A Specification-Based Intrusion Prevention System for Malicious Payloads

Aaron W. Werth

Ph.D. Student in Computer Engineering - Cybersecurity

University of Alabama in Huntsville

SCADA Systems

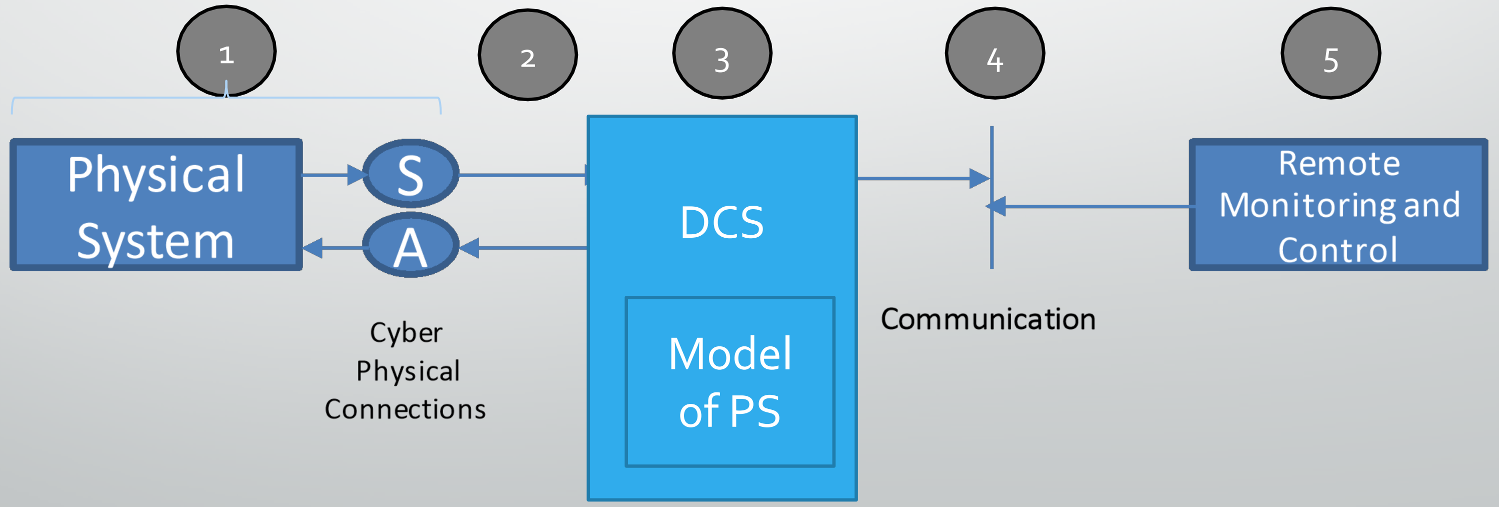
Abstract

Abstract. In this work, a control/command analysis-based intrusion prevention system (IPS) is proposed. This IPS will examine incoming command packets and programs that are destined for a PLC interacting with a physical process. The IPS consists of a module that examines the packets that would alter settings or actuators and incorporates a model of the physical process to aid in predicting the effect of processing the command and specifically whether a safety violation would occur for critical variables in the physical system. Essentially, a simulation of both the model of the physical system and a process running a copy of the ladder logic of the real PLC is performed in the module. Also, uploaded programs will be evaluated to determine whether the programs would cause a safety violation. Previous research has studied making predictions based on the payloads of packets where cumbersome specifications must be developed by a human expert for the model of the physical system and safety conditions. This work seeks to eliminate or minimize the amount of specifications to be developed by a human through system identification and machine learning to allow the IPS to be more generic and deployable. Another contribution of this work is a broader and more generic understanding of the threat model that causes unsafe or inefficient consequences. The Accuracy in prediction and latency in analysis are metrics used when evaluating the results in this work.

Keywords: SCADA, Cybersecurity, Industrial Control Systems, Intrusion Detection, Semantic-based, specification-based, malware, Ladder Logic Bomb

Command Analysis-based Intrusion Detection (Prevention) Systems

- Specification consists of (1) Model of physical system. (2) Acceptable values for states and outputs. (3) Efficient behavior.
- Command packets evaluated to determine what effect they would have on the system based on the specification.



Motivation and Background for Command-based Analysis IPS

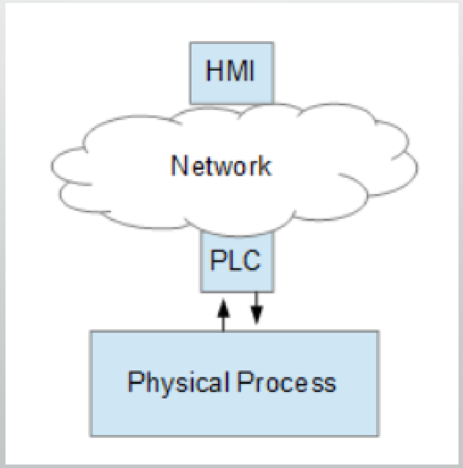
- **Motivation** for this work comes from three real-world examples:
  - Stuxnet
  - Maroochi Attack
  - Attacks on the Ukrainian Power Grid
- **Common theme:** Examples involve a *malicious command or program* that places the physical system in an *undesired state – whether unsafe or inefficient*.
- These examples may differ in how the command or program is introduced to the SCADA network.
- **The goal**: Predict the behavior of the system caused by the program/command by analyzing the *payload* of the packet. Essentially this behavior is detected before it is allowed to occur.

Motivation and Background (Continued)

- IDSs for detecting malicious behaviors on networks have been researched extensively in the literature. IDSs use certain info:
  - Increased/decreased volume of traffic,
  - larger/smaller packets,
  - changes in interarrival time between packets
  - other characteristics of the network traffic (ports and IP addresses used, suspicious patterns of operating states in the network protocols).
- However, threat may appear normal on the network yet malicious since the packets contain commands or other instructions with harmful effect.
- Instructions may be *benign* also, depending on the *current state* of the cyber-physical system and how the instructions are used.
- An IDS detecting the maliciousness of packets appearing normal on the network is *complimentary* to IDSs that do detect malicious network behavior.

Problem Formulation

- The conceptual SCADA system as described by [8] is used.
- The following is assumed concerning the SCADA system:
  - A PLC or multiple PLCs directly interface with the physical process or system.
  - A network facilitates communication between the PLC (s) and the HMI or Centralized Computer.



Problem Formulation: Modules of SCADA systems

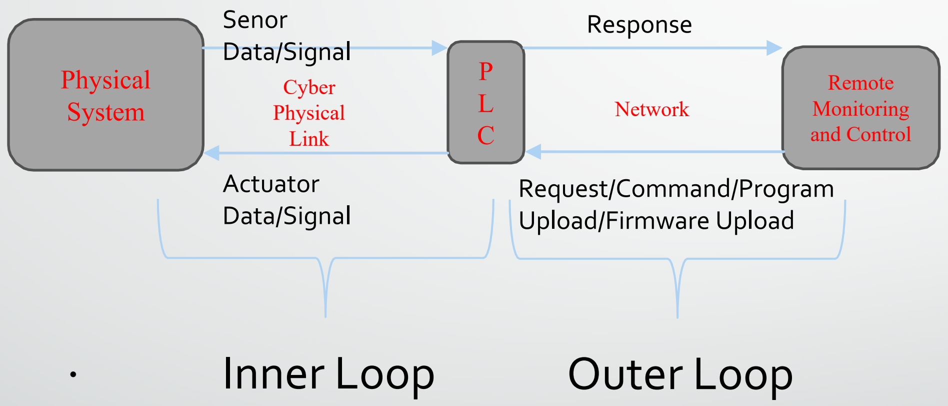
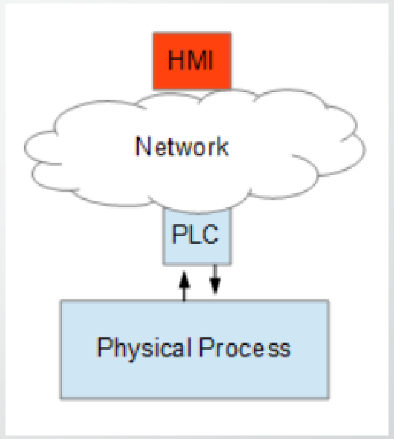


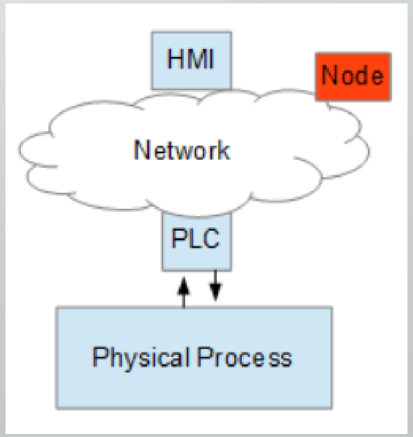
Figure 1. SCADA System with Five main Components as described by [8]

- Outer loop - Attack surface is on this loop, which consists of the network.
- Inner loop - virtually no attack surface since it consists of wires and is transmitted vary low-level information or signals.
- Proposed IPS will examine the Commands and Program Uploads to determine if they are acceptable for the PLC to execute.

Threat Model

- **Assumption:** an adversary has compromised a node on the network considered a *trusted system* or that a *foreign malicious node* has access to the network.
- Threat consists of the advisory's ability to send command packets or upload malicious programs to the PLCs .
- Interesting because of threat's impact on physical system.
- Threat cannot hide its ultimate objective even if the threat is hidden to many other types of counter measures and IDS.
- The proposed IDS will examine the payload to determine what is the effect and whether that is unsafe or inefficient.
- Threat is assumed to have reached an advanced stage of the ICS Cyber kill chain [9].





Threat Model - Consequences

- The threat model of this work in terms of consequences: (1) safety violations and (2) inefficiencies for the SCADA system [10].
- The threat model that can induce above consequences
    - (1) command injections attacks born of the SCADA network to the PLC that direct the PLC to change settings, states, or output of the PLC,
    - (2) Programmation/Ladder Logic Updates – logic and time bombs
    - (3) Firmware Updates*
- * Possibly out of scope of this work. Require a full virtual machine to investigate changes of underlying software. May have much greater complexity to evaluate such treats.

[10] Huang, Y. L., Cárdenas, A. A., Amin, S., Lin, Z. S., Tsai, H. Y., & Sastry, S. (2009). Understanding the physical and economic consequences of attacks on control systems. *International Journal of Critical Infrastructure Protection*, *2*(3), 73-83.

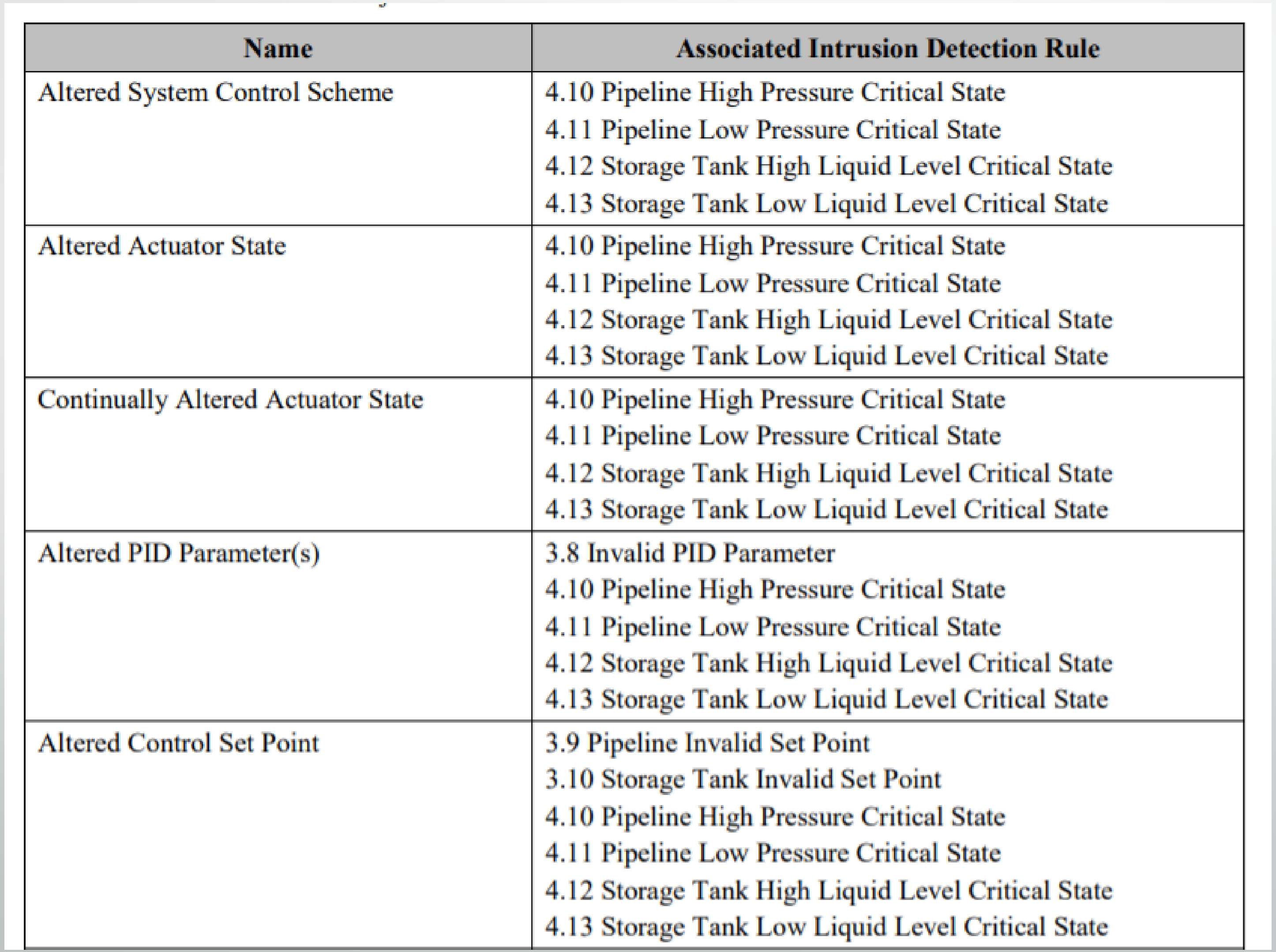
Threat Model: Three Layers of PLC Software

Layer 2 - Settings: Internal parameters or settings within the ladder logic, States of the actuator directly affecting the output

Layer 1 - Programmation: Ladder Logic Program with inputs, outputs, states, and timers

Layer 0 - Firmware: OS, Kernel, drivers for hardware, supporting software

Threat Model: Malicious Command (affecting Layer 2)



Name	Associated Intrusion Detection Rule
Altered System Control Scheme	4.10 Pipeline High Pressure Critical State 4.11 Pipeline Low Pressure Critical State 4.12 Storage Tank High Liquid Level Critical State 4.13 Storage Tank Low Liquid Level Critical State
Altered Actuator State	4.10 Pipeline High Pressure Critical State 4.11 Pipeline Low Pressure Critical State 4.12 Storage Tank High Liquid Level Critical State 4.13 Storage Tank Low Liquid Level Critical State
Continually Altered Actuator State	4.10 Pipeline High Pressure Critical State 4.11 Pipeline Low Pressure Critical State 4.12 Storage Tank High Liquid Level Critical State 4.13 Storage Tank Low Liquid Level Critical State
Altered PID Parameter(s)	3.8 Invalid PID Parameter 4.10 Pipeline High Pressure Critical State 4.11 Pipeline Low Pressure Critical State 4.12 Storage Tank High Liquid Level Critical State 4.13 Storage Tank Low Liquid Level Critical State
Altered Control Set Point	3.9 Pipeline Invalid Set Point 3.10 Storage Tank Invalid Set Point 4.10 Pipeline High Pressure Critical State 4.11 Pipeline Low Pressure Critical State 4.12 Storage Tank High Liquid Level Critical State 4.13 Storage Tank Low Liquid Level Critical State

Methods – for IPS to Detect Threat

- Predicting potential behavior of SCADA system requires a special intrusion detection system (IDS) or Intrusion prevention system (IPS).
- The IPS is software-based or hardware-based module and is a proxy between PLC and HMI.
- This IPS will perform deep-packet inspection of the packets (Modbus) and will simulate the command against a model of the physical system to determine consequences, whether unsafe or inefficient.
- Necessary to have (1) a model of the physical system, (2) specifications on safety conditions and/or efficient operation.

Methods – for IPS to Detect Threat, Modeling and Specifications

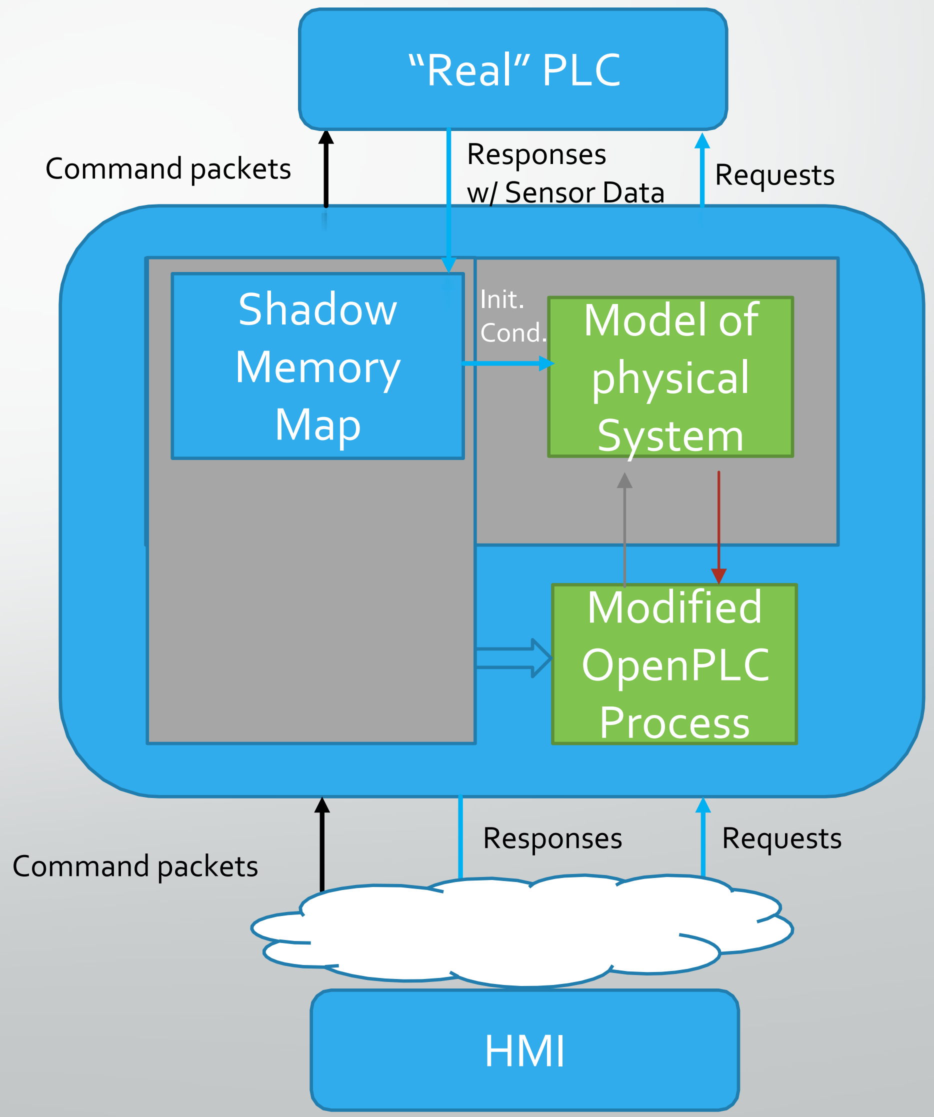
- Models of Physical systems and specifications of safety conditions either (1) a white-box, (2) grey-box, and (3) black-box.
- **White-box** models and specifications - human-defined, tedious to implement, created by an expert in the domain of engineering relevant to the given SCADA system.
- **Black-box** models and specifications - not known beforehand but must be determined from inputs and outputs of the physical system time series data to construct a model. Safety conditions may be determined in a similar manor as described in [6], which is meant to be semantically agnostic. The safety conditions may be determined from previous data to determine an acceptable range of behavior.
- **Grey-box** models and specifications are a hybrid of white-box and black-box.

IPS Module

- Module is assumed to have access to input and output data. Module performs two stages as part of its IPS Process – (1) Training and (2) Testing.
- The advantage of placing the module at the PLC is that cyberattacks that affect the network between the PLC and the centralized location are limited in their ability to affect the training data.
- On the other hand such a module at the PLC may tax the resources of the PLC and the centralized computer system may have more processing ability.

IPS Module: Design (Mainly Testing Stage)

- Many of the previously mentioned papers have an "analysis engine." to analyzie incoming commands
- Modified OpenPLC or representative Ladder logic program Process (ModOpenPLC) is used; has virtually no delay between scan cycles, only the time involved in calculating the current time cycle.
- Model of physical system "in sinc" with ModOpenPLC in terms of simulated timing.
- When packet arrives that contains command, several steps are taken.
  1) Model of physical system receives state variable info from real system for initial conditions (e.g. Water tank level.)
  2) The command to be evaluated is applied to ModOpenPLC.
  3) The simulation of the Physical System Model coupled with the Modified OpenPLC Process is run.
  4) The simulation is evalulated to determine if any "critical variables" veer outside a safe region.
  5) A decision is made by the IPS depending on whether the potential action by the command will cause harm (unsafe, inefficient)
     - If safe, process
     - Else, alert, drop, delay, etc



IPS Module: Design – Shadow Memory Map

- Request and Response Modbus (Possibly other types) packets pass through the IPS Module:
- Requests are periodically sent by the HMI. to the PLC (Through the IPS Proxy/Relay)
- PLC sends responses to HMI (Through the IPS Proxy/Relay)
- Shadow memory map can reconstruct the
  - Stored as a List of tuples:

    (Memory Address, Value),

    (Memory Address, Value),

    ....

(http://www.simplymodbus.ca/FC03.htm)

```
Read Holding Registers (FC=03)

Request
This command is requesting the content of analog output holding
registers # 40108 to
 40110 from the slave device with address 17.

11 03 006B 0003 7687

11: The Slave Address (11 hex = address17 )
03: The Function Code 3 (read Analog Output Holding Registers)
006B: The Data Address of the first register requested.
             ( 006B hex = 107 , + 40001 offset = input #40108 )
0003: The total number of registers requested. (read 3 registers
40108 to 40110)
7687: The CRC (cyclic redundancy check) for error checking.

Response

11 03 06 AE41 5652 4340 49AD

11: The Slave Address (11 hex = address17 )
03: The Function Code 3 (read Analog Output Holding Registers)
06: The number of data bytes to follow (3 registers x 2 bytes each =
6 bytes)
AE41: The contents of register 40108
5652: The contents of register 40109
4340: The contents of register 40110
49AD: The CRC (cyclic redundancy check).

 (http://www.simplymodbus.ca/FC03.htm)
```

System Identification and Machine Learning for Training Stage

- Currently using Matlab's tools to perform system identification
- Gradient decent, a technique used in machine learning, may be used in system identification to tune the parameters of models.
- Mean Square Error is also a standard method that is used for tuning parameters.

IDS Process – Stage1: Training

- Step 1 – Collecting Input and Output Data from PLC
  - Some Data allocated for Parameter Estimation. Other data allocated for Validation.
- Step 2 – Parameter Estimations of Models: Several structures of models are tested such as different transfer functions with different numbers of parameters.
- Step 3 – Validation and Selection of Best Model

IDS Process – Stage 2: Testing

- Step 4 - Monitoring, simulating, and evaluating incoming commands and programs
- Step 5 – IPS Response
  - Delay
  - Drop
  - Process
  - Alert

IDS Process – Stage 2: Testing – Step 4 for Malicious Commands

- Important commands to change registers will have specific codes in the **Modbus Protocol**:

Function Code	Function Name
5	Write Single Coil
15	Write Multiple Coils
6	Write Single Holding Register
16	Write Multiple Holding Registers

- Not all the commands would be considered harmful or malicious. The IPS will forecast the effect of the command to make a determination on whether safe or not
- Safety may be defined as a range of values for the process variables that are acceptable to Human Expert.
- Safety or reasonable behavior may also be defined based on previous training data, where
  - Max allowed is max of training data; Min allowed is min or training data OR
  - Stewart Limits: {Lmin, Lmax} – upper and lower operational limit of the process variable defined as three standard deviations from the estimated mean.

Research Hypothesis

- The proposed IPS will be able to detect the previously defined threats that involve payloads causing malicious consequences.
- The IPS will be able to neutralize the threat in a an appropriate manner that will not adversely impact the operation of the SCADA System.

Evaluation

- IPS will be evaluated in terms of its **accuracy** in predicting the output of processing a command or a program using specific metrics and also how close the Estimated time to criticality is to the actuality if the physical process will eventually enter the critical state.
- IPS will also be evaluated in terms of its **latency** to simulate hypothetical scenarios of attacks and programs.
- IPS will also be evaluated in terms of its overall vulnerability to cyberattacks given various placements of the module.
- What are the false positives and negatives.

Metrics to Evaluate Accuracy

- It is desired to compare the *prediction* of the behavior of the Physical Process due to processing the command/program against an *actual experiment* of processing the command/program.
- Time series data of the affected process variables may be compared with these metrics:
  - MAPE
  - SMAPE
  - MATLAB's methods for determining Fitness

MAPE

- Mean Absolute Percentage Error
- MAPE is defined by the following equation:

$$MAPE = \frac{100\%}{n} \sum_{t=1}^{n} \left| \frac{A_t - F_t}{A_t} \right| ,$$

- Where A is the actual value and F is the forecast value.

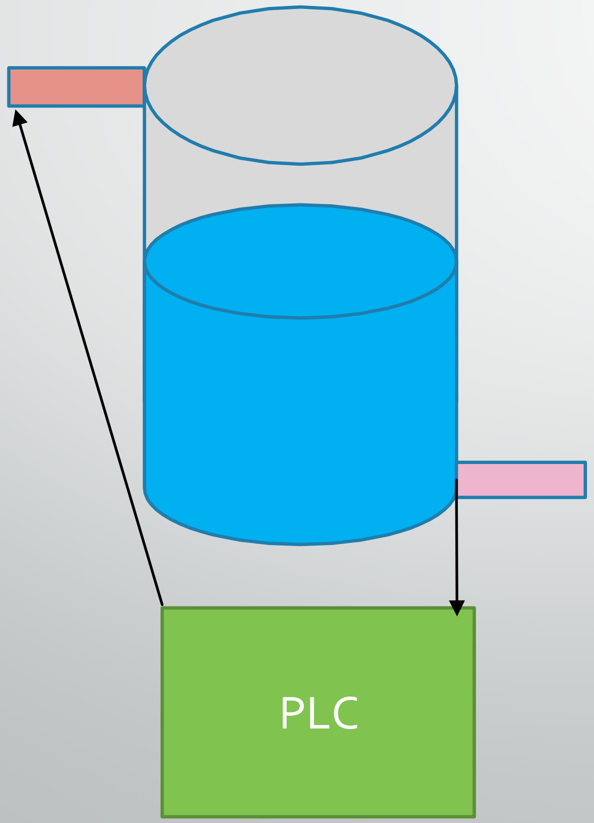
SPAPE

- Symmetric Mean Absolute Percentage Error
- Defined by this equation:

$$SMAPE = \frac{100\%}{n} \sum_{t=1}^{n} \frac{|Ft - At|}{(|A| + |F|)/2}$$

- Where A is the actual value and F is the forecast value.

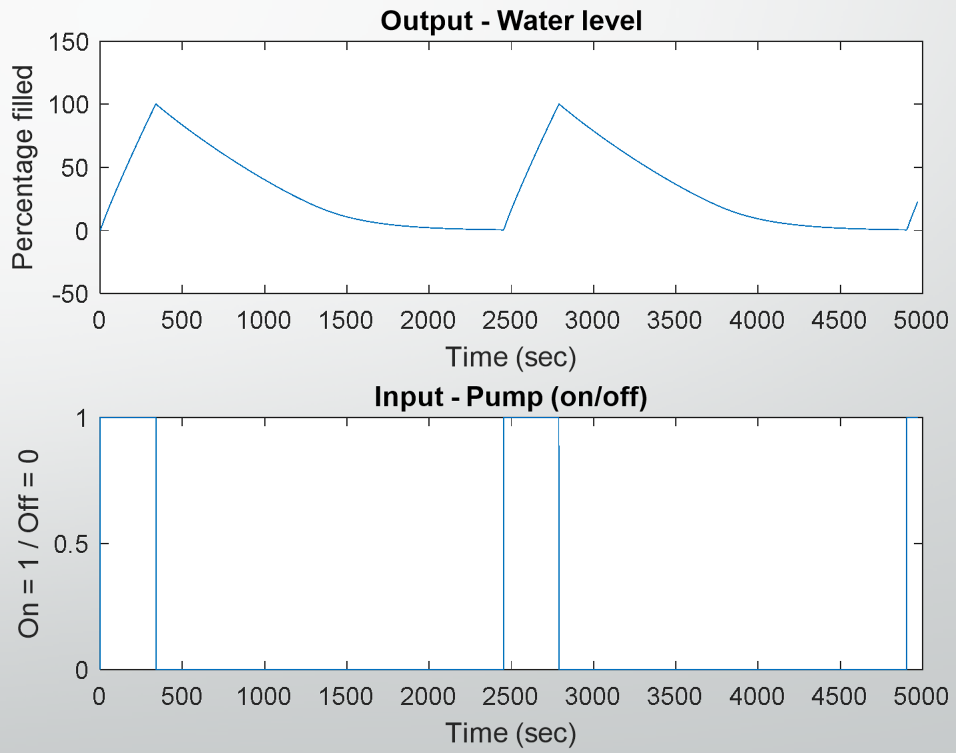
Case Study: Water Tank



- **Input** to the physical system - flow of water controlled by a valve (discrete values – on/off {0,1} or range of values [0, 1]). The PLC through its actuators supplies this value.
- **Output** from the physical system - height or pressure of the water in tank (Water Level). sensor used to determine this value. The sensor data goes to the PLC.

Case Study: Water Tank

- Physical System - single input and single output (SISO) system.
- Multiple Input Single Output (MISO) system version possible also.
- Time Series Data (Input and Output) is collected form the physical system while a standard program for the ICS is run.
- A standard program is run for PLC that exemplifies normal operation as data is collected.



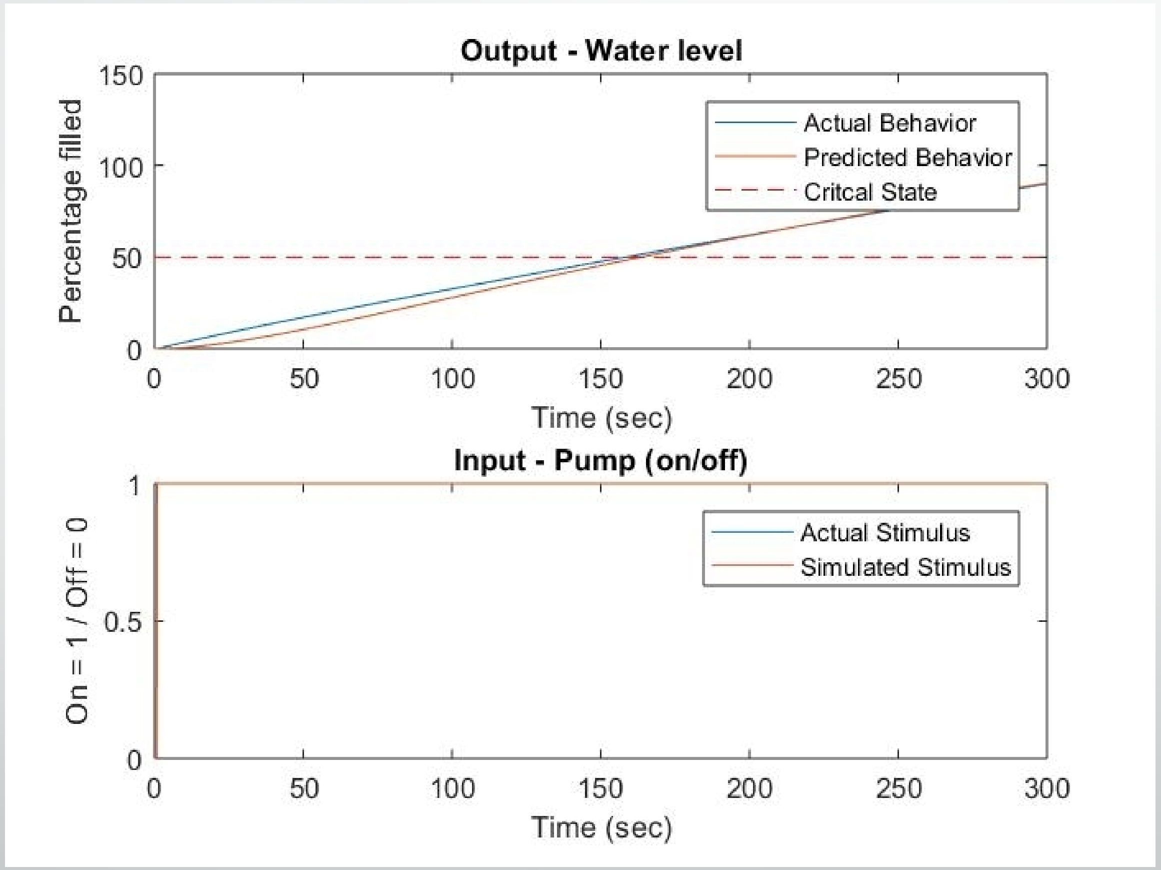
Case Study: Water Tank

- For the Testing phase of the process, new commands or programs may be accessed by the IDS.
- Essentially, once the packet is received, it is processed to determine what effect it would have on the overall system.
- To do this would require knowing the current state of the system (water level).

Experiments in Prediction for the Testing Phase

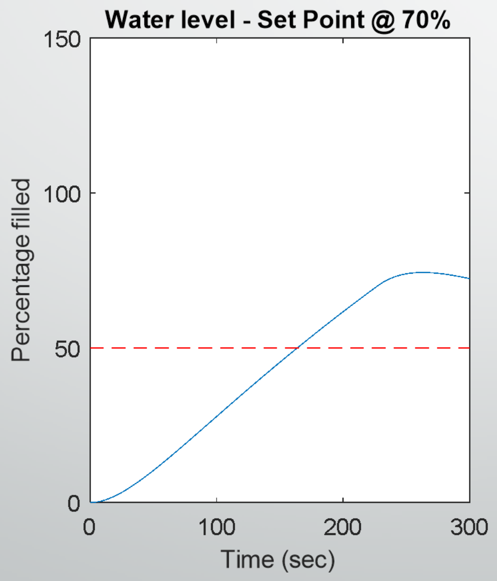
- Several basic preliminary experiments were performed with a virtual machine (VM) assigned a single core and Raspberry Pi
- Allows for baseline results and understanding.
- Latency and accuracy were observed.
- Time to Criticality Observed.

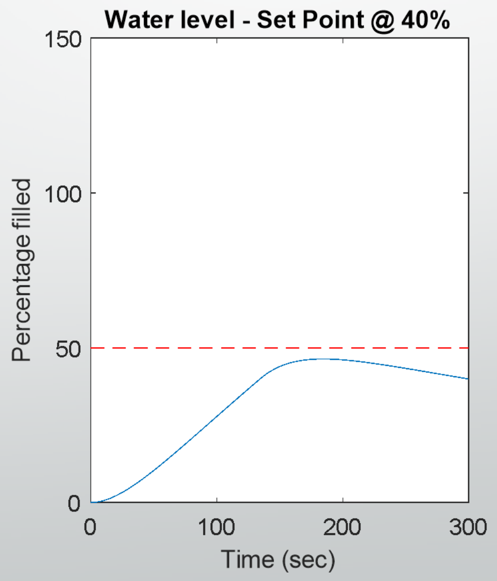
Results – Simple Commands to Run Pump



Results: On-Off Controller

- Commands to change parameters of control system – high setpoint in this case.





Results

	Saving Data to File		Storing to Data Structure	
Platform	VM on single Core	Raspberry Pi	VM on single Core	Raspberry Pi
Latency (secs)	0.111261	0.361271	0.147690	0.29277
Final Simulation Time (secs)	1000.00	1000.00	1000.0	1000.0
Latency once critical state reached (Secs)	0.020707	0.058519	0.031252	0.046684
Estimated Time to reach critical state (secs)	163.9	163.9	163.9	163.9

Conclusions

- Methods detect whether a malicious attack occurring based on predicting the effect of the attack on the physical system using a model before carrying out the command or loading the program.
- Attack takes the form of a command or a program with a malicious objective.
- In SCADA systems, where physical processes are involved, it is important to take into account that certain actions can perturb these physical processes in undesired and harmful ways.
- General IT mitigations may be effective in detecting and combatting attacks with abnormal network behavior.
- However, General IT methods may fail because the malicious command/program may appear normal to common IDSs from a network perspective.

Appendix

IDS Process – Stage 2: Testing – Step 4 for Malicious Programs

- **Dynamic Analysis of Malware**
  - One approach to analyzing software to determine whether it is malware or not and how it behaves is to simply allow the software to run in a special environment, such a virtual machine and observe how it behaves over time.
  - Major differences between generic software that may be malicious and malicious ladder logic. Ladder logic as defined in this work is limited in that it can only affect the output based on internal states, timers, and the input that it receives.
  - Generic malware, on the other hand, does not possess these constraints. Generic malware can affect file systems and utilities within the operating system of the host machine, whereas ladder logic affects the output of the PLC. Therefore, generic malware should be evaluated in an environment fully representative to what it would encounter when it is affecting real systems. On the other hand ladder logic can be observed in a single OpenPLC process to perform dynamic analysis.

Case Study: Water Tank – black box

- The collected Data is split between (1) training data and (2) validation data.
- Input and Output Training Data are used to train a set of models to represent the behavior of the physical system.
- The validation input data is applied to each model to produce output. This resulting output for each model is compared with the validation output.
- The closest match determines the model selected.

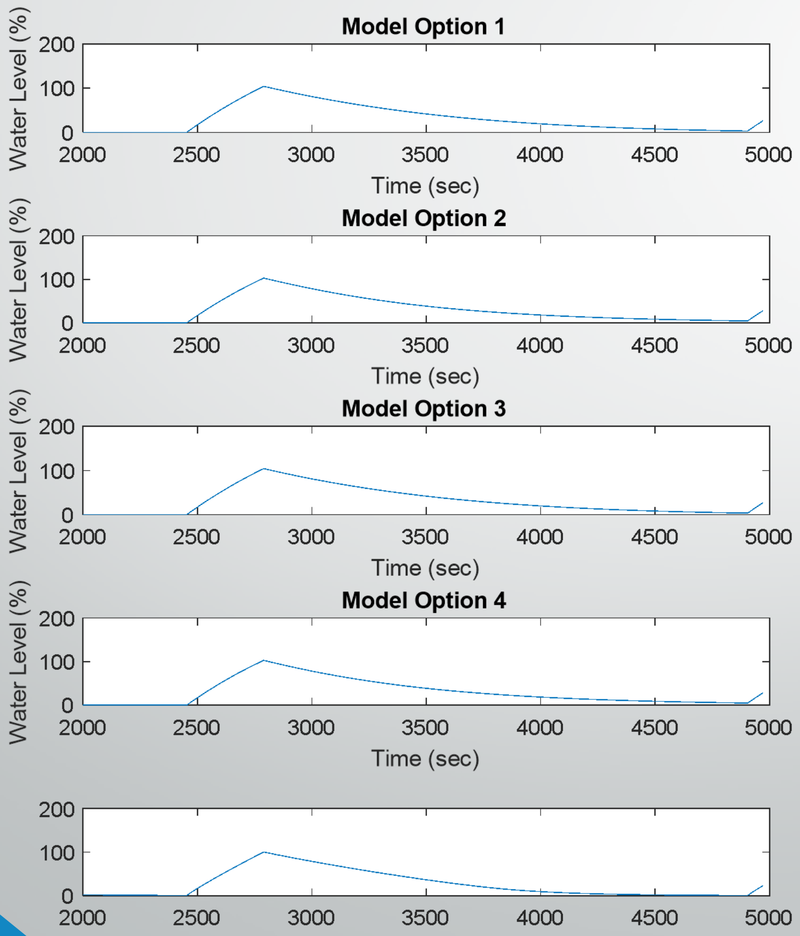
Case Study: Water Tank – black box

- Model Developed. Several Models structures are selected (e.g. transfer functions with different coefficients in denominator and numerator).
- Test inputs applied to these models to generate output, which is compared with the actual output. The model that produces output closest to the actual output according to a given metric is the one that will be selected.

Case Study: Water Tank

- ARMA Model tried with deferent numbers of parameters for the input and output of the physical Model:
  - Model 1: $y[t] = a_1y[t-1] + a_2y[t-2] + a_3y[t-3] + a_4y[t-4] – b_0u[t] + b_1u[t-1] + b_2u[t-2] + b_3u[t-3]$
  - Model 2: $y[t] = a_1y[t-1] + a_2y[t-2] + a_3y[t-3] - b_0u[t-1] + b_1u[t-2] + b_2u[t-3]$
  - Model 3: $y[t] = a_1y[t-1] + a_2y[t-2] + a_3y[t-3] + a_4y[t-4] - b_0u[t] + b_1u[t-1] + b_2u[t-2]$
  - Model 4: $y[t] = a_1y[t-1] + a_2y[t-2] + a_3y[t-3] - b_0u[t] + b_1u[t-2] + b_2u[t-3] + b_3u[t-4]$

Case Study: Water Tank



- Fit results for the four models:
- Model 1: 0.8215
- **Model 2 0.8426**
- Model 3 0.8115
- Model 4 0.8307

Methods – for IPS to Detect Threat (Cont.)

- Previous works assume that model is readily available [1 - 5] for Power Systems. The model in those works are essentially white-box models.
- However this availability should not always be assumed if it is desired to have an IDS that is generic and deployable.
- System Identification (methods for building models that represent physical processes from data) techniques are employed in this work for creating a model during a training stage of the IPS. These methods would be used for the case that the model is not explicitly given.
- Input and Output data over time are collected through sensors and actuators.
- The purpose of these techniques are to allow for the generation of models that represent the physical system.

Research Questions

- Are there limitations as to what physical systems can be modeled using the methods of this work?
- How generic and deployable are the methods in other words?
- If no human-defined specification is given for the model and/or the safety criteria, would the IPS still be reasonably accurate in detecting packets with malicious payloads?
- Would the IPS from a black-box perspective be as accurate as an IPS from a white-box perspective.
- If the former is not as accurate as the latter, is the accuracy within some reasonable tolerance?

Layers

Layer 0

- The firmware is the underlying software of the PLC.
- It consists of drivers for the hardware that serve as interface for the ladder logic to retrieve input values and to set output for the hardware.
- Firmware may also have an operating system, such as Linux or an RTOS (Real Time Operating System)

Layer 1

- Consists of the ladder logic program or Programmation, which serves as the software to control the PLC. ladder logic is characterized by a scan cycle, in which input is read and new outputs are computed based on the internal logic of the program.
- Scan cycle is repeated over many iterations during the operation of the industrial plant that the PLC serves.

Layer 2

- The programmation or ladder logic will have internal states or settings that may be modified by the operator through Modbus or other SCADA protocol.
- Examples: settings may be setpoints, such as the set point for a PID controller, high low and low set points for an on/off controller, gains of the PID controller or other control algorithm. The output is important to consider since it directly affects the physical plant.

Threat Model: (2) Malicious Programation Uploads (affecting Layer 1)

- Malicious Programation Upload Packets may also be sent to the PLC.
  - ***Basic Malicious Program upload*** **– software whose malicious component* is active once the software is run.**
  - ***Time Bomb*** **– software whose malicious component is activated at a predetermined time or date.**
  - ***Logic Bomb*** **– software whose malicious component is activated at a specific set of predefined conditions.**
  - * Malicious component is the part or function of the program that when activated immediately causes the harmful consequences of the cyber-physical system.

Selected Literature Review

[1] Chromik, Justyna J., Anne Remke, and Boudewijn R. Haverkort. "What's under the hood? Improving SCADA security with process awareness." In Cyber-Physical Security and Resilience in Smart Grids (CPSR-SG), Joint Workshop on, pp. 1-6. IEEE, 2016.

[2] Chromik, Justyna J., Anne Remke, and Boudewijn R. Haverkort. "Improving SCADA security of a local process with a power grid model." In ICS -CSR. 2016.

[3] Semantic Security Analysis of SCADA Networks to Detect Malicious Control Commands in Power Grids

[4] Lin, Hui, Adam Slagell, Zbigniew Kalbarczyk, Peter Sauer, and Ravishankar Iyer. "Runtime semantic security analysis to detect and mitigate control-related attacks in power grids." IEEE Transactions on Smart Grid (2016).

[5] Etigowni, Sriharsha, Dave Jing Tian, Grant Hernandez, Saman Zonouz, and Kevin Butler. "CPAC: securing critical infrastructure with cyber-physical access control." In Proceedings of the 32nd Annual Conference on Computer Security Applications, pp. 139-152. ACM, 2016.

[6] Hadžiosmanović, Dina, Robin Sommer, Emmanuele Zambon, and Pieter H. Hartel. "Through the eye of the PLC: semantic security monitoring for industrial processes." In Proceedings of the 30th Annual Computer Security Applications Conference, pp. 126-135. ACM, 2014.

[7] S. McLaughlin et al., "A Trusted Safety Verifier for Process Controller Code," Proc. Networks and Distributed Systems Security Symposium (NDSS 14), 2014

[8] Alves, T., Das, R., Werth, A., Morris, T., Virtualization of SCADA Testbeds for Cybersecurity Research: A Modular Approach, in Computers & Security, Volume 77, August 2018, Pages 531-546, https://doi.org/10.1016/j.cose.2018.05.002

[9] Assante, Michael J., and Robert M. Lee. "The industrial control system cyber kill chain." *SANS Institute InfoSec Reading Room* 1 (2015).

[10] Gao, Wei, and Thomas H. Morris. "On cyber attacks and signature based intrusion detection for modbus based industrial control systems." *Journal of Digital Forensics, Security and Law* 9, no. 1 (2014): 3