

Implementing OpenPLCs into a Cyber Defense Competition

Team 16

Dr. Doug Jacobson
Dr. Julie Rursch

Nick Springer - Security Engineer
Matthew McGill - Project Manager
Val Chapman - Software Testing Engineer
Josh Przybyszewski - Software Engineer
Joseph Young - Software Engineer
Liam Briggs - Hardware Engineer
Brennan Fergusson - Hardware Engineer

sdmay18-16@iastate.edu
<http://sdmay18-16.sd.ece.iastate.edu>

Revised: 10/29/2017

Contents

1 Introduction	2
1.1 Project statement	2
1.2 purpose	2
1.3 Goals	2
2 Deliverables	2
3 Design	3
3.1 Previous work/literature	3
3.2 Proposed System Block diagram	3
3.3 Assessment of Proposed methods	3
3.4 Validation	3
4 Project Requirements/Specifications	4
4.1 functional	4
4.2 Non-functional	4
5 Challenges	4
6 Timeline	5
6.1 First Semester	5
6.2 Second Semester	5
7 Conclusions	6
8 References	6
9 Appendices	7

1 Introduction

1.1 PROJECT STATEMENT

This project's task is to explore the OpenPLC project and determine how it can be implemented into a Cyber Defense Competition (CDC), with the intention of simulating a real-world cyber physical environment. Up to this point, the CDC has utilized virtual systems and scenarios to emulate network interactions. These scenarios have enabled students to learn and grasp security-related concepts in incredible ways, but they are limited in their ability to simulate physical equipment. By incorporating programmable logic controllers, which are computers that have been adapted to control manufacturing processes, the competition will provide more realistic and practical scenarios that will prepare competitors for interactions with this industry standard equipment. Additionally, adding support for programmable logic controllers into the competition environment provides a plethora of new opportunities for encouraging sponsor interaction. Learning to protect virtual environments is absolutely essential in the world of security, but security sometimes extends beyond the virtual world; this is the improvement our project shall bring to the CDC.

1.2 PURPOSE

With the advent of the Internet of Things (IoT), many physical systems are now relying on network connectivity to provide functionality to users. These systems are often large, and provide an invaluable service such as power or water management. In addition, they are often undersecured. By incorporating the OpenPLC project (which simulates the PLC hardware required to control and monitor these types of systems) into the CDC, we hope to provide the competition's contestants with valuable experience in securing cyber physical systems from malicious actors.

1.3 GOALS

Our first task is to become familiar with the OpenPLC project; we must have a workable understanding of what a PLC is, how it operates, and how it may be programmed. In addition, we must analyze the existing CDC environment, so we are familiar with its architecture. From there, we should develop a basic web interface which allows users to send signals to the PLC and monitor signals being received. We should first seek to produce this base integration with the CDC environment, so that it can be extended upon in future years for different scenarios. Once we have a base integration, and have brainstormed several potential scenarios that may be interesting for contestants, we will select one or two scenarios to serve as a proof of concept. We will test any new hardware required for the scenario, and develop a module which allows this scenario to serve as an extension on top of the base integration with the CDC delivery system (an ISERink). If possible, we should seek to complete the design and as much testing as possible during the first semester of the project.

2 Deliverables

A main goal of the implementation of this project is to be easily replicable for any Cyber Defense Competition. To successfully maintain this characteristic, the project's deliverable will not require specific hardware, but will be based on a connection of virtual machines. Each team present in the competition will require three virtual machines, two running Linux operating systems to host OpenPLC and the web app to monitor it and the other running Windows Server to host Factory IO.

The Linux servers will run Ubuntu Server. Currently, version 16.04 is the most current LTS edition of Ubuntu and will be used. The technical requirements of these servers are very minimal, as OpenPLC and the web app do not require many resources. The Windows Server will use the 2012 edition of the operating system. This system will need significantly more resources than the OpenPLC server because of the graphical requirements of the Factory IO software. Factory IO and OpenPLC will come with pre-built scenarios by our team. These scenarios will be well documented so the contestants understand how they work and can competently secure them.

Creating and distributing images of these machines is a simple and well practiced task during the initialization of the competition. Teams will be responsible to configure connections between the machines according to our provided documentation. They will also be responsible for securing the systems, but explicit instructions for this will not be included in the documentation.

3 Design

3.1 PREVIOUS WORK/LITERATURE

Historically, the Cyber Defense competitions have only done two cyber physical CDC's. This project is a continuation and expansion of a previous implementation using 3D printed cities. These cities only provided output in the form of LEDs to denote whether your system has been hacked or not. In the project, the goal is to simulate a factory environment using software called Factory.io. Essentially, Factory.io is a virtual factory environment that can interface with OpenPLC. Utilizing both OpenPLC and Factory.io will allow for simulation of real world environments inside of the CDC. OpenPLC is the platform that will act as the interface between the server side of the CDC and the factory simulations.

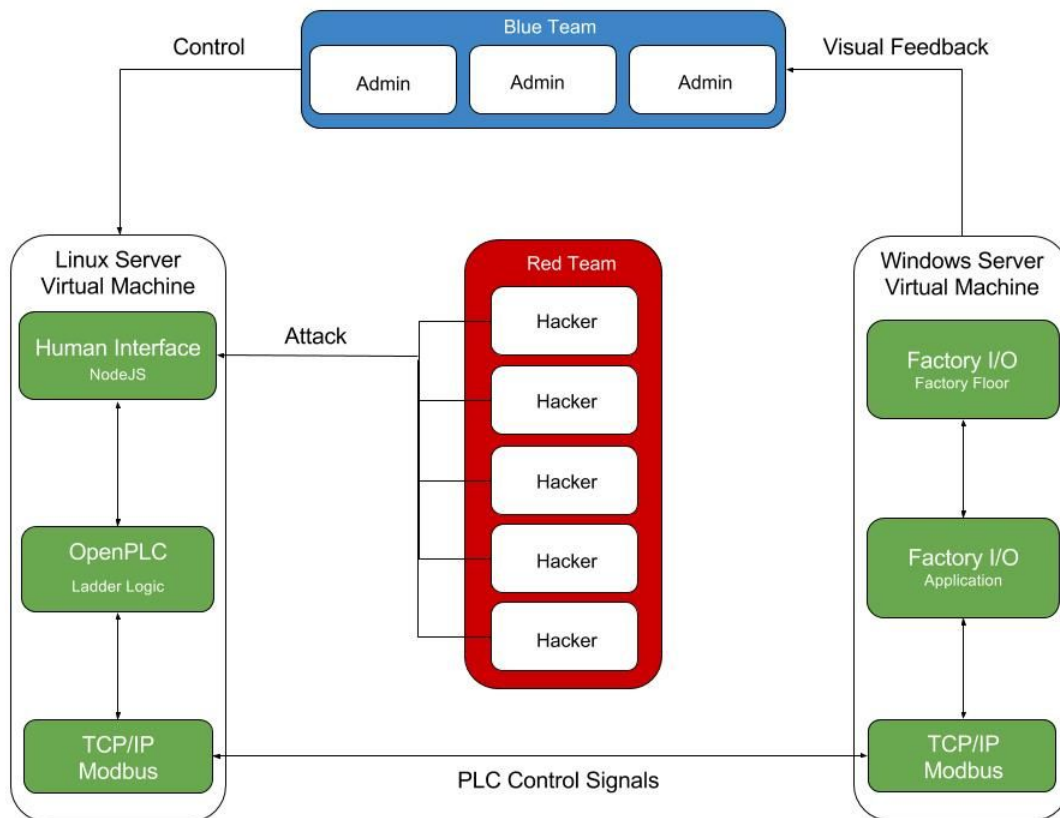


The team has spent some time looking around on the internet and we haven't found many previous projects that would provide a foundation for us. As a result, there is no foundation to

build off of other than the OpenPLC software in a Linux environment that will connect to the Factory.io software.

3.2 PROPOSED SYSTEM BLOCK DIAGRAM

The goal of the project is to use OpenPLC to simulate cyber physical systems. In this project, the cyber physical system will be represented as a real world factory floor. Factory I/O will be used to simulate the factory floor, and provide visual feedback to the CDC contestants, or the Blue Team. To control the factory floor, the Blue Team will use a web interface written in NodeJS to control the OpenPLC software running in the background. Using the TCP/IP Modbus protocol, OpenPLC will control various sensors in the simulated factory floor to complete a task. Lastly, the Human Interface will contain various security flaws that the Blue Team will have to patch in order to prevent the hackers, or the Red Team, from exploiting it. The block diagram below shows an overall view of the design described in this paragraph.



3.3 ASSESSMENT OF PROPOSED METHODS

At a high level, the project does seem to be something that will integrate well with the CDCs. The system will be able to directly connect to the participants for virtual access and will also

allow the competitors to interact with the devices physically. The setup process will be virtual in order to allow remote participants the capability to work on the systems off campus.

The solution provided will also meet the requirement we have which is to help the students learn more about cyber physical system. Being able to simulate a physical system and implement vulnerabilities into it will teach

3.4 VALIDATION

We will verify that our solutions work by doing incremental testing. Initially, the validation will start by building some Linux/Windows virtual machines to connect the OpenPLC and Factory.io software.. A great place to start would be trying to send data to control the factory systems from OpenPLC. Any simple experiments to better understand how the systems can interact together will be beneficial.. This process will not only help us learn how the system works, but will provide us with an idea of what will and won't work.

4 Project Requirements/Specifications

4.1 FUNCTIONAL

Blue Team Services: We will have virtual access to servers/web apps to control and protect the factory floor service. Additionally, the white/green team should have easy-access to all required software and hardware.

Hardware Devices: Raspberry Pi, any physical connected sensors and devices (light sensors, gauges, lights, pipes, etc.). Since our project will likely *not* have hardware, we will produce a list of potential interfaces for future and other school's CDCs to use.

Software Interface: We will have a NodeJS server for uploading to Raspberry Pi/OpenPLC, web interface for monitoring physical systems and sending signals to devices, and virtual PLC running on a Linux box.

4.2 NON-FUNCTIONAL

Documentation: Thoroughly document vulnerabilities and design decisions. Documentation is very important for posterity, so future CDCs and CDCs at other schools can understand what we were thinking in order to duplicate and improve the scenario.

Intentional Hardware Vulnerabilities: This would be a physical location or interface for red team to insert a USB drive or access a WiFi network.

Intentional Software Vulnerabilities: Multiple modular software vulnerabilities for red team to exploit. They need to be modular enough so that the CDC planners can choose a few from a larger selection so that red team doesn't see the same vulnerabilities every CDC.

NOTE (On the ethics of intentional vulnerabilities): Since the CDC is designed to give real-world experience with unethical hackers, our choice to leave intentional vulnerabilities is made in full light of the circumstance. In the real-world, hacking is illegal, immoral, and bad. However, hackers

exist, and the CDC gives “the good guys” a chance to find out what hackers will do. Our intentional vulnerabilities give the blue teams the opportunity to patch these areas.

4.3 STANDARDS

Code style: Standardizing our code-base is essential to the longevity and maintainability of our software. The coding standard will be defined by language. For example, C might be standardized by K&R's C, Dart will be standardized by Dart standards, etc.

5 Challenges

There are several challenges that may slow us down in our development process. These include documentation of the open source projects we use, and developing a CDC project that could be used to teach students cyber security systems.

We plan on using OpenPLC in our project, which presents a few opportunities, but also several challenges. This resource is great because it is a part of the open source community, meaning it's completely free to use. It also has received a decent amount of contribution from the development community, including engineers who have been working with PLCs in the industry for decades, making them super knowledgeable and helpful in this project. However, all of those nice things being said, the documentation and detailed use cases of this project are severely limited. This has made working with the OpenPLC project very difficult. Our team has also had several discussions with CDC industry partners about staying away from OpenPLC due to its open-source nature. Companies, of course, will want to have more exclusive rights to research and intellectual property, thus their lack of interest in open source. At this point, we will continue to stick with using OpenPLC, but this may come at the expense of third party companies pulling support and resources from future CDC progress.

PLC's use a language called ladder logic, which is a language that no members of the team have experience in, so we will all have to learn this new language to implement it into our design. When an entire team is learning a new language, there are definitely going to be growing pains.

Financially, our team has had few problems or major road blocks. The beauty of choosing to use Factory I/O over building each CDC team a physical environment is that it is *much* cheaper, yet still retains many of the same functionality a physical system would offer. Because our team has chosen to pursue this path, our project has become easily scalable and can adapt to a large number of teams. There is a cost to purchasing a Factory I/O license, which ranges from \$115 to \$800 for the ultimate edition. We are currently in the discussion phase for which version we would like to purchase, but this would be a one-time cost for us to use the software across all CDC teams.

6 Timeline

6.1 FIRST SEMESTER

Scenario's Picked and Designed: 11/1

Project Plan(v1): 9/22

Project Plan(v2): 10/27

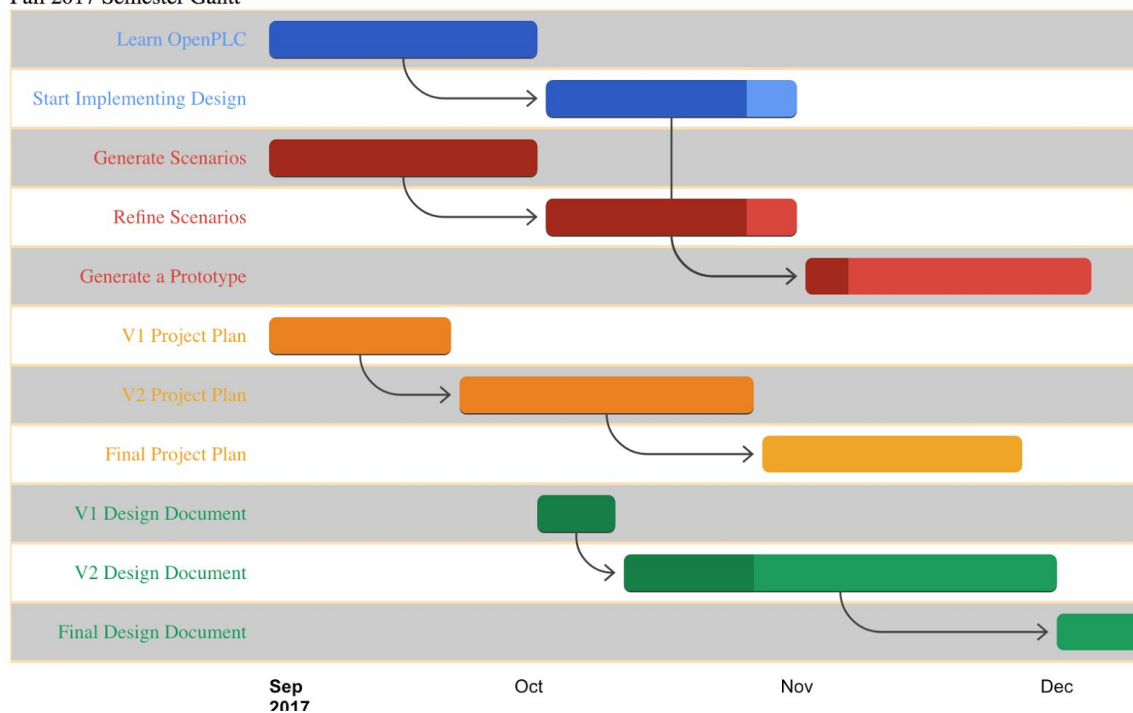
Project Plan(Final): 12/1

Start Implementing Design: For this milestone we wanted to connect Factory I/O on a Windows Machine and have it controlled by OpenPLC on a linux machine. We have connected the two machines, and are currently working on fine tuning the connections and setting up the Ladder logic.

Refine Scenario: This milestone is deciding on what our scenario will be for the CDC. This is an important milestone, because we need to make sure that our product, capitalizes on our software while still giving a great learning experience for our users.

Generate a Prototype: For the end of the semester we are hoping to have a prototype that is connected and set up on our ISEAGE server. We would like a few vulnerabilities on our system, and documented ways to get into these servers.

Fall 2017 Semester Gantt



6.2 SECOND SEMESTER

Second semester schedule is *much* more tentative.

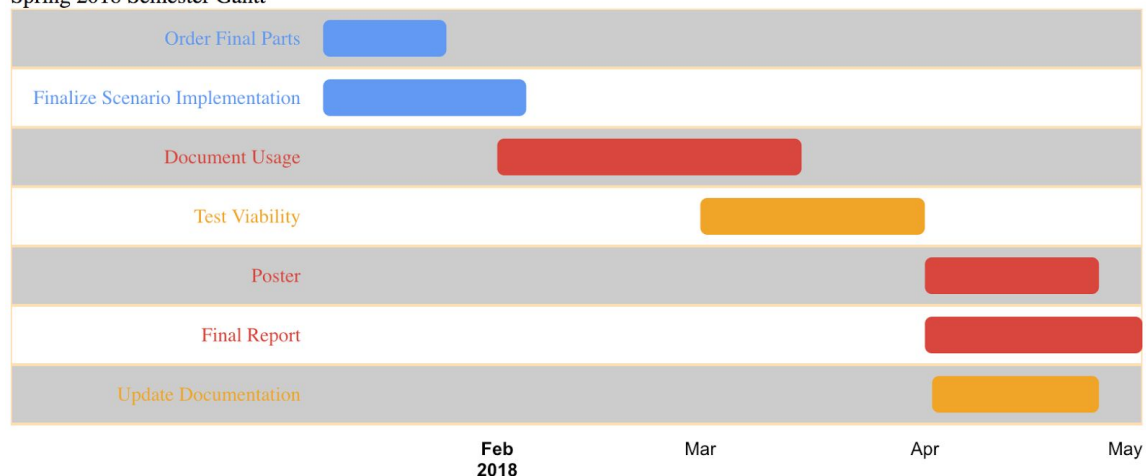
January: The first milestone of the second semester we would like to have our licensed Factory I/O setup on a linux server on ISEAGE, communicating with the windows server and ready to have test users give it a try.

February: The February milestone is to have our idea implemented and ready to start testing personally, and documenting how our users are able to use the product.

March: This milestone is having users test our product and its documentation to make sure that our documentation makes sense to new users.

April: The final Milestone will be working completely on documentation, and finalizing bugs found. We will then be finalizing our final presentations.

Spring 2018 Semester Gantt



7 Conclusions

As one can identify from the above report, our team's overall goal for this project is to take a programmable logic controller and implement it within an Iowa State cyber defense competition (CDC). The need to emulate real cyber-physical systems within the context of these CDCs is really the next step in educating students about the necessity of security in almost every single industry.

To tackle this problem, my team has broken this lofty vision and long-term goal down into manageable steps. We will spend the rest of this semester and next expounding upon our work with the Factory I/O Platform, building bigger/more complex systems over time. We have carefully crafted several different scenarios that will make for some great CDCs, bringing together the virtual

aspect that has made the CDCs so successful in the past with the physical realm that will launch the CDCs into yet another generation of student success.

8 References

<http://www.openplcproject.com> - The OpenPLC Project Official Webpage and Documentation

<http://www.plcsimulator.net/login.php> - PLC Online Simulator

<https://factoryio.com> - Factory I/O

<https://www.dartlang.org/> - Dart language specification

9 Appendices

This section is not applicable for v2, but will continue to play a larger role in later versions of this document.