 9.

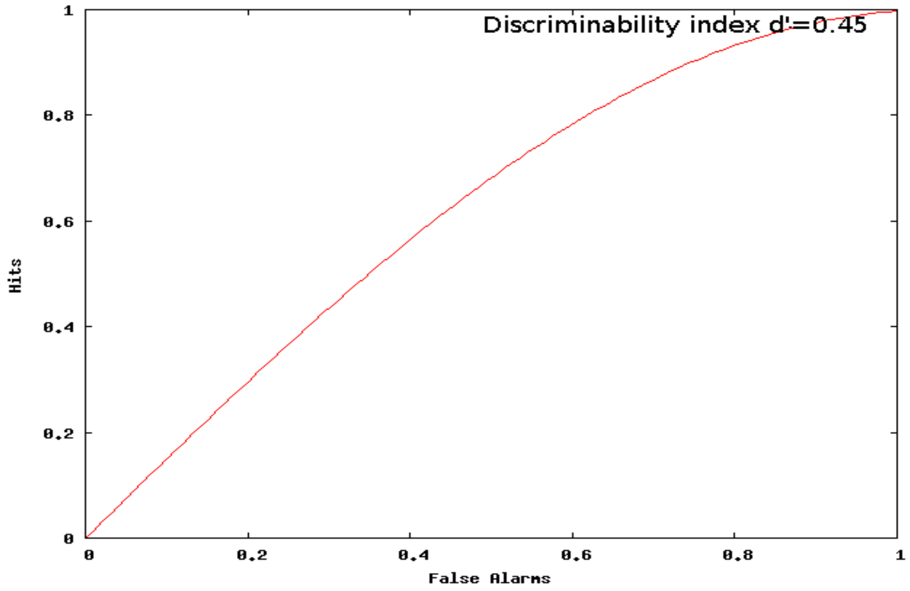

Figure 9: ROC curve that corresponds to the POC curves of Figure 8

4.0 CONCLUSION

In this paper we discussed our research on mirage theory, which is a control system specific approach to intrusion detection that we derived from MILDEC and deep studies of historical military operations. We provided a description of the boundary between continuous and discrete spaces, which forms the basis of mirage theory from the concealment perspective. We then explained the internals of mirage theory as built upon the said boundary. We provided an overview of how an adversary conducts reconnaissance that leads to target identification and attack engineering, and showed how mirage theory exploits the adversary's perception of a target cyber-physical system to cause him/her to voluntarily make the decision of targeting a simulated or emulated physical process or equipment.

In mirage theory the deceptive event generation process is performed via simulation or emulation of a continuous space. In this regard we discussed continuous simulation and traffic mirroring along with techniques for detecting intrusions once an adversary is deceived into attacking a simulated or emulated target. We then developed a practical attack-defense model in which we analyzed and quantified the deception capabilities of mirage theory. More precisely, we applied signal detection theory to empirically quantify the deception effects of mirage theory on the adversary's mind. Our evaluations of mirage theory indicate that the said theory is highly deceptive, and hence a viable approach to intrusion detection in process controls networks.

5.0 ACKNOWLEDGEMENT

Julian L. Rrushi wishes to express his most sincere gratitude and appreciation to Professor Roy H. Campbell of the University of Illinois at Urbana-Champaign for advising him during the development of this research.

6.0 REFERENCES

[1] Bjorck, A. (1996). Numerical Methods for Least Squares Problems. SIAM, 1996.

[2] Bristow, M. (2008). ModScan: A SCADA MODBUS Network Scanner. DEFCON 16.

[3] Cellier, F.E., & Kofman, E. (2006). Continuous System Simulation. Springer.

[4] Cybenko, G., Giani, A., & Thompson, P. (2002). Cognitive Hacking: A Battle for the Mind. IEEE Computer, 35(8), pp. 50-56.

[5] Erickson. K. (2005). Programmable Logic Controllers: An Emphasis on Design and Application. Dogwood Valley Press.

[6] Herrnstein, R., & Murray, C. (1994). The Bell Curve: Intelligence and Class Structure in American Life. Free Press.

[7] Hoeschele, D.F. (1994). Analog-to-Digital and Digital-to-Analog Conversion Techniques. 2nd Edition, Wiley-Interscience.

[8] Holz, T., Goebel, J., & Hektor J. (2006). Advanced Honeypot-based Intrusion Detection. ;login:, 31, 6, USENIX.

[9] Holz, T., & Raynal, F. (2005). Detecting Honeypots and Other Suspicious Environments. Proceedings of the 6th IEEE Information Assurance Workshop, United States Military Academy, West Point, NY, USA.

[10] Hughes, A. (2005). Electric Motors & Drives. Newnes.

[11] International Electrotechnical Commission. (2007). IEC 61505: Nuclear Reactor Instrumentation - Boiling Water Reactors (BWR) - Stability Monitoring. Distributed through American National Standards Institute.

[12] Jones, C.S. (1999). The Perception Management Process. Military Review, The Professional Journal of the U.S. Army. http://www.au.af.mil/au/awc/awcgate/milreview/jones_perception.pdf

[13] Kay, S.M (1998). Fundamentals of Statistical Signal Processing, Volume 2: Detection Theory, Prentice Hall Publishing.

[14] Kreibich, C., & Crowcroft, J. (2003). Honeycomb - Creating Intrusion Detection Signatures Using Honeypots. Proceedings of the 2nd Workshop on Hot Topics in Networks, Cambridge, MA USA.

[15] Krutz, R.L. (2006). Securing SCADA Systems. Wiley Publishing.

[16] Larsen, J. (2008). Breakage. Blackhat Federal 2008.

[17] Lucas, P.J., & Riccardi, F. Concurrent Hierarchical State Machine. http://chsm.sourceforge.org

[18] Marcum, J.I. (1947). A Statistical Theory of Target Detection by Pulsed Radar. U.S. Air Force Project RAND.

[19] Modbus Organization. (2004). Modbus Application Protocol Specification. http://www.modbus.org/docs/Modbus_Application_Protocol_V1_1b.pdf

[20] Montagu, E. (1954). The Man Who Never Was. Lippincott Publishing House.

[21] Rowe, N.C. (2007). Finding Logically Consistent Resource-Deception Plans for Defense in Cyberspace. Proceedings of the 3rd International Symposium on Security in Networks and Distributed Systems, Niagara Falls, Ontario, Canada.

[22] Rowe, N.C., & Rothstein, H. (2003). Deception for Defense of Information Systems: Analogies from Conventional Warfare. Department of Computer Science and Defense Analysis, U.S. Naval Postgraduate School, USA. http://www.au.af.mil/au/awc/awcgate/nps/mildec.htm

[23] Rowe, N.C., & Rothstein, H. (2004). Two Taxonomies of Deception for Attacks on Information Systems. Journal of Information Warfare, 3(2), pp. 27-39.

[24] Rrushi, J.L. & Kang, K.D. (2008). CyberRadar: A Regression Analysis Approach to the Identification of Cyber-Physical Mappings in Process Control Systems. Proceedings of the IEEE/ACM Workshop on Embedded Systems Security, Atlanta, Georgia, USA.

[25] Spitzner, L. (2002). Honeypots: Tracking Hackers. Addisson-Wesley Professional.

[26] Stouffer, K., Falco, J., & Scarfone, K. (2007). Guide to Industrial Control Systems Security. NIST Special Publication 800-82. http://csrc.nist.gov/publications/drafts/800-82/2nd-Draft-SP800-82-clean.pdf

[27] The Honeynet Project. (2004). Know Your Enemy: Learning about Security Threats. 2nd Edition, Addison-Wesley Professional.

[28] Thomas, T.L. (2004). Russia's Reflexive Control Theory and the Military. Journal of Slavic Military Studies 17: pp. 237-256.

[29] Tropper, C., & Boukerche, A. (1993). Parallel Simulation of Communicating Finite State Machines. Proceedings of the Workshop on Parallel and Distributed Simulation, pp. 143-150, San Diego, CA, USA.

[30] U.S. Joint Chiefs of Staff. (2006). Military Deception. Joint Publication 3-13.4 http://www.dtic.mil/doctrine/jel/new_pubs/jp3_13_4.pdf

[31] WinTECH Software. ModScan Tool. http://www.win-tech.com/html/modscan32.htm

[32] Young, M., & Stamp, R. (1989). Trojan Horses - Deception Operations in the Second World War. Bodley Head, London, UK.

[33] Yuill, J., Zappe, M., Denning, D., & Freer, F. (2004). Honeyfiles: Deceptive Files for Intrusion Detection. Proceedings of the 5th IEEE Workshop on Information Assurance, U.S. Military Academy, West Point, NY, USA.