LAB 1: INTRODUCTION TO SCADA CONTROL SYSTEMS

Estimated Time: 1 hour and 40 minutes

Purpose: The purpose of this Lab exercise is to familiarize the student with the virtual SCADA Control Systems, and begin configuring the virtual environment that will be used throughout the remaining labs.

Objective: The student will research software, protocols and tools used by SCADA Control Systems and the components within the UAH environment. The student should be able to operate the control system environments at the conclusion of this lab.

Lab Setup and Requirements: The student will need a browser with internet access to conduct research and download any necessary lab files. This lab assumes the student has previous experience using computer simulations.

EXERCISE #1 - INTRODUCTION TO THE UAH SCADA AND DOCKER ENVIRONMENT

The UAH environment will utilize Virtual Box to simulate three industrial control systems:

1. One Water Storage Tank
2. One Station Gas Pipeline
3. Heat Exchanger

The models can only be run individually.
Login for the virtual machine = username:ccre, password:ccre

The following scripts are provided on the virtual machine: /home/ccre/Scada lab/scripts. These scripts will be utilized throughout the labs, so familiarize yourself with the content and usage of each script.

1. gaspipeline.sh - Stops & removes all containers, and then starts the gas pipeline simulation
2. watertank.sh - Stops & removes all containers, and then starts the Water Tank simulation
3. heatexchenger.sh – Stops & removes all containers, and then starts the Heat Exchanger simulation

4. netstart.sh - Initialize the virtual network for the simulation (Netstart should only be run once during initial setup of the environment. In addition, it will fail if there are any existing containers attached to either network)
5. netstop.sh - Stops the virtual networks
6. cleanup.sh - Brute Force, removes all containers

The following references are provided to supplement or provide additional information on the tools, protocols, and software used within this environment.

- OpenPLC
- HMI
- SCADABR
- MATLAB
- ModBus
- Virtual Box
- OpenPLC Editor

EXERCISE #2 - SETUP SCADA LAB ENVIRONMENT

Section 1: Import the virtual machine into virtual box.

1. Download and install Virtual Box on your host machine.
2. Copy scadalab.ova file from the UAH Cybersecurity SCADA Labs disc provided.
3. In the Virtual Box Manager, select File>Import Appliance to import the virtual machine.
4. Click "Choose a virtual appliance file to import" icon and browse to the scadalab.ova. Click "Import".

Section 2: Login to virtual machine and run Heat Exchanger Docker containers.

1. To start the virtual machine in the VirtualBox Manager, select the scadalab VM, right-click and select Start>Normal Start. Login using the credentials provided in Exercise 1 (username:ccre, password:ccre).
2. Open Terminal by clicking on Applications -> Terminal Emulator
3. Navigate to the scripts folder with the command:

```
cd /home/ccre/scadalab/scripts
```

4. Run the script to configure the network with the command:

```
./netstart.sh
```

   If it asks for a password, type: ccre

5. Start the Heat Exchanger simulation with the command:

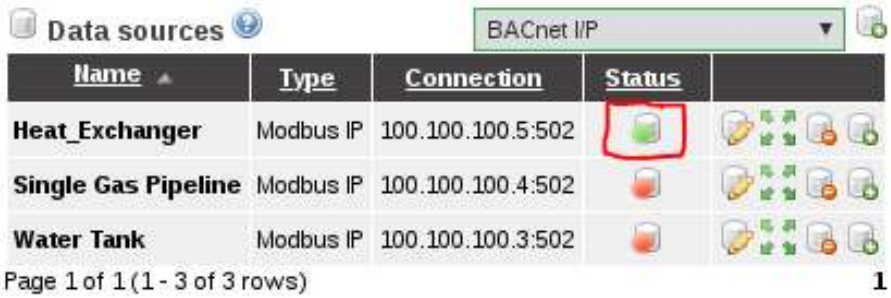
```
./heatexchanger.sh
```

6. Launch the heat exchanger HMI by opening the internet browser (Applications > Web Browser) and navigate to:

   100.100.100.2:8080/ScadaBR

   Login to ScadaBR using username:admin, password:admin

7. Click on Data Sources on the top menu and then enable the Heat_Exchanger data source to allow ScadaBR to pull data from OpenPLC.



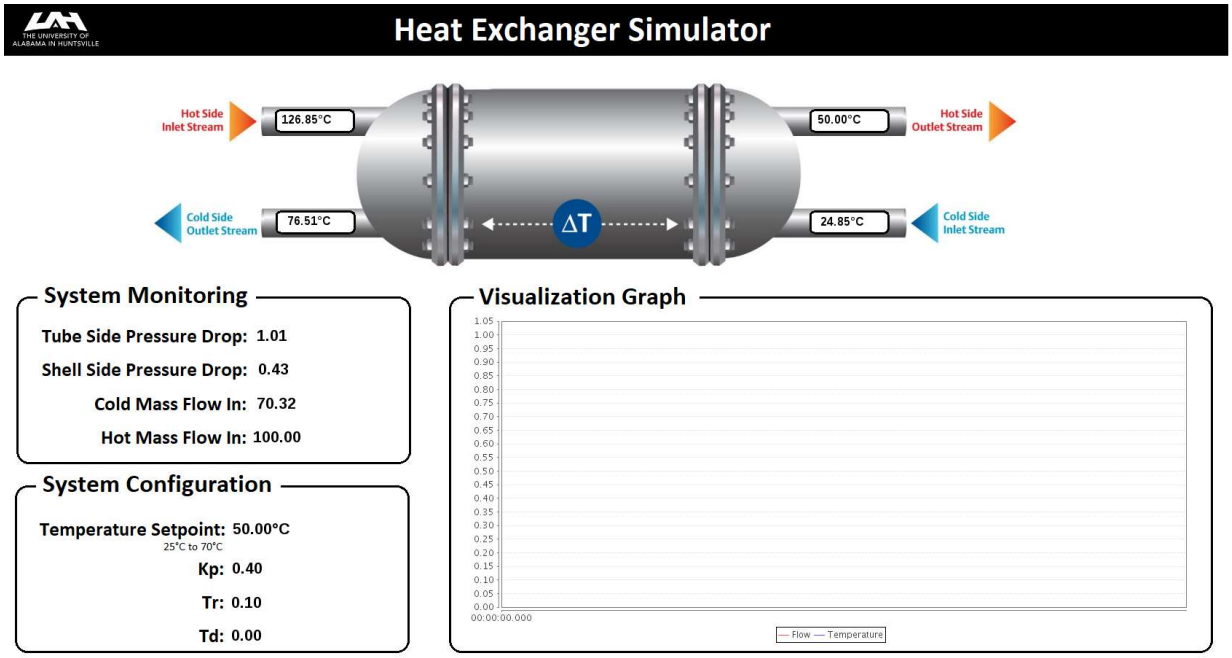
8. Click on Graphical Views, select the Heat_Exchanger HMI from the drop down menu.

CHECKPOINT: Refer to Troubleshooting Guide for FAQs regarding configuration and setup.

EXERCISE #3 - INTERACT WITH HMI AND CONFIGURE FOR UNSAFE OPERATIONS

Section 1: Interact with the Heat Exchanger HMI

1. Pull up the Heat Exchanger HMI in ScadaBR. You should see a screen similar to this one:



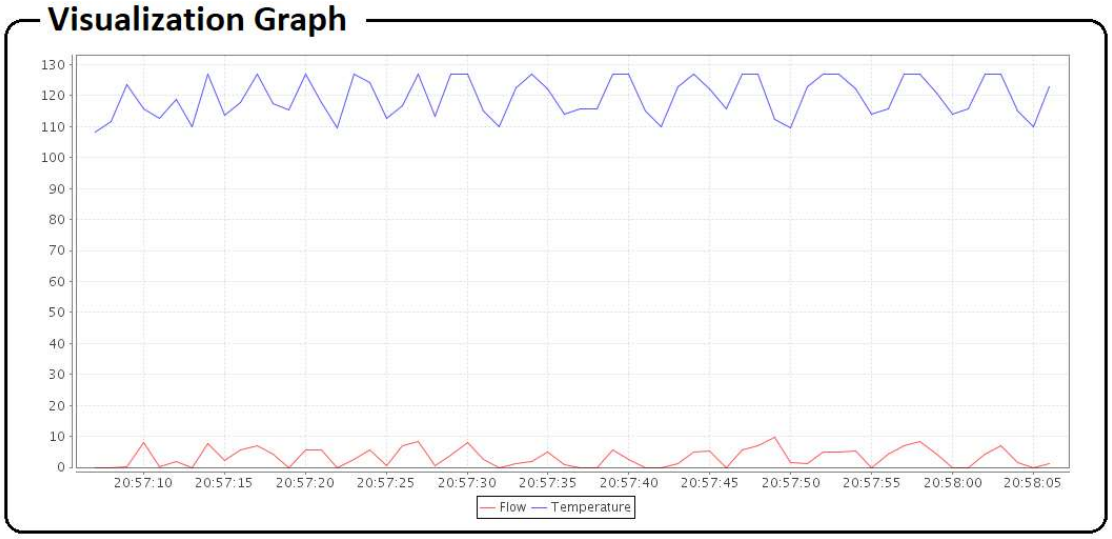
The Heat Exchanger has two inputs: a hot stream and a cold stream. OpenPLC runs a PID algorithm that varies the cold mass input flow so that the hot stream output temperature matches the setpoint temperature. The parameters of the PID algorithm are Kp, Tr and Td, and they all can be changed by the user. The temperature setpoint can also be changed by the user.

2. Change the temperature setpoint and observe the fluctuations of the flow and temperature in the graph. Keep the changes between 25°C and 70°C

Section 2: Configure Exchanger for Unsafe Operation

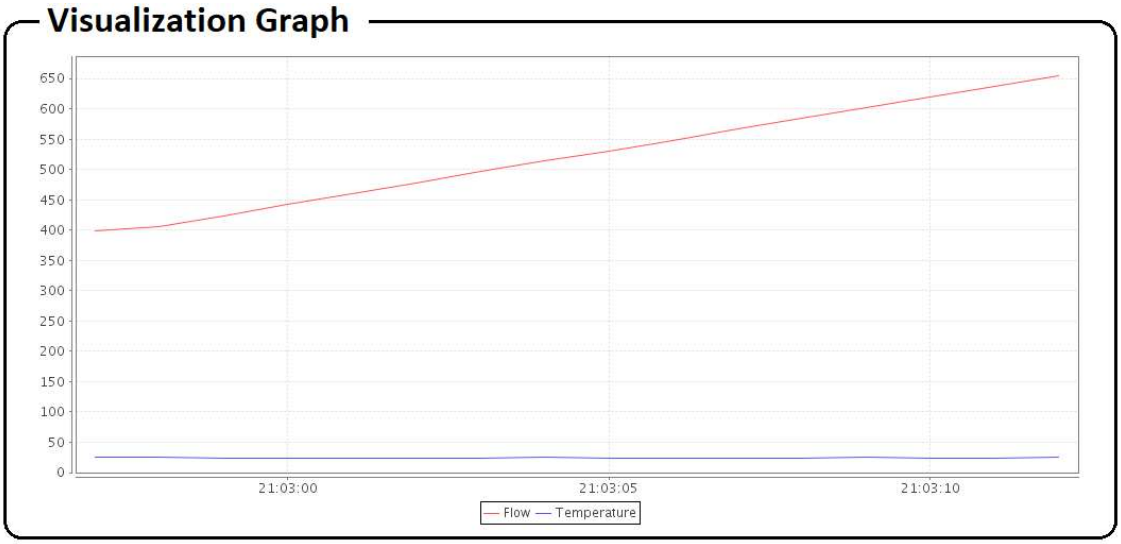
The following modifications could be made by an attacker which would cause undesirable results.

1. Attack 1: Set the temperature setpoint to 120.00°C



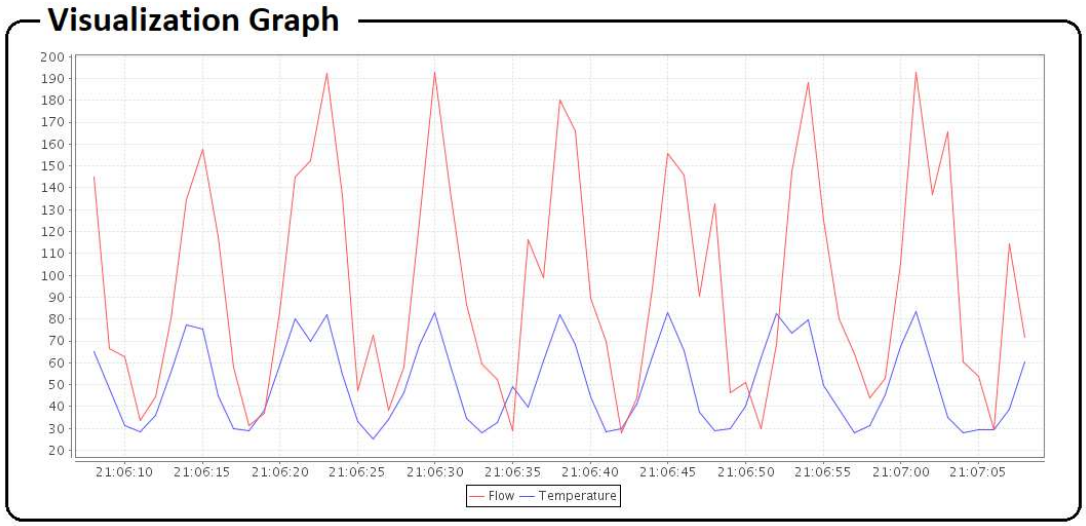
Expected result: The cold mass input flow to keep the temperature at 120°C is too low. The PID algorithm cannot keep a steady state.

2. Attack 2: Set the temperature setpoint to 20°C.



Expected Result: It is physically impossible to reach any temperature colder than the cold input stream. The PID algorithm tries to increase the flow infinitely and only accumulates error.

3. Attack 3: Set the temperature setpoint to 50°C and the proportional gain (Kp) to 1.8.



Expected Result: The proportional gain is too high which causes the system to oscillate.

EXERCISE #4 - INTRO TO LADDER LOGIC

This exercise will introduce basic Ladder Logic and familiarize students with how Ladder Logic can be created using the PLCopen Editor. The PLCopen Editor is a software that enables you to write PLC programs, which contains the logic that the PLC must execute.

This lab will be conducted using the virtual machine and Heat Exchanger Docker Containers

SECTION 1: It was verified in the previous section that if the user inserts an invalid temperature setpoint (out of the 25°C - 70°C range) the system becomes unstable. On this section you will modify the Heat Exchanger Ladder Logic to add a temperature setpoint protection mechanism.

1. Open a terminal (Applications -> Terminal Emulator) and navigate to PLCOpen Editor folder:

```
cd /home/ccre/scadalab/lab1/editor/
```

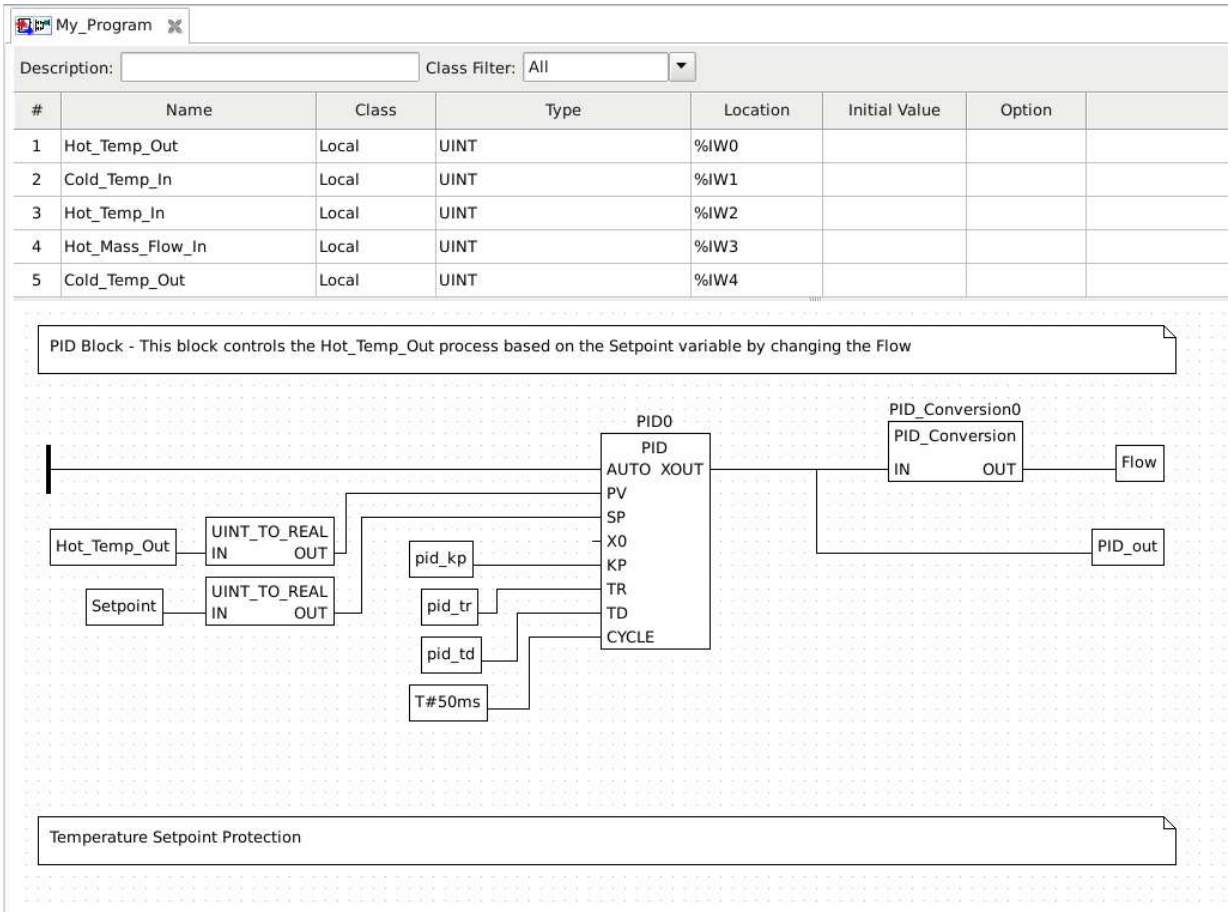
2. Run PLCopen Editor with the command:

```
python PLCOpenEditor.py
```

3. In the PLCopen Editor, select File>Open. Navigate to ccre/scadalab/lab1 folder (you can find the ccre folder on the left sidebar). Double click the heat_exchanger.xml file in the Lab 1 Directory.

4. Double-click on "My Program" to add in the new logic.

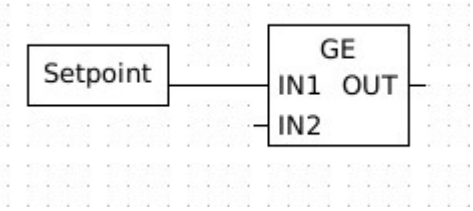
5. Select the "CMT" icon from the menu bar and draw a new block at bottom and add "Temperature Setpoint Protection" in the Comments. Select OK.



6. Right-click and select "Add>Block". In the "Block Properties" box, expand Comparison and select GE (Greater than or Equal). Select OK.

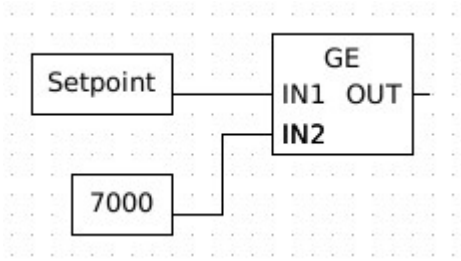
7. Right-click and select "Add>Variable". Set Expression = Setpoint. Select OK.

8. Attach line from variable "Setpoint" to "IN1" on the Comparison block.



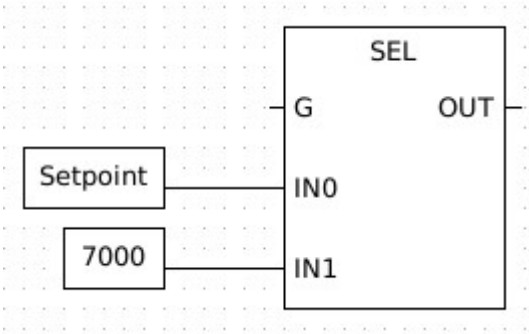
9. Right-click and select "Add>Variable", Set Expression = 7000 (Note: Equal to 70.00°C). Select OK.

10. Attach line from variable "7000" to "IN2" on the Comparison block.

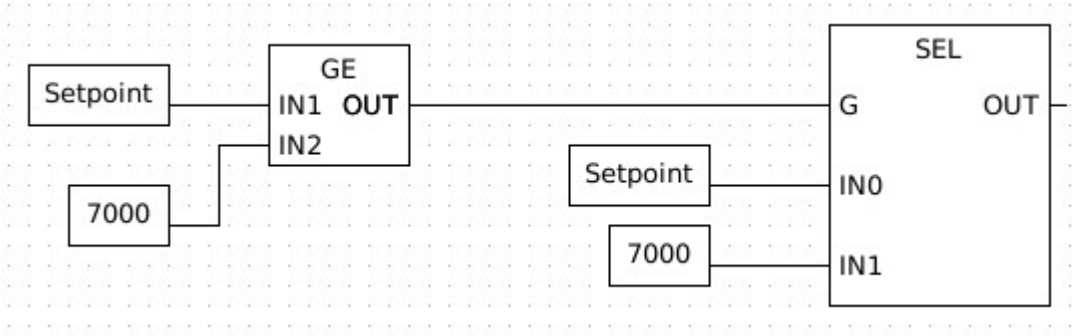


11. Right-click and select "Add>Block". In the "Block Properties" box, expand Selection and select SEL (Binary selection). Select OK.

12. Repeat the "Add>Variable" process to add another Setpoint variable and a 7000 variable and connect them to the IN0 and IN1 inputs of the SEL block.

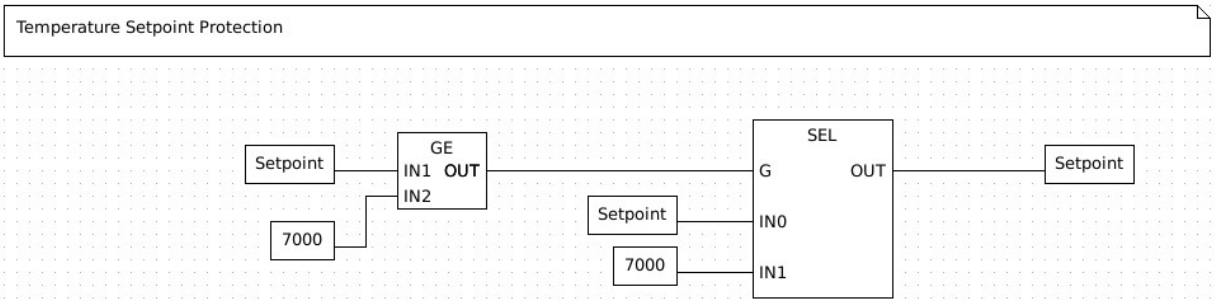


13. Connect the OUT pin of the GE block to the G pin of the SEL block.



14. To the right of the SEL block, Right-click and select "Add>Variable". In the "Class" drop down menu, select Output. Set Expression = Setpoint and click OK.

15. Connect the output Setpoint variable to the OUT pin of the SEL block. Your final Ladder logic will look like the image below:



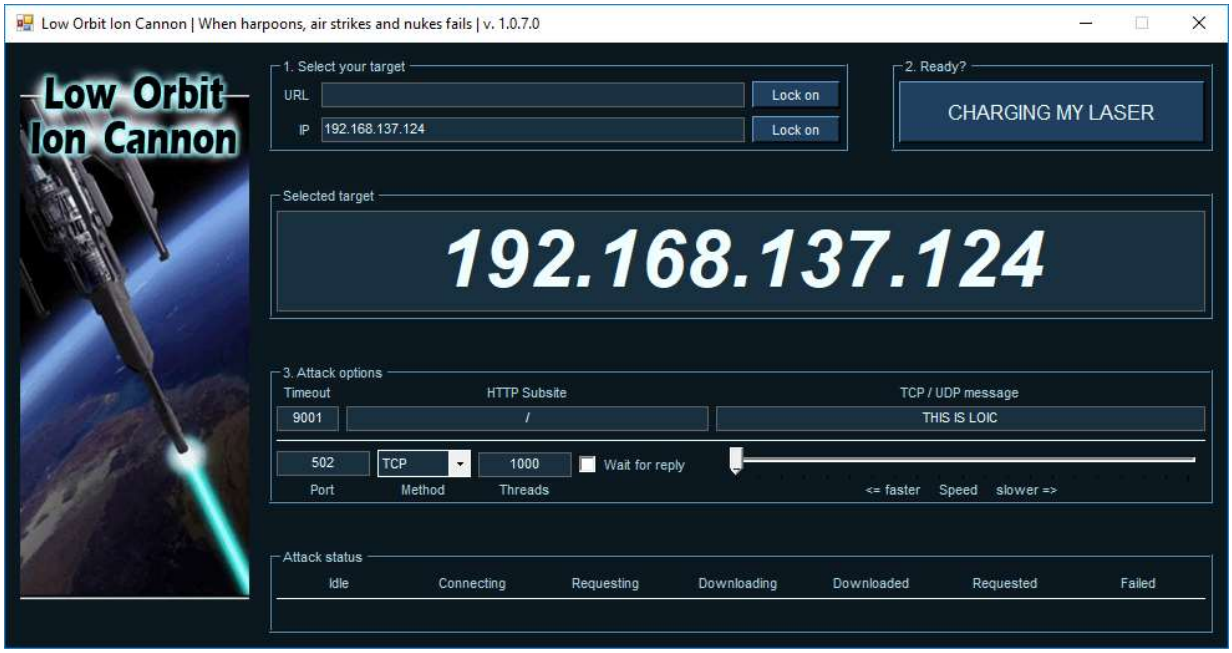
16. Save file.
17. Select File>Generate Program and save file as heat_protected.st in the folder ccre/scadalab/lab1
18. Launch the internet browser and go to 100.100.100.5:8080
19. Click "Choose File" and grab the heat_protected.st file created. Click Upload Program. You should receive a "Program compiled without errors" message. (Disregard any errors regarding conversion.)
20. Open ScadaBR again. If closed, launch the internet browser on the virtual machine and navigate to 100.100.100.2:8080/ScadaBR and login to ScadaBR (username:admin, password:admin)
21. Test the protection by changing the temperature setpoint to anything greater than 70°C.

EXERCISE #5 – TAKING A TARGET DOWN

This exercise will introduce the concepts of Denial of Service (DoS) and Distributed Denial of Service (DDoS). Both techniques have the purpose of taking a target down by flooding the target with a large amount of data in a short time. The Low Orbit Ion Cannon (LOIC) will be used to perform the attacks for this exercise.

All students will target the heat exchanger PLC running on the lecturer's computer. At first, only one student at a time will be allowed to attack the simulation (DoS). Then, all students in the class will perform the attack at the same time (DDoS).

1. Download LOIC from here and extract the contents of the zip file on a folder.
2. Open LOIC.exe. On the main window, insert the target IP and click on Lock on.
3. Under "Attack options" type 502 on port, select Method "TCP", type 1000 in Threads, and uncheck the "Wait for reply" option.
4. Click on "CHARGING MY LASER" and observe the results on the lecturer computer.

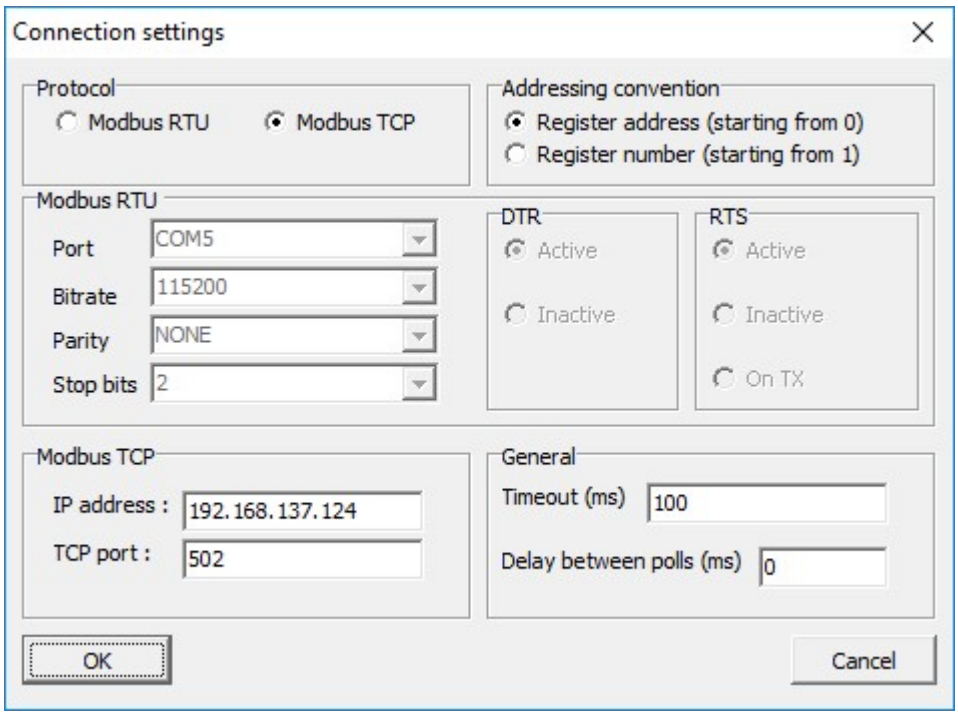


5. Verify the behavior of the system when only one student is performing the attack and when all students are attacking at once.

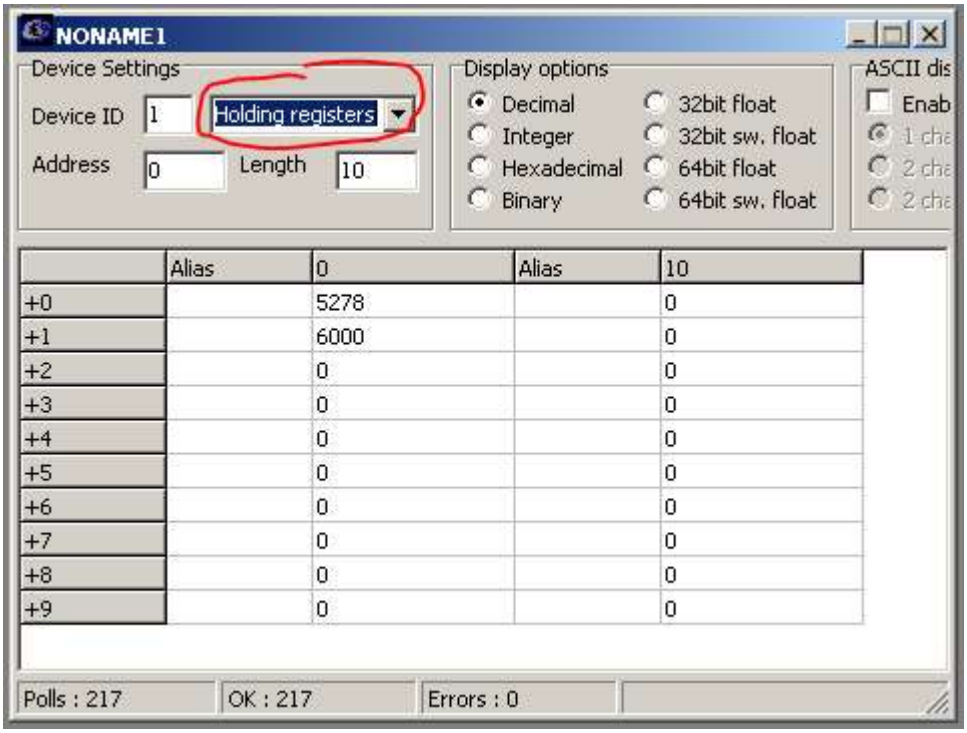
EXERCISE #6 – INJECTION ATTACK

On this exercise students will perform an injection attack by fabricating messages with different settings and sending them to the target PLC. Students will use the Radzio! software to fabricate the messages. The target will be the heat exchanger PLC running on the lecturer's computer. At first, only one student at a time will be allowed to attack the simulation. Then, all students in the class will perform the attack at the same time.

1. Download Radzio! from here and extract the contents of the zip file on a folder.
2. Open RMMS.exe. On the main window, go to Connection->Settings. Select Modbus TCP under "Protocol", Register address starting from 0 under "Addressing convention", and type the target PLC address on "IP address: " field. Also, make sure that the TCP port is 502.



3. Click on File->New and Connection->Connect. On the new spreadsheet that appears, select Holding registers to view the PLC memory data



4. The number on the second line of the spreadsheet (+1) has the temperature setpoint multiplied by 100: 6000 = 60°C. Change the temperature settings by double-clicking on the second line and inserting a new value.
5. Evaluate the change of behavior on the simulation after modifying the settings with the fabricated message. Try to use some of the values suggested on Exercise #3.

ACKWOWLEDGEMENTS

This lab was developed at the University of Alabama in Huntsville by Stefanie Smith, Ben McGee, Thiago Alves, Joseph Lee, and Tommy Morris.

OpenPLC is a completely open programmable logic controller with development environment, human machine interface, programmable logic controller source code, and reference hardware available at http://www.openplcproject.com/. The OpenPLC project was founded by Thiago Alves of the University of Alabama in Huntsville.

The Simulink models, human machine interface implementation, and ladder logic program for the gas pipeline and water storage tank test beds used on this laboratory exercise are copyrighted property of the University of Alabama in Huntsville.