Industrial Contol Systems

**Protecting a PLC From Corruption**

# plc-APP

**Lab Description:** This lab explores a security issues related to the use of Programmable Logic Controllers (PLCs) in the management of Industrial Control Systems (ICS), or similar forms of infrastructure.

You should read this "Lab Description" section before starting the lab. The student is expected to have performed the Labtainer "onewayhash" lab, or otherwise learned about the use of openssl generate digests.

This PLC lab simulates the system illustrated in Figure 1. A PLC manages the water level of a creek-fed catfish pond, ensuring the water level does not exceed minimum and maximum limits.

You will interact with the sys\_management system to load a program and configuration data into the PLC. You will also use the sys\_management to check the status of the PLC and to query which program and configuration data the PLC is running.

The monitor system is used to query the status of the PLC (which can also be performed at the sys\_management system). The monitor system must be able to continually monitor the PLC, or the farmer will have a nervous breakdown.

You will not have direct access to the PLC subsystem, though you can interact with it via the sys\_management and monitor computers.

A "Security Proxy" sits between the sys\_management and montior computers and the PLC. This device can be configured to

1. Filter commands destined for the PLC, constraining the commands that may be issued from a given IP address.
2. Prevent unauthorized programs or data from being loaded into the PLC. The proxy uses a whitelist of authorized MD5 digests to validate files destined for the PLC.

Start the lab as noted below (if you have not already done so).

sys\_management

Security

proxy

PLC

Subsystem

Water level gauge sensor



Valuable crops

Creek flows

into pond

Pump

removes

water

monitor

Figure 1: Farmer Jones Catfish Pond’s Critical Infrastructure

**Lab Environment:**

The lab is started from the Labtainer workspace directory on your Docker-enabled host, e.g., a Linux VM. From there, issue the command:

labtainer plc-app

The resulting virtual terminals will include:

* A display of the status of the fish pond level, titled "Physical\_World".
* A bash shell on the sys\_management computer.
* A bash shell on the monitor computer.
* A bash shell on the Security Proxy, titled "ubuntu@proxy".
* A display of the Security Proxy log file titled "PROXY\_LOG".

NOTE: When the lab starts, observe the status window. The PLC is initially disabled, and thus the pump does not run and the water rises.

# Tasks

## Explore

The Physical World display is notional. It is not generated by any of the components of figure 1. It helps you understand what is happening in the physical world, independent of the subsystems.

Use:

manage\_plc status

from the sys\_management and monitor systems to observe the state of the PLC. Observe the log messages on the Proxy Log. Notice how there is periodic traffic? That is from a service on the monitor computer.

You can initialize the PLC from the sys\_management window using:

./manage\_plc.py load plc config.txt

The "plc" parameter is the name of the plc program file in your home directory. The "config.txt" is a configuration file in your home directory. This operation will initialize the PLC, leading to the pump to run.

The configuration file directs the PLC to keep the pond level between 20 and 30 feet. Just watch what happens over the course of about a minute.

After you've watched the status window for a full cycle of disaster, poke around a bit.

Hints:

* Use "stoplab" and "labtainer plc" from your Linux host to stop and restart the lab -- this is the best way to restart the lab or reset the PLC if it becomes corrupt. Any files saved on the components will be preserved.
* Watch the log on the proxy. Is the traffic through the proxy what you would expect?
* The manage\_plc.py tool lets your retrieve the code/data from the PLC. Are those the files you loaded?
* The sys\_management computer includes the openssl utility that you used in the onewayhash lab, might that help determine if the files are the same?
* Explore the proxy functions, e.g., use “proxy -h” and consider how they might be used to only permit authorized files to be loaded into the PLC?

NOTE: The solution must use the manage\_plc.py as-is. Modifying the code will void the warranty offered by the PLC system vendor!

**Lab Files that are Needed:** All of the files necessary for the lab are within the Labtainer components. Labtainers is retrieved from <https://my.nps.edu/web/cisr/labtainers>

### **Lab Exercise/step 1**

Alter the proxy configuration on the Security Proxy computer to prevent exploitation of the PLC. You are not expected to make changes to the sys\_management system, though you are free to explore it. However, credit will only be given if changes to the proxy configuration mitigate the attack.

## What to submit

When you have completed the lab, use “stoplab” to stop the lab, and provide the resulting zip file to your instructor.

This lab was developed by the Center for Cybersecurity and Cyber Operations at the Naval Postgraduate School in Monterey California.