Mitigation of Industrial Control System Attacks Using A Boosted Filter Ensemble

Kelvin Ly*, Orlando Arias*, Jacob Hazelbaker*, Andrew Hughes*
Faculty Advisor: Yier Jin†
*Department of Electrical and Computer Engineering, University of Central Florida
†Department of Electrical and Computer Engineering, University of Florida
rangertime@knights.ucf.edu, oarias@knights.ucf.edu, yier.jin@ece.ufl.edu

***Abstract*—We are developing a solution that uses an ensemble of modelling methods to detect sensor and PLC compromises/failures. These different methods are combined together using gradient boosting to produce a highly robust filter. Various other novelties in our approach are mentioned as well.**

I. Introduction

The CSAW 2017 Embedded Security Challenge focuses on security industrial control systems against attacks on PLCs and sensors. The challenge is based around the problem of securing a system built on top of legacy PLCs. These PLCs will be running code that we will not have access to, and consequently, they effectively act as black boxes that may or may not be controlled by the attacker. Similarly, the sensors may or may not be compromised by the attacker. Following Shannon's maxim, we assume the attacker has an almost perfect knowledge of the system, in terms of its implementation, aside from not knowing some set of secrets. The goal overall is to deploy a system on top of the existing infrastructure to mitigate an attacker's ability to cause the system to break by allowing it to enter an unsafe state.

There are two different forms of attacks, attacks that alter the output of a PLC, and attacks that alter the outputs of sensors in the system. Our defense against both of these consists of providing predictors that try to predict the future state of the system, estimators that support the predictors by estimating unknown hidden variables, and classifiers that use the predictions and estimations to determine whether a given deviation should be classified as an attack or failure. The predictors and estimators here will be a combination of some of the ones commonly found in control theory and those used in machine learning. This ensemble of classifiers will be combined together using gradient boosting to provide a hopefully robust single classifier that can accurately detect attacks. Apart from sharing this basic framework, PLC outputs and sensor outputs will also have some additional security measures that take advantage of their individual attributes.

II. Related Works

False data injection detection is very much still an open research area. A lot of work has been done in it for the specific case of large power grid systems, beginning with the work by Yao which demonstrated that a family of attacks were possible which were undetectable from normal power grid operation from the standpoint of state estimation [1]. Various other papers devised methods to overcome this shortcoming [2].

The

III. Implementation

The project can be separated into a testing framework integrating MATLAB that allows control systems built in Simulink to be used as test platforms, and the actual protection system used to detect and warn about potential attacks on the simulated control system. In a real life use case, essentially everything could be stripped out of the system except for the defense module, which would act as it does now as a proxy between the control system and the PLCs.

A. Test Framework

The control system is simulated using MATLAB on a separate computer using Simulink. The PLC is connected using a set of Simulink blocks created during this project that can translate Simulink signals into Modbus commands. These blocks are written in C, using MATLAB's S-block functionality, and use `libmodbus` to communicate using Modbus TCP. Each block keeps track of real time as well as simulation time, so that they can ensure the simulation stays in time with real world time. All communication past this point is done using Modbus TCP. MATLAB's Simulink PLC Coder was used to generate IEC61131-3 Structured Text function blocks, which were fed into a code generator that produces Structured Text code that would simulate a set of simulated PLCs. This step basically involves creating multiple instances of the function block and feeding inputs and outputs correctly to each of them.

A stack of Modbus proxies, with OpenPLC's Modbus slave endpoint at the bottom, provides the combination of attack simulation and defense testing. The Simulink blocks communicate with the top of the stack, which then forwards the request down in the stack, while potentially altering the request or response. In this way, the first attack proxy can alter sensor input by modifying sensor data write requests it receives from MATLAB. This allows attacks that modify the sensor inputs to the PLCs to be simulated.

After this first proxy is the defense module. During sensor write data requests the defense module duplicates the sensor data so that each simulated PLC will receive its sensor data, and also keeps a sample of the sensor data for attack detection. When the control system tries to read the PLC output, the defense module duplicates that request as well, but modifies the response so that the control system receives whatever response it deems correct. The exact details of this module will be discussed further in this paper.

The defense module proxies requests to the other attack module, which simulates attacks on PLCs. In this attack module, the responses from the PLC are altered during PLC output read requests. This module then communicates directly to the OpenPLC Modbus slave, completing the connection between the Simulink system and the PLC logic.

A lot of the complexity in this system is primarily meant to make it possible to use many existing tools and knowledge to build the control systems that are to be tested. The first step in preparing a system for testing is to either create or find a Simulink model for it. This model is then used to generate the PLC code, and then it is also slightly modified to allow it to function with the rest of the test framework. This is a fairly simple procedure, and thus testing PLC security using this framework could potentially be easily integrated into existing engineering design workflows fairly easily.

B. Defense module

Our method builds on top of the current standard for using redundancy to improve system reliability, enhancing it to be more robust. Our system shold be able to withstand some attacks where the attacker controls a majority of the controllers, and can detect some attacks on the sensors within the system as well. In an attempt to provide a robust, general solution that can reliably detect a large number of attacks than simple majority voting can't. The system will use an ensemble of approaches to determine the likelihood of error or attack originating from any of the PLCs. These of these approaches will individually calculate the likelihood of a failing or compromised PLC. These approaches will include modeling the PLCs as time delay neural networks, hidden Markov models, and low order control systems, using simple majority voting, and classifying their behavior using support vector machines. After some experimentation, some of the lower scoring approaches may be pruned to improve the performance of the system.

1) Security mechanisms exclusive to PLCs: Only one, if any, of the PLCs, will be virtually connected to the plant at any moment. All the PLCs will receive sensor data for each time step, but only the output of one of them will actually be fed into the actuators in the system. If the system predicts that an attacker has assumed complete control over all the PLCs, the system will temporarily switch into running using the predictors and estimators described above. Their control over the actuators will be time multiplexed, with the length of each PLC's time slot governed by the classifiers' belief in that given PLC's trustworthiness, preventing any attacker from exerting too much control over the system.

Additionally, the devices which are not at a given instant controlling the plant may be subject to various disturbances, which will provide additional data that can be used to characterize the PLCs. In particular, this will prevent some of the machine learning algorithms from overfitting by broadening the conditions they observe. It is expected that the systems are probably naturally fairly stable, and consequently most of the data collected will be of the system working at or near its set point. Emulating disturbances allows a more complex model that more accurately predicts system behavior to be developed.

2) Filters to be implemented: This section will cover the different systems that will feed into the gradient boosting algorithm. Each of these will independently model, predict, and monitor the inputs and outputs of PLCs and sensors.

One filter will be a simple majority filter. Given an odd number of redundant PLCs, the filter will select the median value of the PLCs. The standard deviation of the output will be measured online, and this estimate will be continually improved as the system runs. Any PLCs deviating from the median value past a certain multiple of the standard deviation will be marked as faulty.

Another filter will be use least means square to estimate the control system, and each PLC using third order models. The filter will constantly accept new data to improve its estimates, and it will also maintain some confidence value for the accuracy of the parameters. A PLC whose parameters deviate noticeably from those of the others, or whose signal violates its own predicted signal by a noticeable margin, will be marked as faulty.

A third filter will attempt to use time delay neural networks to model the system, PLCs, and sensors. It will attempt to classify the signal at any moment as belonging either within stable operation, or as belonging to a compromised or failed device.

IV. Discussions

This system uses many separate, fairly robust, well studied algorithms as a result of the very general nature of the problem it is attempting to solve. Industrial control systems are very diverse, and consequently any attempt to protect them in a general case must be very robust to adequately work in a large number of systems. It isn't expected that any one of these algorithms can adequately model all control systems, but it is expected that at least some of them will perform well enough for all control systems. The gradient boosting performed acts to select the set of algorithms that work well for each system.

V. Future Work

One very promising alternative approach towards control system security would be to use a variation of generative adversarial networks to potentially discover and learn new

possible attacks much more quickly than the previous set up. In the canonical generational adversarial network setup, one algorithm is learning to forge something, while the other algorithm is learning how to distinguish forgeries from the real product [**?**]. The objective in that case was to train the generative model to produce articles that are nearly indistinguishable from the real thing, whereas in this case there would be a few subtle changes to use the same basic idea to develop a model that can detect increasingly deceptive attacks, as well as a model that develops increasingly stealthy attacks. This would be done by adjusting the algorithm slightly. Thus, instead of attempting to just fool the defense module, the generative model would also be scored on how effective the attacks that went undetected were. Some score, such as the total amount the state variables of the system deviated from their setpoints, could be used here. The defense module would be given a sample of data from both sensors and the PLC outputs, and would try to determine whether this was from normal operation or from some type of attack. Over time, the generative model should learn how to generate more sophisticated data modifications to try to fool the defense module, and the defense module should learn how to deal with more covert attacks.

This idea was discovered fairly late in the process, and thus was not completed by the deadline for this paper. This may or may not be finished by the time of the competition.

VI. Conclusion

TBD

References

[1] Y. Liu, P. Ning, and M. K. Reiter, "False data injection attacks against state estimation in electric power grids," *ACM Transactions on Information and System Security (TISSEC)*, vol. 14, no. 1, p. 13, 2011.

[2] R. B. Bobba, K. M. Rogers, Q. Wang, H. Khurana, K. Nahrstedt, and T. J. Overbye, "Detecting false data injection attacks on dc state estimation," in *Preprints of the First Workshop on Secure Control Systems, CPSWEEK*, vol. 2010, 2010.