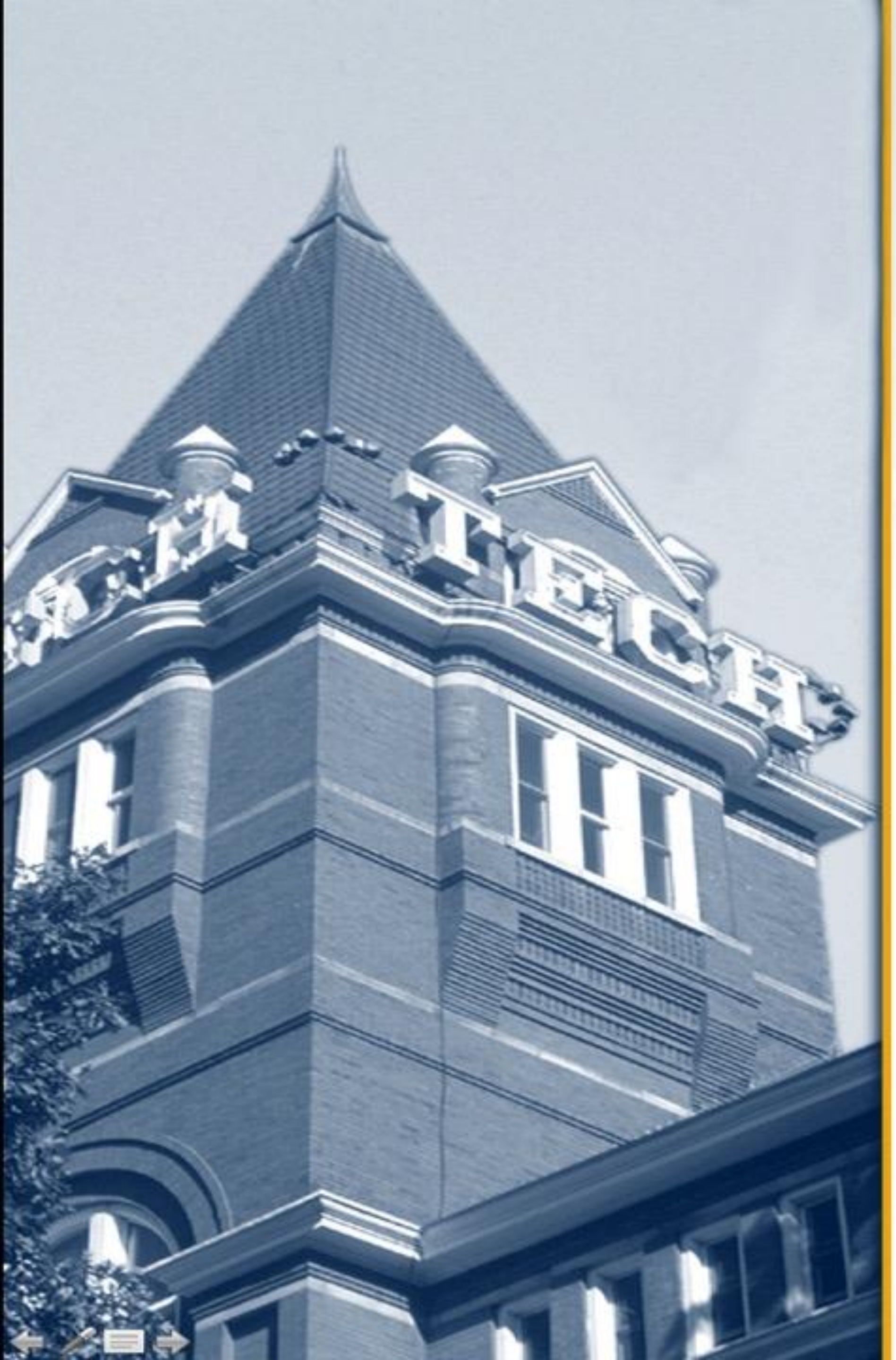


Lowering the Barriers to Industrial Control System Security with GRFICS

David Formby, Milad Rad, and Raheem Beyah

Georgia Institute of Technology

FORTIPHYD LOGIC

Introduction

- Large cybersecurity skills gap even with good tools
  - Metasploitable, Kali Linux, previous ASE work
- ICS security gap even larger
  - Expensive equipment and software
  - Expensive and dangerous to practice physical attacks
  - No ICS equivalent to Metasploitable
- ICS personnel misinformed about security
  - “Air gap”, not a target, not possible

Related Work

- Hardware testbeds
  - Singapore University of Technology
    - Water treatment facility, water distribution network, and small scale electric power grid network.
  - Department of Energy - SCADA testbed
  - Not scalable
- Virtual
  - OpenPLC
  - Not convincing

ICS Background

- Insecure by design
  - No/weak passwords, password policies
  - No message authentication
  - Life cycle > 10 years
  - Difficult to patch
- Network defenses critical
- Physical “exploit”

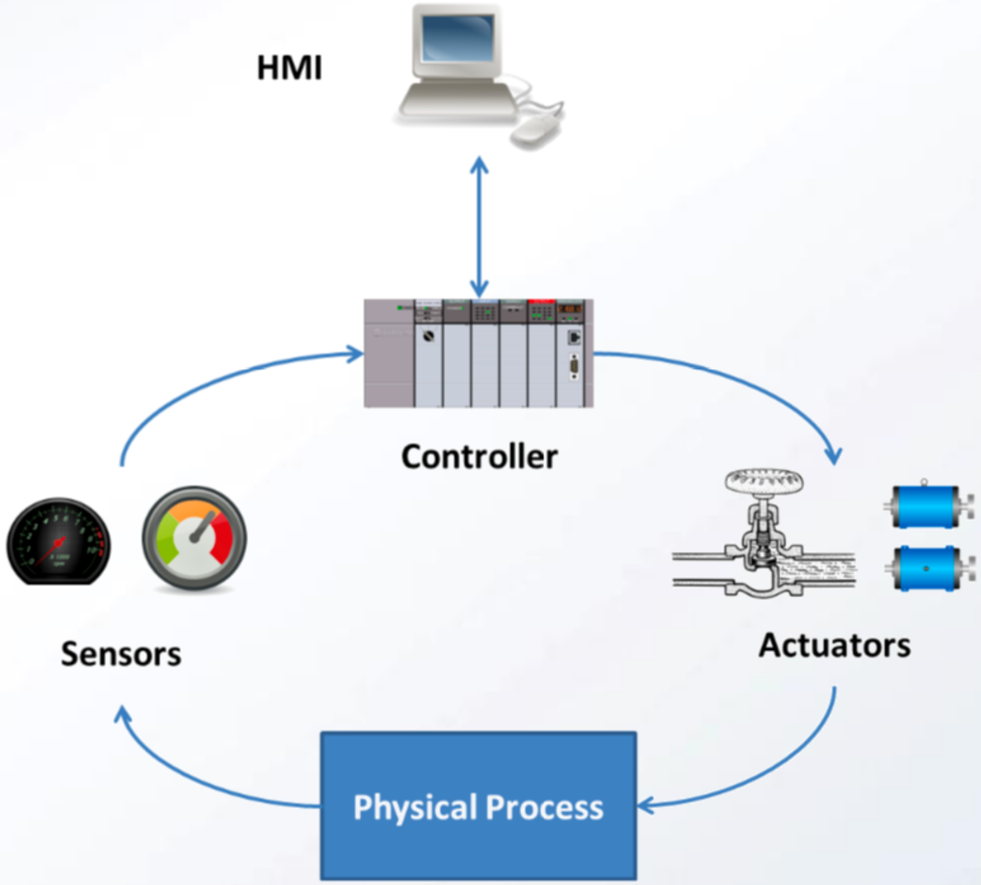


Figure 1: High level structure of ICS network

Programmable Logic Controller (PLC) Background

- Essentially industrialized microcontroller
  - Ruggedized, real-time constraints
  - Control physical equipment
- Programming languages
  - Ladder logic – graphical, like hardware relays
  - Structured text – C like language
  - Instruction list – assembly like language
  - Function block diagram
  - Sequential function charts

GRFICS Overview

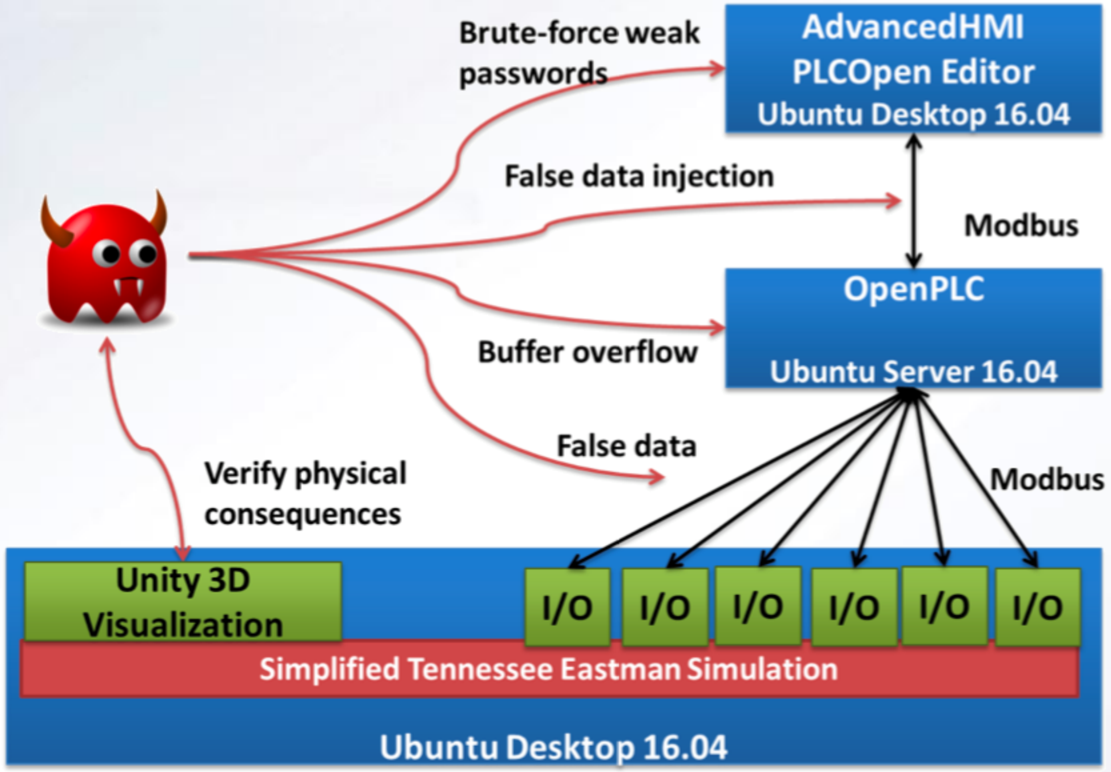


Figure 3: Architecture of GRFICS framework

GRFICS Overview

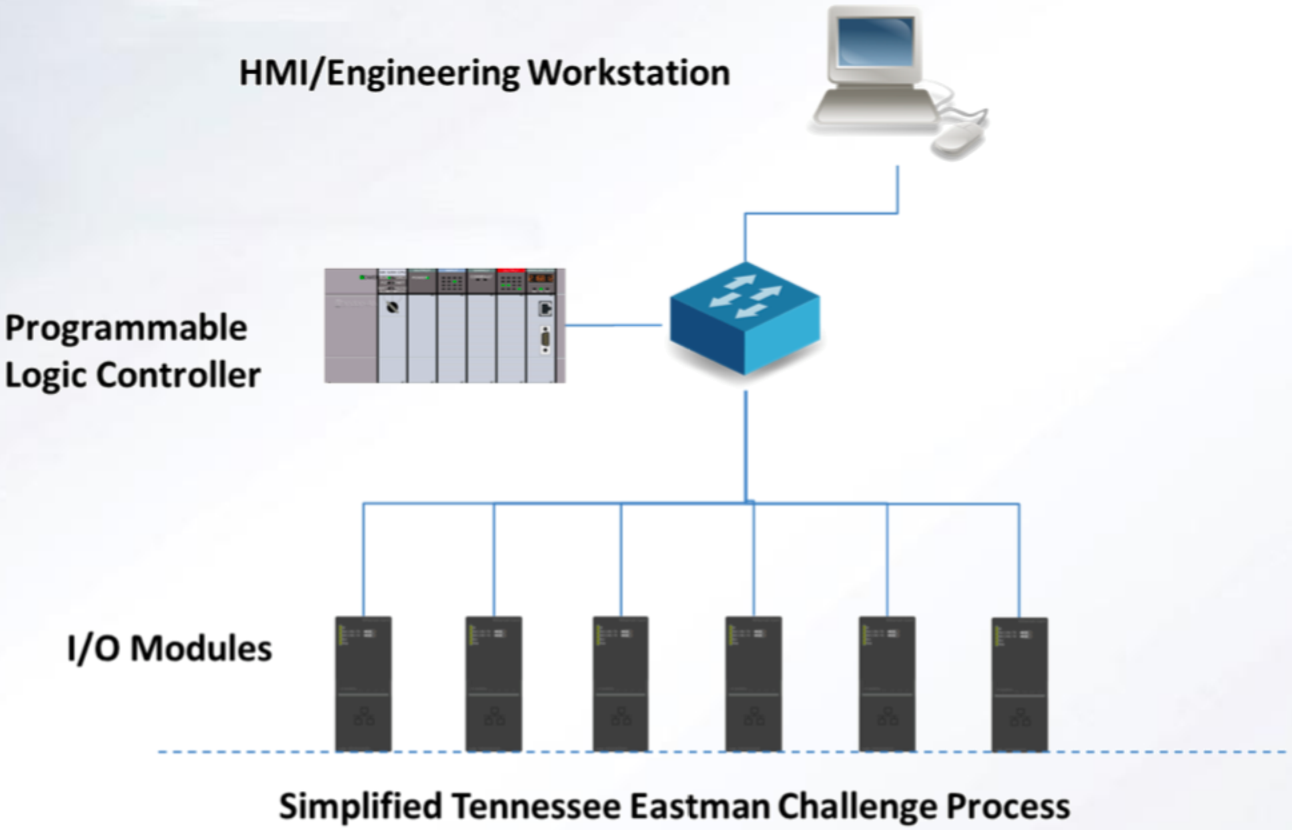

HMI/Engineering Workstation
Programmable Logic Controller
I/O Modules
Simplified Tennessee Eastman Challenge Process

Figure 2: Network Diagram for Virtualized Network

Physical Process Simulation

- Tennessee Eastman Challenge Process
  - Exothermic chemical reactor simulation
  - Originally for process control engineers, in Fortran
  - Two input feeds, product output and purge valve
  - More efficient at higher pressure
- Key measurements
  - Reactor pressure and level
  - Cost – i.e. how much is wasted through purge
- C++, JSON API over port 55555

Simplified Tennessee Eastman Chemical Process

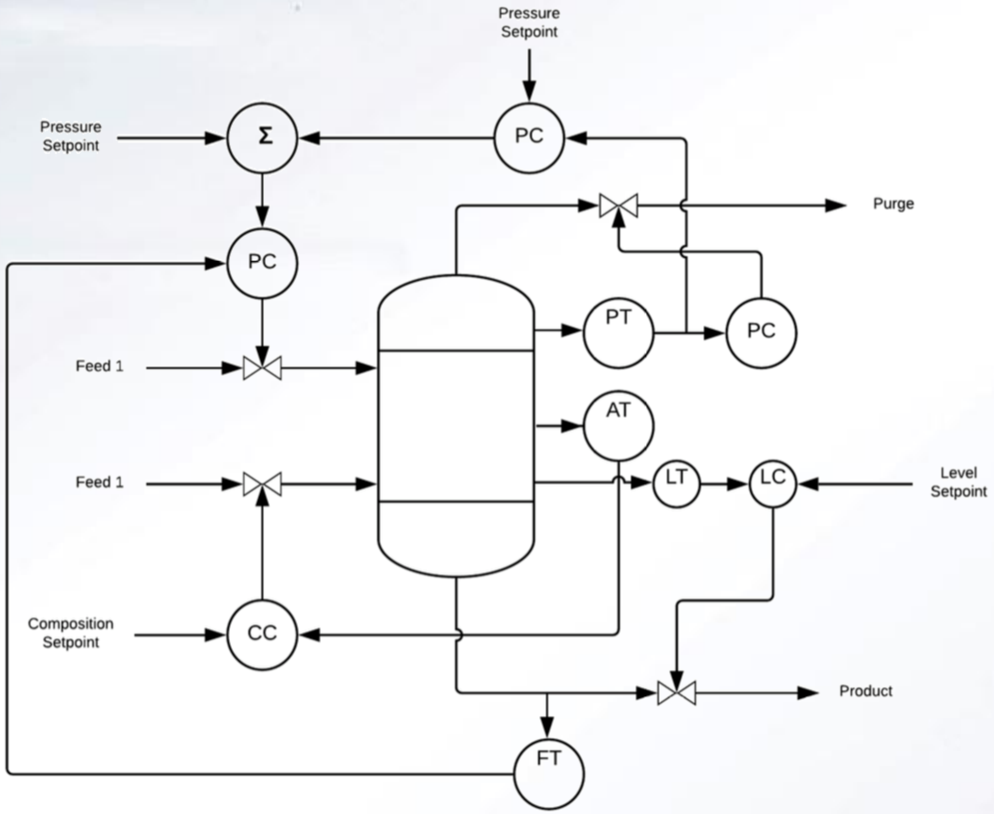
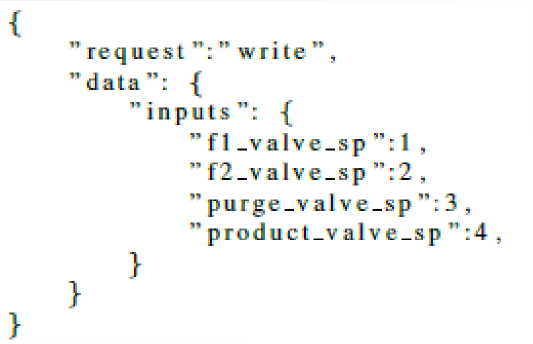


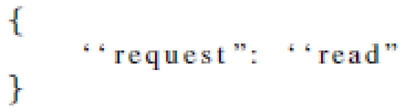
Figure 4: Piping & Instrumentation Diagram (P&ID) for the simplified Tennessee Eastman Challenge Process

Remote IO Layer

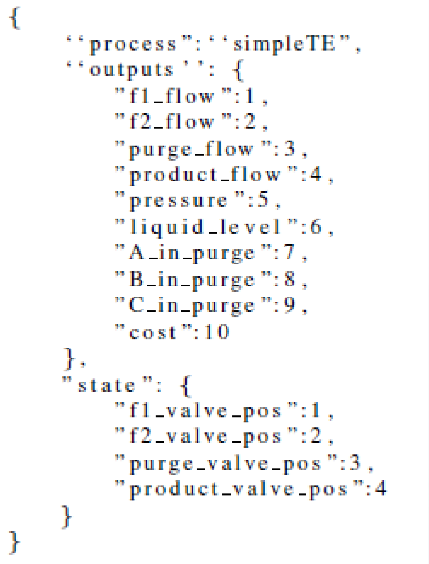
- Modbus servers on 6 IP address aliases
  - Report measurements to PLC
  - Take commands from PLC to control valves
- JSON API
  - Get current values from simulation and update valves



```
{
    "request":"write",
    "data": {
        "inputs": {
            "f1_valve_sp":1,
            "f2_valve_sp":2,
            "purge_valve_sp":3,
            "product_valve_sp":4,
        }
    }
}
```



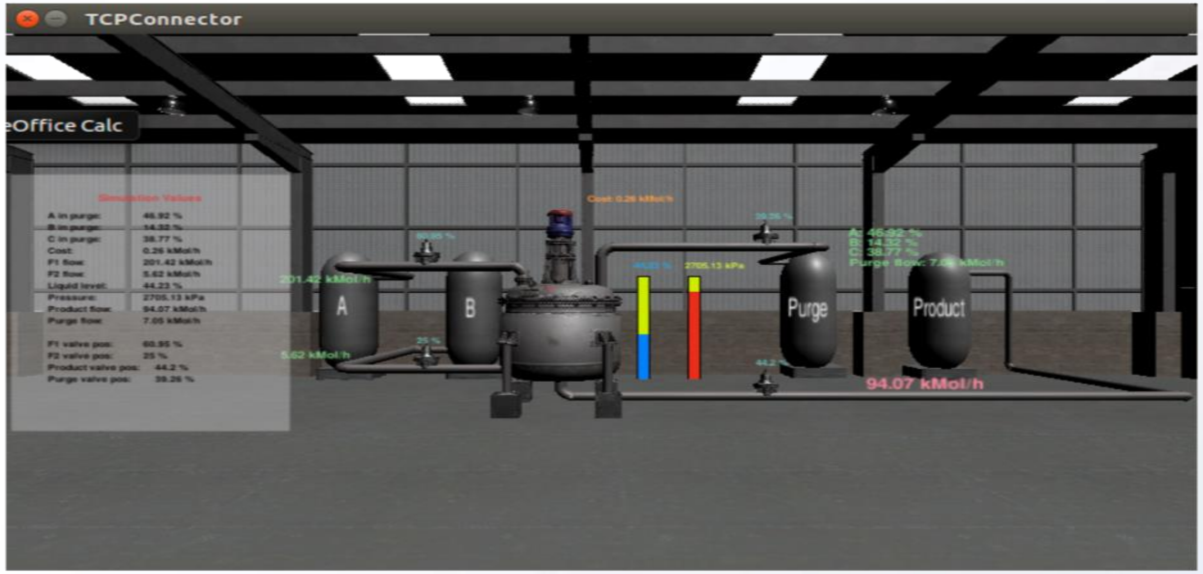
```
{
        ‘‘request”:  ‘‘read”
}
```



```
{
    ‘‘process”:‘‘simpleTE”,
    ‘‘outputs’’: {
        "f1_flow":1,
        "f2_flow":2,
        "purge_flow":3,
        "product_flow":4,
        "pressure":5,
        "liquid_level":6,
        "A_in_purge":7,
        "B_in_purge":8,
        "C_in_purge":9,
        "cost":10
    },
    "state": {
        "f1_valve_pos":1,
        "f2_valve_pos":2,
        "purge_valve_pos":3,
        "product_valve_pos":4
    }
}
```

Process Simulation

- Unity 3D Game Engine
  - Built-in physics engine for collisions
  - Popular with active and supportive community
- GRFICS
  - Purchased 3D models of reactor, pipe, valves, warehouse
  - Get values using JSON API
  - Overlay current values and summarize in sidepane



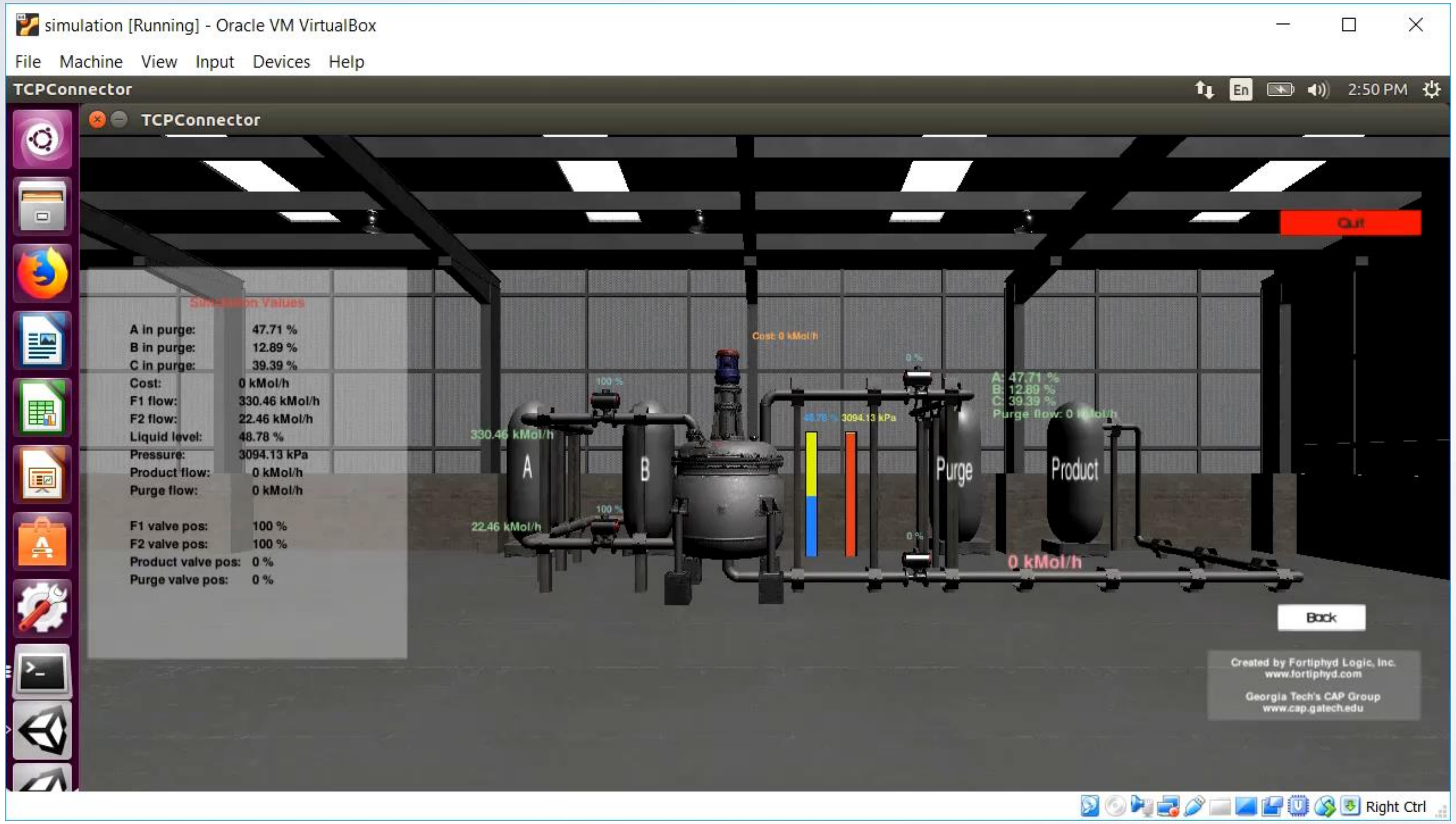
```
Simulation Values

A in purge:            46.92 %
B in purge:            14.32 %
C in purge:            38.77 %
Cost:                  0.26 kMol/h
F1 flow:               201.42 kMol/h
F2 flow:               5.62 kMol/h
Liquid level:          44.23 %
Pressure:              2705.13 kPa
Product flow:          94.07 kMol/h
Purge flow:            7.05 kMol/h

F1 valve pos:          60.95 %
F2 valve pos:          25 %
Product valve pos:     44.2 %
Purge valve pos:       39.26 %
```

Figure 7: “Normal” Operation

Visualization of Successful Attacks



Simulation Values

```
A in purge:         47.71 %
B in purge:         12.89 %
C in purge:         39.39 %
Cost:             0 kMol/h
F1 flow:          330.46 kMol/h
F2 flow:          22.46 kMol/h
Liquid level:     48.78 %
Pressure:         3094.13 kPa
Product flow:       0 kMol/h
Purge flow:         0 kMol/h

F1 valve pos:       100 %
F2 valve pos:       100 %
Product valve pos:  0 %
Purge valve pos:    0 %
```

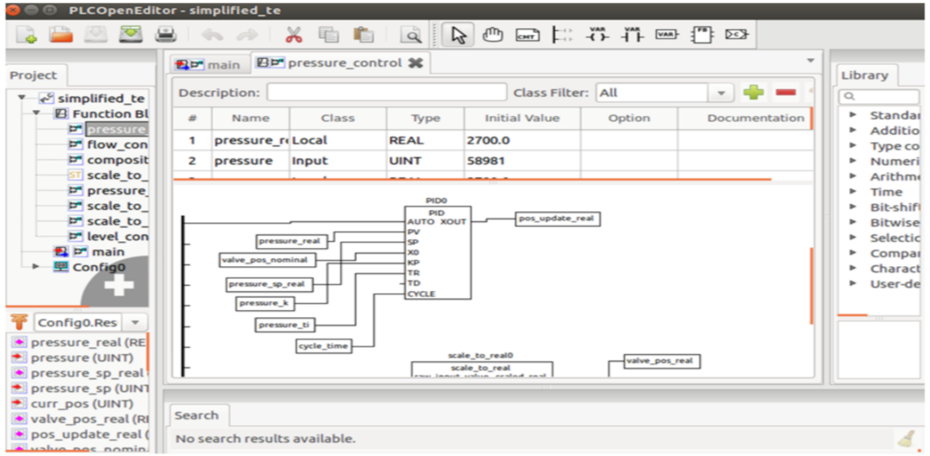
FORTIPHYD LOGIC

Georgia Tech

OpenPLC and libmodbus

- OpenPLC – Open source software PLC
  - Primarily speaks Modbus
    - Old, common, super simple
    - Move raw data using registers
- Buffer overflow for libmodbus <= v3.0.2
  - Mismatch in max number of bytes and number of registers requested
  - Binary data, no need to encode payload
- Standard Debian package

Engineering Workstation/HMI

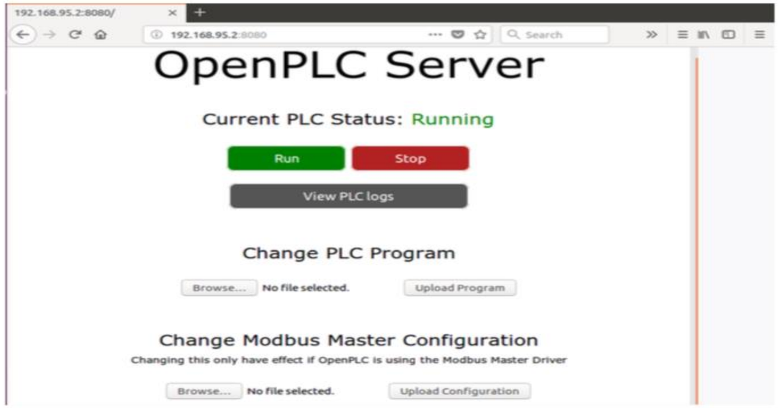


Description: Class Filter: All

#	Name	Class	Type	Initial Value	Option	Documentation
1	pressure_r	Local	REAL	2700.0		
2	pressure	Input	UINT	58981		

No search results available.

Figure 9: PLCOpen Editor



OpenPLC Server

Current PLC Status: Running

Change PLC Program

No file selected.

Change Modbus Master Configuration

Changing this only have effect if OpenPLC is using the Modbus Master Driver

No file selected.

Figure 10: OpenPLC Web Interface

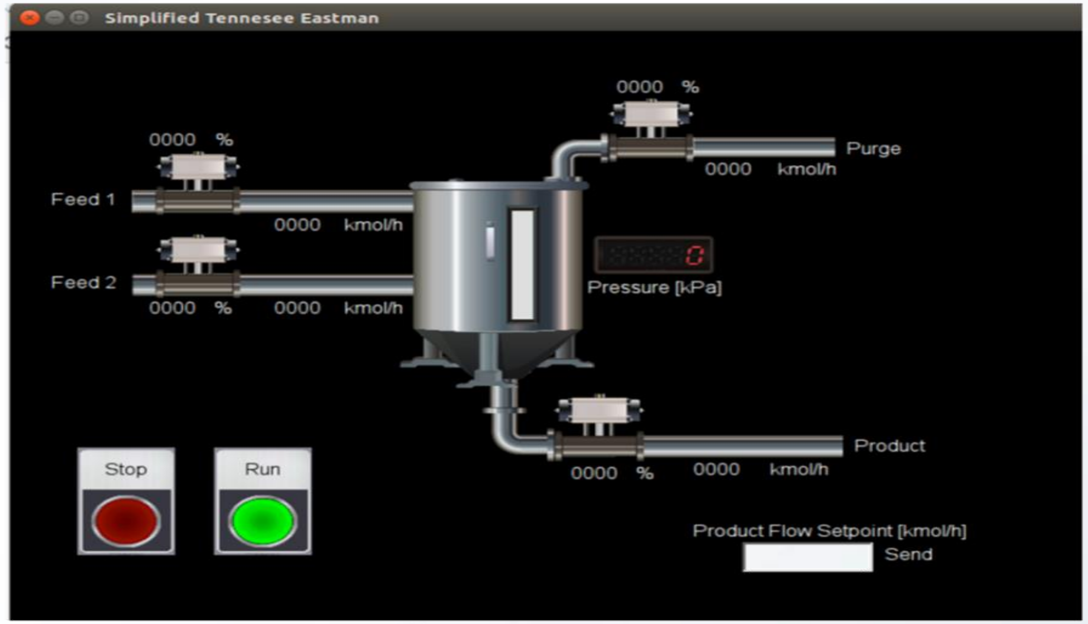


Figure 11: Operator Human Machine Interface (HMI)

FORTIPHYD LOGIC

Georgia Tech

Example Attacks

- MITM
- Command injection
- False data injection
- Reprogramming PLC
  - Stuxnet
- Loading malicious binary payload
  - TRITON
- Common IT attacks
  - Password cracking
  - Buffer overflow

FORTIPHYD
LOGIC

Georgia Tech

Example Defenses

- Network segmentation
  - ISA 95 Reference model
- IDS/IPS
  - Snort rules to detect and/or stop buffer overflow

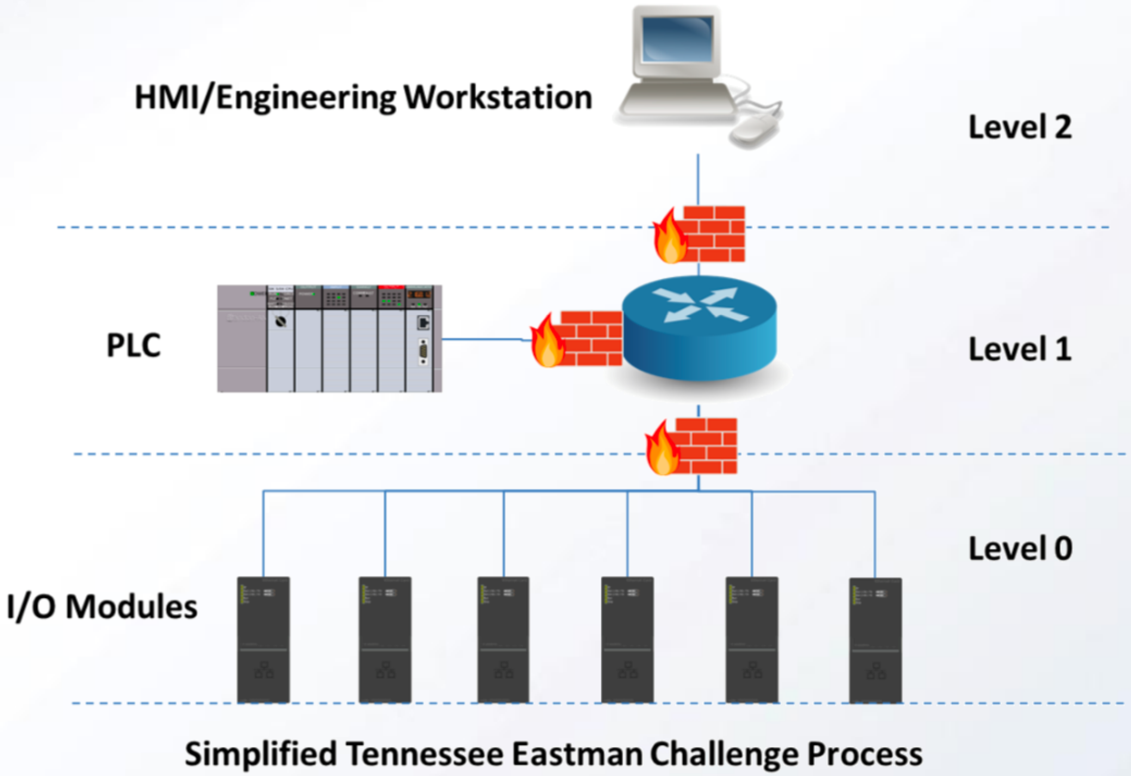

HMI/Engineering Workstation
Level 2
PLC
Level 1
Level 0
I/O Modules
Simplified Tennessee Eastman Challenge Process

Figure 13: Segmented network architecture according to Purdue Reference Model

FORTIPHYD LOGIC

Georgia Tech

Discussion

- Installation
  - Tedious if from scratch, also have pre-built VMs for download
  - Good hardware required – 30GB HDD, 8GB RAM, Quad core
- Fidelity
  - Simplified simulation, open source quality not industrial quality
  - Good enough physics for CS background
- Performance
  - Unity visualization can get slow under attacks

Conclusions and Future Work

- ICS security skills gap is larger due to higher barriers to entry
- GRFICS
  - Free and open source
  - Improved realism and engagement over previous work
- Future work
  - Incorporate into graduate level “Cyber physical system security” class
  - Improve fidelity
    - “Real” ICS software, larger network
  - Add more scenarios

Questions?

Thank you!

David Formby

djformby@gatech.edu dformby@fortiphyd.com

@fortiphyd

FORTIPHYD LOGIC

Georgia Tech