

410.1

ICS Overview

SANS

Copyright © 2019, Secure Anchor Consulting. All rights reserved to Secure Anchor Consulting and/or SANS Institute.

PLEASE READ THE TERMS AND CONDITIONS OF THIS COURSEWARE LICENSE AGREEMENT ("CLA") CAREFULLY BEFORE USING ANY OF THE COURSEWARE ASSOCIATED WITH THE SANS COURSE. THIS IS A LEGAL AND ENFORCEABLE CONTRACT BETWEEN YOU (THE "USER") AND THE SANS INSTITUTE FOR THE COURSEWARE. YOU AGREE THAT THIS AGREEMENT IS ENFORCEABLE LIKE ANY WRITTEN NEGOTIATED AGREEMENT SIGNED BY YOU.

With the CLA, the SANS Institute hereby grants User a personal, non-exclusive license to use the Courseware subject to the terms of this agreement. Courseware includes all printed materials, including course books and lab workbooks, as well as any digital or other media, virtual machines, and/or data sets distributed by the SANS Institute to the User for use in the SANS class associated with the Courseware. User agrees that the CLA is the complete and exclusive statement of agreement between The SANS Institute and you and that this CLA supersedes any oral or written proposal, agreement or other communication relating to the subject matter of this CLA.

BY ACCEPTING THIS COURSEWARE, YOU AGREE TO BE BOUND BY THE TERMS OF THIS CLA. BY ACCEPTING THIS SOFTWARE, YOU AGREE THAT ANY BREACH OF THE TERMS OF THIS CLA MAY CAUSE IRREPARABLE HARM AND SIGNIFICANT INJURY TO THE SANS INSTITUTE, AND THAT THE SANS INSTITUTE MAY ENFORCE THESE PROVISIONS BY INJUNCTION (WITHOUT THE NECESSITY OF POSTING BOND), SPECIFIC PERFORMANCE, OR OTHER EQUITABLE RELIEF.

If you do not agree, you may return the Courseware to the SANS Institute for a full refund, if applicable.

User may not copy, reproduce, re-publish, distribute, display, modify or create derivative works based upon all or any portion of the Courseware, in any medium whether printed, electronic or otherwise, for any purpose, without the express prior written consent of the SANS Institute. Additionally, User may not sell, rent, lease, trade, or otherwise transfer the Courseware in any way, shape, or form without the express written consent of the SANS Institute.

If any provision of this CLA is declared unenforceable in any jurisdiction, then such provision shall be deemed to be severable from this CLA and shall not affect the remainder thereof. An amendment or addendum to this CLA may accompany this courseware.

SANS acknowledges that any and all software and/or tools, graphics, images, tables, charts or graphs presented in this courseware are the sole property of their respective trademark/registered/copyright owners, including:

AirDrop, AirPort, AirPort Time Capsule, Apple, Apple Remote Desktop, Apple TV, App Nap, Back to My Mac, Boot Camp, Cocoa, FaceTime, FileVault, Finder, FireWire, FireWire logo, iCal, iChat, iLife, iMac, iMessage, iPad, iPad Air, iPad Mini, iPhone, iPhoto, iPod, iPod classic, iPod shuffle, iPod nano, iPod touch, iTunes, iTunes logo, iWork, Keychain, Keynote, Mac, Mac Logo, MacBook, MacBook Air, MacBook Pro, Macintosh, Mac OS, Mac Pro, Numbers, OS X, Pages, Passbook, Retina, Safari, Siri, Spaces, Spotlight, There's an app for that, Time Capsule, Time Machine, Touch ID, Xcode, Xserve, App Store, and iCloud are registered trademarks of Apple Inc.

PMP and PMBOK are registered marks of PMI.

SOF-ELK® is a registered trademark of Lewes Technology Consulting, LLC. Used with permission.

SIFT® is a registered trademark of Harbingers, LLC. Used with permission.

Governing Law: This Agreement shall be governed by the laws of the State of Maryland, USA.

WELCOME TO ICS410 ICS/SCADA SECURITY ESSENTIALS

As you are waiting for class to begin, please

- Copy all files from the course USB to your laptop
 - If you are using Linux on your host, you must have ExFAT drivers installed
- Read the file named "ICS410 Student Instructions.pdf"
- Import both virtual machines (the two OVA files) into VMware
 - Start with the Windows 10 VM since we will use that one first
 - If importation immediately fails, select the retry option once or twice
 - Still failing? Try recopying the files from a different course USB
 - Still? Ask in instructor or facilitator for help
- Ensure both virtual machines are working in VMware on your laptop
- Instructions for the Velocio PLC will be provided later in the course

This page intentionally left blank.



ICS Overview

© 2019 Justin Searle | All Rights Reserved | Version E01_01

The **SANS ICS410**, ICS/SCADA Security Essentials course, was developed by a collection of experts whose diverse work experiences, knowledge, and skills truly blend together to cover the very specific content areas for this course.

Justin Searle is the Director of ICS Security at InGuardians, specializing in ICS security architecture design and penetration testing. Justin led the Smart Grid Security Architecture group in the creation of NIST Interagency Report 7628 and has played key roles in the Advanced Security Acceleration Project for the Smart Grid (ASAP-SG), National Electric Sector Cybersecurity Organization Resources (NESCOR), and Smart Grid Interoperability Panel (SGIP). Justin has taught courses in hacking techniques, forensics, networking, and intrusion detection for multiple universities, corporations, and security conferences. He is currently a Senior Instructor for the SANS Institute. In addition to electric power industry conferences, Justin frequently presents at top international security conferences such as Black Hat, DEFCON, OWASP, Nullcon, and AusCERT. Justin co-leads prominent open source projects including the The Control Thing Platform, Samurai Web Testing Framework (SamuraiWTF), Samurai Security Testing Framework for Utilities (SamuraiSTFU), Yokoso!, and Laudanum. Justin has an MBA in International Technology and is a CISSP and SANS GIAC certified Incident Handler (GCIH), Intrusion Analyst (GCIA), Web Application Penetration Tester (GWAPT), and GIAC Industrial Control Security Professional (GICSP).

Dr. Eric Cole is an industry-recognized security expert with over 20 years of hands-on experience. Dr. Cole has experience in information technology with a focus on helping customers focus on the right areas of security by building out a dynamic defense. Dr. Cole has a master's degree in computer science from NYIT and a doctorate from Pace University with a concentration in information security. He served as CTO of McAfee and Chief Scientist for Lockheed Martin. Dr. Cole is the author of several books, including *Advanced Persistent Threat*, *Hackers Beware*, *Hiding in Plain Sight*, *Network Security Bible*, 2nd Edition, and *Insider Threat*. Eric is the inventor of over 20 patents and is a researcher, writer, and speaker. He is also a member of the Commission on Cyber Security for the 44th President and several executive advisory boards. Dr. Cole is the founder and an executive leader at Secure Anchor Consulting, where he provides leading-edge cybersecurity consulting services, expert witness work, and leads research and development initiatives to advance the state-of-the-art in information systems security. Dr. Cole is actively involved with the SANS Technology Institute (STI). He is a SANS faculty Fellow who works with students, teaches, and develops and maintains courseware.

Eric Cornelius is the Director of Critical Infrastructure and Industrial Control Systems (ICS) at Cylance, Inc. He is responsible for the thought leadership, architecture, and consulting implementations for the company. His leadership keeps organizations safe, secure, and resilient against advanced attackers. Previously, Eric served as the Deputy Director and Chief Technical Analyst for the Control Systems Security Program at the US Department of Homeland Security. As an active researcher in the field of cybersecurity since 2002, Eric supported many "boots-on-the-ground" engagements involving penetration testing, forensics, and malware analysis. Through these engagements, he aided multiple government, military, and private sector organizations in protecting their networks and industrial control systems. In addition to his years of technical leadership, Eric literally wrote the book on incident response in the ICS arena. Eric's extensive knowledge of critical infrastructure and those who attack it will be brought to bear at Cylance as he leads a team of experts in securing America's critical systems.

Contributing Authors

Michael Assante is currently the SANS project lead for Industrial Control System (ICS) and Supervisory Control and Data Acquisition (SCADA) security. He served as Vice President and Chief Security Officer of the North American Electric Reliability Corporation (NERC), where he oversaw industry-wide implementation of cybersecurity standards across the continent. Prior to joining NERC, Michael held a number of high-level positions at Idaho National Labs and he served as Vice President and Chief Security Officer for American Electric Power. His work in ICS security has been widely recognized and he was selected by his peers as the winner of *Information Security Magazine's* security leadership award for his efforts as a strategic thinker. The RSA 2005 Conference awarded him its outstanding achievement award in the practice of security within an organization. He has testified before the US Senate and House and was an initial member of the Commission on Cyber Security for the 44th Presidency. Prior to his career in security, Michael served in various naval intelligence and information warfare roles, and he developed and gave presentations on the latest technology and security threats to the Chairman of the Joint Chiefs of Staff, Director of the National Security Agency, and other leading government officials. In 1997, he was honored as a Naval Intelligence Officer of the Year.

Tim Conway is currently the Technical Director of ICS and SCADA programs at SANS. He is responsible for developing, reviewing, and implementing technical components of the SANS ICS and SCADA product offerings. He was formerly the Director of CIP Compliance and Operations Technology at Northern Indiana Public Service Company (NIPSCO) where he was responsible for Operations Technology, NERC CIP Compliance, and the NERC training environments for the operations departments within NIPSCO Electric. Tim was previously an EMS Computer Systems Engineer at NIPSCO for eight years, with responsibility over the control system servers and the supporting network infrastructure. He previously served as the Chair of the RFC CIPC, is the current Chair of the NERC CIP Interpretation Drafting Team, a current member of the NESCO advisory board, the current Chair of the NERC CIPC GridEx 2013 Working Group, and the current Chair of the NBISE Smart Grid Cyber Security panel.

TABLE OF CONTENTS		PAGE
GICSP Overview		6
ICS Concepts		11
EXERCISE 1.1: Learning from Peers		26
Purdue Levels 0 and 1		29
EXERCISE 1.2: Programming a PLC		44
Purdue Levels 2 and 3		64
SCADA		78
EXERCISE 1.3: Programming an HMI		88
IT and ICS Differences		102
Physical and Cybersecurity		116
Secure ICS Network Architectures		128
EXERCISE 1.4: Architecting a Secure DCS		141



This page intentionally left blank.

GOALS OF THIS COURSE

Bring IT and OT professionals together in the same room to discuss

- Diverse set of ICS industry sectors and applications
- Keeping the operational environment safe, reliable, and resilient
- Cybersecurity essentials in control systems
- Basic tools and techniques for cybersecurity in industrial settings

As an industry, we need to learn knowledge and skills needed to safeguard our critical infrastructures

Some here to prepare for the GICSP (Global Industrial Cybersecurity Professional) certification

The ICS410 courseware has been developed to equip both security professionals and control system engineers with the knowledge and skills they need to safeguard our critical infrastructures. This course provides students with the essentials for conducting security work in industrial control system (ICS) environments. Students will learn the language, the underlying theory, and the basic tools for ICS security in industrial settings across a diverse set of industry sectors and applications. This course will introduce students to ICS and provide the necessary information and learning to secure control systems while keeping the operational environment safe, reliable, and resilient.

Course Roadmap

Day 1: ICS Overview

Day 2: Field Devices and Controllers

Day 3: Supervisory Systems

Day 4: Workstations and Servers

Day 5: ICS Security Governance

1. Introduction
2. GICSP Overview
3. ICS Concepts
 - Processes and Roles
 - Industries
 - **Exercise 1.1: Learning from Peers**
4. Purdue Levels 0 and 1
 - Controllers and Field Devices
 - Programming Controllers
 - **Exercise 1.2: Programming a PLC**
5. Purdue Levels 2 and 3
 - HMIs, Historians, Alarm Servers
 - Special Applications and Master Servers
 - Control Rooms and Plants
 - SCADA
 - **Exercise 1.3: Programming an HMI**
6. IT and ICS Differences
 - ICS Life Cycle Challenges
7. Physical and Cybersecurity
8. Secure ICS Network Architectures
 - ICS410 Reference Model
 - Design Example
 - **Exercise 1.4: Architecting a Secure DCS**

This page intentionally left blank.

WHAT IS THE GICSP?

The GICSP is a new certification that focuses on the foundational knowledge that professionals securing critical infrastructure assets should know.

Holders of the GICSP will demonstrate a globally recognized level of competence that defines the architecture, design, management, risks, and controls that assure the security of critical infrastructure.

The GICSP is the bridge to bring together IT, engineering, and cybersecurity professionals to achieve security for ICS from design through retirement.

The GICSP is expected to be adopted on a global basis as a gateway certification for critical infrastructure-industrial control system professionals



This page intentionally left blank.

GICSP IS A GLOBAL INITIATIVE

The GICSP was developed and guided by a steering committee of leaders from major industrial companies, advisors, and suppliers from around the world, and it includes members from the following companies:



- Shell
- ABB
- Emerson
- British Petroleum
- KPMG
- Wurldtech (now GE)
- Chevron
- Saudi Aramco
- Invensys (now Siemens)
- Pacific Gas and Electric
- SANS-GIAC
- Rockwell Automation

This page intentionally left blank.

GICSP – CERTIFICATION OBJECTIVES

Knowledge, skill, and ability in the domains of:

- Architecture
- Assessments
- Configuration and Change Management
- Log Collection and Management
- Business Continuity
- Incident Management
- Information Risk and Security Management
- Physical Security
- Industrial Control Systems
- System Hardening
- Cybersecurity Essentials
- Access Management



This page intentionally left blank.

GICSP GLOBAL INDUSTRIAL CYBERSECURITY PROFESSIONAL



This certification will be leveraged across industries to ensure a minimum set of knowledge and capabilities that IT, engineer, and security professionals should know if they are in a role that could impact the cybersecurity of an ICS environment.

1

The Test

- 115 questions, 100 scored
- Open book
- 3-hour exam

2

Scheduling

- 2 practice tests
- Available 7–10 days after ICS410
- Available immediately for challenge
- Available at Pearson Vue
- Must take within 120 days

3

Maintenance

- Renewal every 4 years
- Test retake or earn CPEs
- Continuing Professional Education credits

For more information on GICSP certification, please visit:

<https://www.giac.org/certification/global-industrial-cyber-security-professional-gicsp>

Test delivery is computer-based and proctored by Pearson VUE at over 3,400 global testing centers.

Certification is valid for 4 years; continuing professional education requirements are consistent with GIAC standards.

Reference:

<http://www.giac.org/certifications/renewal>

Course Roadmap

Day 1: ICS Overview

Day 2: Field Devices and Controllers

Day 3: Supervisory Systems

Day 4: Workstations and Servers

Day 5: ICS Security Governance

1. Introduction
2. GICSP Overview
3. ICS Concepts
 - Processes and Roles
 - Industries
 - **Exercise 1.1: Learning from Peers**
4. Purdue Levels 0 and 1
 - Controllers and Field Devices
 - Programming Controllers
 - **Exercise 1.2: Programming a PLC**
5. Purdue Levels 2 and 3
 - HMIs, Historians, Alarm Servers
 - Special Applications and Master Servers
 - Control Rooms and Plants
 - SCADA
 - **Exercise 1.3: Programming an HMI**
6. IT and ICS Differences
 - ICS Life Cycle Challenges
7. Physical and Cybersecurity
8. Secure ICS Network Architectures
 - ICS410 Reference Model
 - Design Example
 - **Exercise 1.4: Architecting a Secure DCS**

This page intentionally left blank.

ICS Concepts

Control systems and their processes, professions, and industries

This page intentionally left blank.

WHAT IS A CONTROL SYSTEM?

A device, or set of devices, that manages, commands, directs, or regulates the behavior of other devices or systems

- A device that can influence the "real world"
- A system that bridges cyber-to-physical

Simple example: Thermostats in our homes

- Keep temperatures at desired temperatures
- Turn on/off furnace and air conditioners
- What can an attacker do if he gains control of this?

Industrial control systems (ICS)

- Exponentially larger, more complex, and more dangerous

Control systems are defined here as "a device, or set of devices, that manages, commands, directs, or regulates the behavior of other devices or systems." This can take a variety of shapes, from a large chemical processing plant to the system controlling your gas furnace at home. The system takes a set of input data from its devices (such as a thermostat), performs logic on that data (the temperature is 70°F when it should be 72°F), and activates components to affect that sensor data (I'll turn on the fireplace so it heats back up to 72°F).

This is a gross simplification, but for example, what kind of logic is required to be energy efficient so the heat of the house isn't constantly bouncing between 72°F and 68°F as the furnace kicks on and off again? Or, what if the temperature sensor fails and reports the room to be 0°F when it is really 78°F? What kind of mechanisms are in place to handle this?

What happens if the thermostat is IP-based and an attacker manages to fool the system into thinking it's 70°F when it's really 5°F?

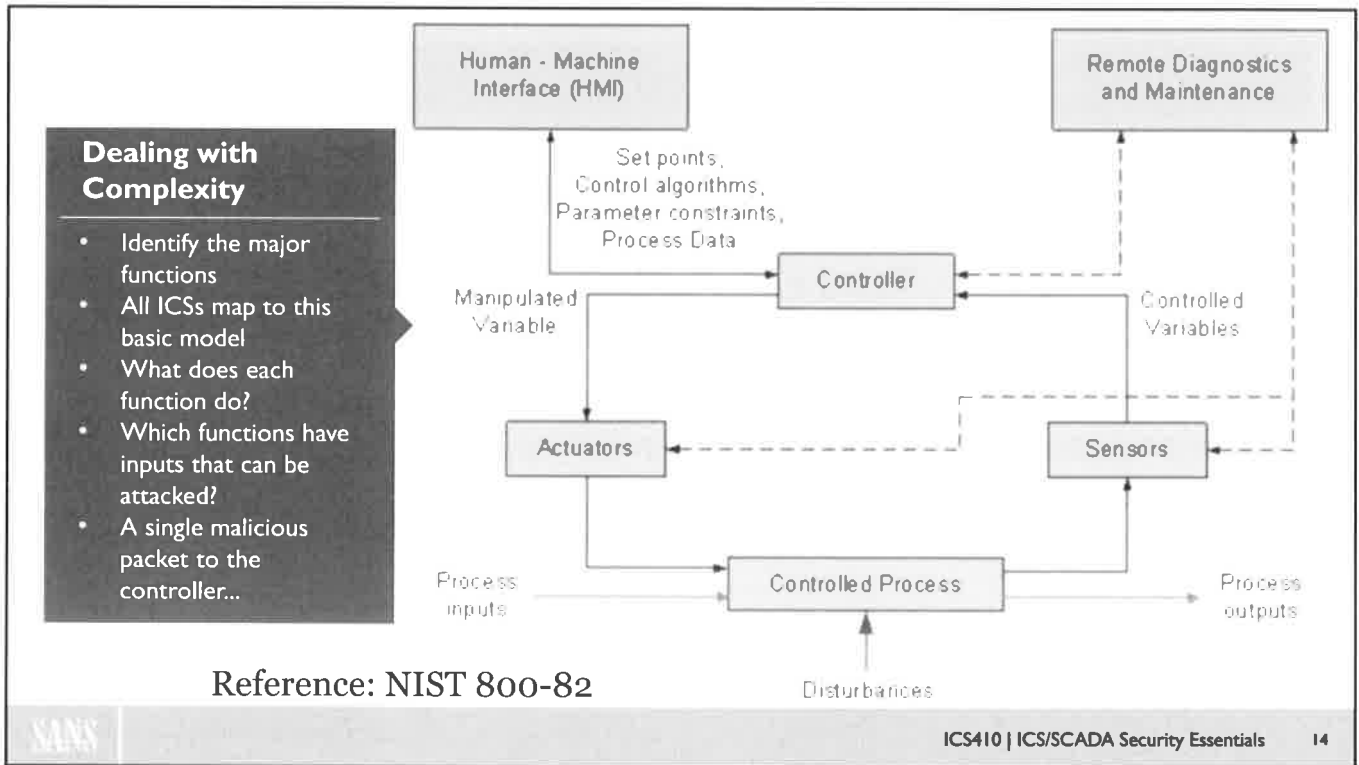
Simply, a control system gathers information and then performs a function based on established parameters and/or information it received. ICS can vary in size and complexity based upon the process it is responsible for monitoring and controlling. ICS can be found in very specific applications (e.g., found on a small skid-mounted system) or manage something as large as a multiple-unit generation facility or oil refinery.

By the end of this course, you'll be familiar with the concepts that drive these questions and the mechanisms by which they are answered.

You will understand the common drivers for ICS technology: Reliability, efficiency, safety, and ease of use.

Reference:

ISA-62443.01.01 definitions, <http://std.iec.ch/glossary>



The logic diagram shown displays typical inputs and outputs to the various control system components as well as typical operations that are performed by the various components.

Control systems can be very difficult and costly to replace and adjust. This is one of the reasons why security in this space is lagging behind. Refreshing a control system is something done very rarely. It is not unusual for a system to remain in place for 20+ years without many changes.

General size of points monitored or controlled:

- Small: 1–2 Workstations, 1–2 Controllers, 0–599 points
- Medium: 3–8 Workstations, 3–8 Controllers, 600–1,499 points
- Large: 8+ Workstations, 8+ Controllers, 1,500+ points

Image Source: NIST SP 800-82

<https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-82r2.pdf>

ICS PROCESS MODELS

There are four main ICS process models

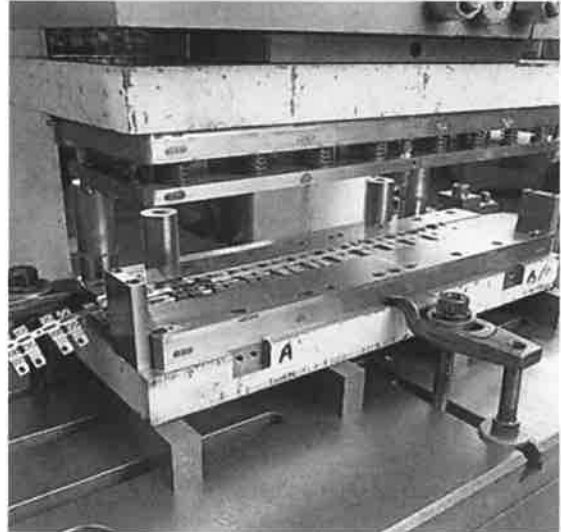
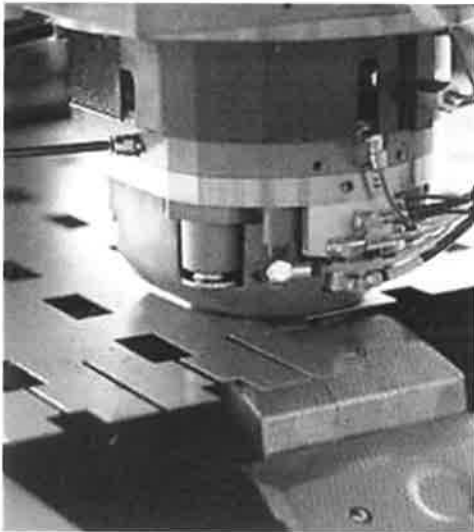
- Discrete
- Batch
- Continuous
- Hybrid

These models have general definitions and are the basic precept for the type of process to be instrumented, monitored, and controlled by an ICS

We will review the four main ICS process models (discrete, batch, continuous, and hybrid). These models have general definitions and are the basic precept for the type of process to be instrumented, monitored, and controlled by an ICS.

*some things happen
over and over
again*

ICS DISCRETE PROCESSES



SIAT

ICS410 | ICS/SCADA Security Essentials

16

The photos above show metal stamping, which is known as a discrete process.

It is a type of process where a specified quantity of material moves as a unit between workstations, and each unit maintains its unique identity.

This process is used in many manufacturing, motion, and packaging applications. Robotic assembly, such as that found in automotive production, can be characterized as a discrete process control. Most discrete manufacturing involves the production of discrete pieces of product, such as metal stamping (license plates).

ICS BATCH PROCESSES



SANS

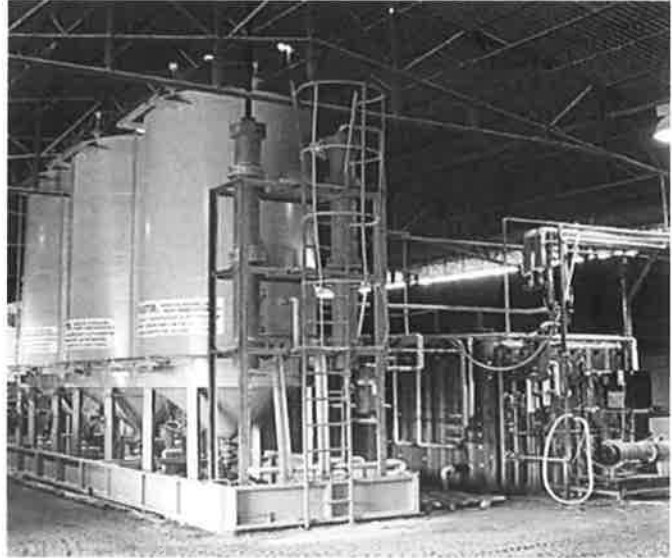
ICS410 | ICS/SCADA Security Essentials

17

The photos above show adhesive manufacturing and liquid medicine manufacturing: both batches are processes.

Batch: Some applications require that specific quantities of raw materials be combined in specific ways for particular durations to produce an intermediate or end result. One example is the production of adhesives and glues, which normally require the mixing of raw materials in a heated vessel for a period of time to form a quantity of end product. Other important examples are the production of food, beverages, and medicine. Batch processes are generally used to produce a relatively low to intermediate quantity of product per year (a few pounds to millions of pounds).

ICS CONTINUOUS PROCESSES



SANS

ICS410 | ICS/SCADA Security Essentials

18

The photos above show chlorine chemical production and fuel production; both are continuous processes.

A physical system is often represented through variables that are smooth and uninterrupted in time. The control of the water temperature in a heating jacket is an example of continuous process control. Some important continuous processes are the production of fuels, chemicals, and plastics. Continuous processes in manufacturing are used to produce very large quantities of product per year (millions to billions of pounds).

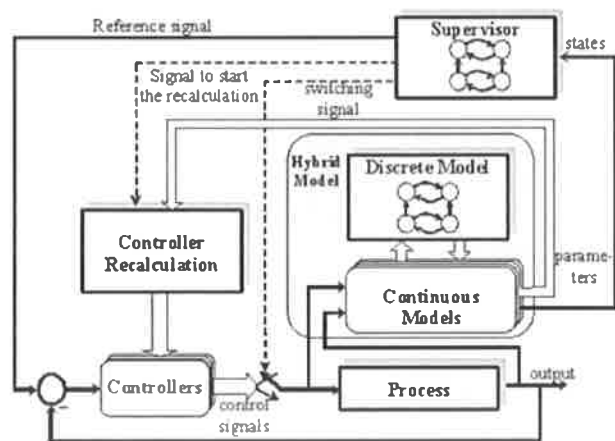
waste water, electricity... continuously flow systems.

ICS HYBRID PROCESSES

A hybrid process is a combination of discrete and continuous components

Tend to be hierarchical

We usually defend at this level



Hybrid systems are generally understood as reactive systems that intermix discrete and continuous components. Hybrid control systems are typically found when continuous processes interact with or are supervised by sequential machines. A hybrid process model allows the controller to optimize the process based on many variables that may change the efficiency of the process at any given moment. Examples of such systems include flexible manufacturing and chemical process control systems, interconnected power systems, intelligent vehicle highway systems, and air traffic management systems.

Image Source:

<http://proaut.ziti.uni-heidelberg.de/joomla/index.php/en/research/projects/finished-projects/116-hybrid-modelling-and-control-of-industrial-processes>

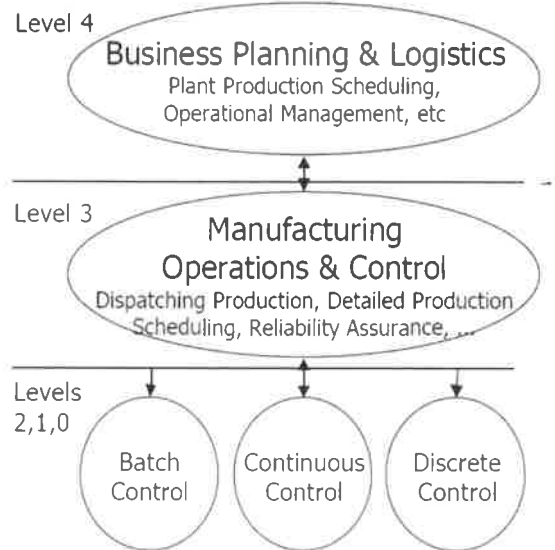
PURDUE ENTERPRISE REFERENCE ARCHITECTURE (PERA)

One of the first reference architectures for industrial control networks

- 4: Business network (not a control network)
- 3: Plant-wide control network
- 2: Individual process/cell/line supervisory
- 1: Individual process/cell/line controllers
- 0: Individual process/cell/line sensors and actuators

Original source for many later reference architectures

- ISA-95
- ISA-99 which became ISA/IEC 62443
- ICS410 Reference Model



SINS

ICS410 | ICS/SCADA Security Essentials

20

The Purdue Enterprise Reference Architecture (PERA) was one of the first reference architectures for ICS networks. It was designed by a public/private consortium made up of individuals from the industry and Purdue University. This effort was led by Theodore J. Williams, a professor of chemical and electrical engineering programs at Purdue. This model has become very popular for designing ICS networks and has been adopted and expanded by later reference models.

The Purdue model suggests dividing your systems into five different levels, numbered zero through four. Level 0 is the lowest level, representing the physical process and the immediate sensors and actuators that make up that process. Level 1 represents the controllers. Level 2 is the supervisory level, where our master servers, Human Machine Interfaces (HMIs), and historians sit. Level 3 is the operations support, containing systems that support operations but that are not directly involved with the real-time execution of the process. And finally, Level 4 represents the business side of the organization related to ICS.

Image Source: https://en.wikipedia.org/wiki/Purdue_Enterprise_Reference_Architecture

ICS PROFESSIONALS' ROLES AND RESPONSIBILITIES

The role of people cannot be overstated

People fill the roles that can't be automated

- **Process Engineer:** Designs and optimizes
- **Field Technician:** Maintains and repairs
- **Programmer:** Writes control logic
- **Operator:** Manages and controls process

People are used to filling in parts of the process that cannot be automated. From a security perspective, people provide an excellent attack surface. We will cover this deeper as the course continues. Common roles for people in the ICS environment are:

- Process Engineer: Designs the systems and processes used in the control environment
- Field Technician: Maintains and repairs field devices
- Programmer: Implements the individual steps of the process as code and deploys the code onto the controllers
- Operator: Works in the operations center, remotely managing and controlling the process

*Take the logic
and make logic
work program
the logic*

These traditional roles must be empowered and equipped to integrate with information technology and cybersecurity roles to tackle the challenges of complexity, change, and cyber threats.

ICS ORGANIZATIONS' ROLES AND RESPONSIBILITIES

ICS roles and responsibilities are divided among several different participants

- **Owner/Operators:** Purchase and use the system
 - Bear primary responsibility for safe operation and meet regulations
- **Vendors:** Manufacture equipment and software
 - ABB, Alstom, Areva, Emerson, GE, Hitachi, Honeywell, Mitsubishi, Rockwell Automation, Schneider Electric, SEL, Siemens, Yokogawa, etc.
- **Integrators:** Design, configure, test, train, and refresh
 - Often provided by vendors or partners of vendors
- **Government:** Implement guidance and regulation
 - Primary goal is to safeguard the public good
 - Examples: NERC CIP, NIST, ENISEA
 - In some instances, are owner/operators as well

Asset owner/operators use control systems as a part of their business. Owner/operators come in many shapes and sizes, from the small mom-and-pop shop to the large multinational corporations. Asset owners and operators must work with vendors and integrators to design and build systems that are efficient and meet government regulations. In recent years, security has become more prominent in the public eye and has received increased scrutiny from government regulators. Asset owners and operators currently bear the burden of making sure security is built into the systems they operate.

Vendors design and build the components used in control systems, such as Programmable Logic Controllers (PLCs) and HMIs. Many vendors work with specific hardware platforms and often will certify products to run on specific models, with specific configurations and with firmware. Many vendors have several different product lines and have millions upon millions of lines in their codebases. Despite the fact that nearly every ICS vendor has begun to adopt security into their development life cycles, their legacy codebases coupled with the slow refresh cycles in most control settings pose a very real security challenge for the ICS community. Most vendors offer integration services for their products as well.

Integrators have the job of selecting, creating, and configuring control systems: This can be a long and arduous process involving many man-hours. Integrators also provide training and support, ongoing control, integration, and automation collaboration, and handle legacy migration and replacement. Legacy migration can take as much time as the rest of them.

sometimes owner-regulatory body
The government is itself an asset owner/operator, as it uses a great number of control systems that do everything from printing money to controlling weapons systems. The government is also responsible for creating and enforcing regulations regarding the safe and continued operation of critical facilities. For example, in the United States, organizations such as DHS, the NRC, and FERC are examples of government bodies that regulate control system operators. The Department of Commerce through the National Institute of Standards and Technology (NIST) sets technical standards for use in the United States. NIST can be found at <http://www.nist.gov/index.html>. The European Union Agency for Network and Information Security (ENISA) can be found at <https://www.enisa.europa.eu>.

INDUSTRIES RELIANT ON ICS

Factory automation segments:

- Aerospace and defense, automotive, electrical electronics and semiconductors, machinery, fabricated metals, furniture and wood products, plastics and rubber, and medical products

Process industry segments:

- Cement and glass, chemical and petrochemical, food and beverage, metals (production), pharmaceuticals, pulp and paper, textiles, and waste and water

Energy industry segments:

- Electric power (generation, transmission, and distribution), oil and gas (exploration, production), oil and gas (pipeline), and oil and gas (refining)

Service industry segments:

- Retail, wholesale (Walmart), transportation (trains, ships, airports, etc.), and logistics

There are many types of industries that rely on ICS, including:

- **Factory automation segments:** Aerospace and defense, automotive, electrical electronics and semiconductors, machinery, fabricated metals, furniture and wood products, plastics and rubber, and medical products.
- **Process industry segments:** Cement and glass, chemical and petrochemical, food and beverage, metals (production), pharmaceuticals, pulp and paper, textiles, and waste and water.
- **Energy sector:** Most nations describe an energy sector that includes all elements: Oil, gas, electricity, and nuclear. Other nations break them into electric power and oil and gas. The European Union defines energy sector with an exclusion of nuclear assets. The US has a very complex definition that combines oil, gas, and electric but excludes commercial nuclear power plants and hydro facilities and dams. Newer definitions are beginning to call out renewable energy technologies as its own subcategory in Energy (e.g., wind farms, solar plants, etc.) Energy sector components are typically highly dependent on other energy sector infrastructure components for the transportation of primary fuels (oil, gas, coal) or for the transportation and storage of energy products being delivered to users. Oil and gas infrastructures are highly dependent on global shipping infrastructure.
- **Service industry segments:** Retail, wholesale (e.g., Walmart), transportation (trains, ships, airports, etc.), and logistics. *amazon, fed-ex, UPS, ... only service industry*

CRITICAL INFRASTRUCTURE

Many ICS industries are often labeled as critical infrastructure

Definitions vary around the world but include:

- Physical and electronic assets of infrastructure
- Communications that enable infrastructure
- Consequences that could impact public safety, economic security, and defense

European Union, India, and US definitions in notes

Often means increased responsibility and regulation

Also means you face nation-state actors . . .

Definitions of critical infrastructure vary around the world, but most have the following three key elements: Physical and electronic assets of infrastructure, communications that enable infrastructure, and consequences that could impact public safety, economic security, and defense.

European Union: *"Critical infrastructure is an asset or system which is essential for the maintenance of vital societal functions. The damage to a critical infrastructure, its destruction or disruption by natural disasters, terrorism, criminal activity or malicious behaviour, may have a significant negative impact for the security of the EU and the well-being of its citizens."* The European Commission – Office of Home Affairs further says: *"The European Programme for Critical Infrastructure Protection (EPCIP) sets the overall framework for activities aimed at improving the protection of critical infrastructure in Europe, across all EU States, and in all relevant sectors of economic activity. The threats to which the programme aims to respond are not only confined to terrorism, but also include criminal activities, natural disasters, and other causes of accidents. In short, it seeks to provide an all-hazards cross-sectoral approach. The EPCIP is supported by regular exchanges of information between EU States in the frame of the CIP Contact Points meetings."*

The government of India has designated the National Critical Information Infrastructure Protection Centre (NCIIPC) of the National Technical Research Organisation (NTRO) as the nodal agency.

The US Department of Homeland Security: *"Critical infrastructure consists of the assets, systems, and networks—whether physical or virtual—so vital to the United States that their incapacitation or destruction would have a debilitating effect on security, national economic security, public health or safety, or any combination thereof."* This is further defined by Presidential Policy Directive 21 (PPD-21).

References:

<http://www.dhs.gov/critical-infrastructure-sectors>

http://ec.europa.eu/dgs/home-affairs/what-we-do/policies/crisis-and-terrorism/critical-infrastructure/index_en.htm

<https://www.homeaffairs.gov.au/about-us/our-portfolios/national-security/security-coordination/security-of-critical-infrastructure-act-2018>

<https://sso.agc.gov.sg/Acts-Supp/9-2018/Published/20180312?DocDate=20180312>

TAKEAWAYS AND RECOMMENDATIONS

Section takeaways

- The Purdue levels are incredibly important to memorize
 - Purdue Level 4: Business network (not a control network) → out of control network.
 - Purdue Level 3: Plant-wide control network
 - Purdue Level 2: Individual process/cell/line supervisory
 - Purdue Level 1: Individual process/cell/line controllers
 - Purdue Level 0: Individual process/cell/line sensors and actuators
- Overcome complexity by mapping components to function model
- Look for the cyber communication inputs
 - That is where attackers attack
 - That is where we need to place defenses

control network.
enterprise IT network

This page intentionally left blank.

- main concern is protecting the input.

Course Roadmap

Day 1: ICS Overview

Day 2: Field Devices and Controllers

Day 3: Supervisory Systems

Day 4: Workstations and Servers

Day 5: ICS Security Governance

1. Introduction
2. GICSP Overview
3. ICS Concepts
 - Processes and Roles
 - Industries
 - **Exercise 1.1: Learning from Peers**
4. Purdue Levels 0 and 1
 - Controllers and Field Devices
 - Programming Controllers
 - **Exercise 1.2: Programming a PLC**
5. Purdue Levels 2 and 3
 - HMIs, Historians, Alarm Servers
 - Special Applications and Master Servers
 - Control Rooms and Plants
 - SCADA
 - **Exercise 1.3: Programming an HMI**
6. IT and ICS Differences
 - ICS Life Cycle Challenges
7. Physical and Cybersecurity
8. Secure ICS Network Architectures
 - ICS410 Reference Model
 - Design Example
 - **Exercise 1.4: Architecting a Secure DCS**

This page intentionally left blank.

EXERCISE 1.1: LEARNING FROM PEERS

DURATION TIME: 15 MINUTES

Time to get to know your neighbors!

Please share with your neighbors 1–2 control systems you've interacted with during your career so far, if possible

Do your best to identify the controller, the process it controlled, and at least one sensor and actuator

OBJECTIVES

Expand your knowledge using your peers as sources

Building your professional connections in the ICS cybersecurity industry

PREPARATION

Break into groups of 4–6 with the people sitting near you

This page intentionally left blank.

EXERCISE 1.1: LEARNING FROM PEERS

TAKEAWAYS AND RECOMMENDATIONS

Section Takeaways:

- There are more categories of control systems and controllers in this world than any one person could ever gain experience on
- Make it a practice to gain experience from everyone you meet in the field
- Learn to work through terminology barriers that exist between ICS industries and even between companies in the same industry
- Never be afraid to admit you don't know or understand something, or you are missing an opportunity to learn it

This page intentionally left blank.

Course Roadmap

Day 1: ICS Overview

Day 2: Field Devices and Controllers

Day 3: Supervisory Systems

Day 4: Workstations and Servers

Day 5: ICS Security Governance

1. Introduction
2. GICSP Overview
3. ICS Concepts
 - Processes and Roles
 - Industries
 - **Exercise 1.1: Learning from Peers**
4. Purdue Levels 0 and 1
 - Controllers and Field Devices
 - Programming Controllers
 - **Exercise 1.2: Programming a PLC**
5. Purdue Levels 2 and 3
 - HMIs, Historians, Alarm Servers
 - Special Applications and Master Servers
 - Control Rooms and Plants
 - SCADA
 - **Exercise 1.3: Programming an HMI**
6. IT and ICS Differences
 - ICS Life Cycle Challenges
7. Physical and Cybersecurity
8. Secure ICS Network Architectures
 - ICS410 Reference Model
 - Design Example
 - **Exercise 1.4: Architecting a Secure DCS**

This page intentionally left blank.

Purdue Levels 0 and 1

Controllers, the field devices they connect to, and everything in between.
Understanding processes and programming controllers.

This page intentionally left blank.

DCS CONTROLLER

The Distributed Control System (DCS) was introduced in 1975

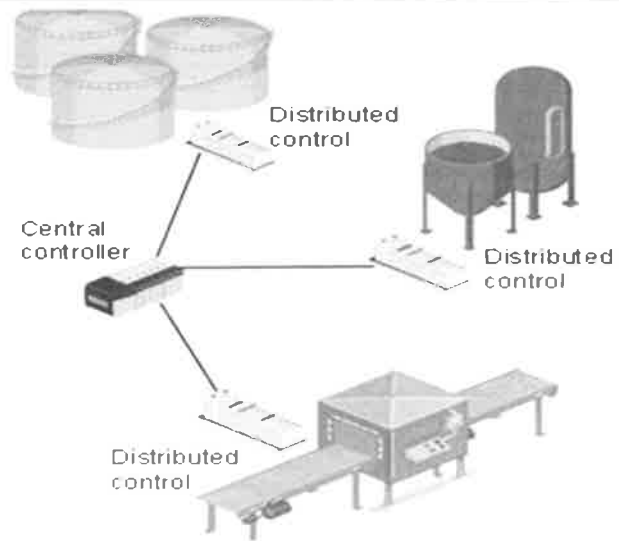
- Custom-designed controllers that often compete with PLC-based solutions
- Cooperatively control complex processes within a large site or plant
- Often highly proprietary interconnects and protocols

Smaller DCS

- 1 to 5 controllers
- I/O in the hundreds

Larger DCS

- More than 5 controllers
- I/O in the thousands



The DCS was introduced in 1975 and largely came about due to the increased availability of computers and the proliferation of microprocessors in the world of process control. A typical DCS consists of functionally and/or geographically distributed digital controllers.

A DCS typically uses custom-designed controllers, proprietary interconnection protocols, and communication protocols. The controller receives information from input modules and sends information to output modules. The input modules receive information from input instruments and transmit instructions to the output instruments in the field. Buses connect the distributed controllers with the central controller and finally to the Human Machine Interface (HMI) or control consoles.

The elements of a DCS can connect directly to physical equipment, such as switches, pumps, and valves, and to an HMI via SCADA. The differences between a DCS and SCADA are often subtle, especially with advances in technology allowing the functionality of each to overlap.

DCSs are dedicated systems typically used to control manufacturing processes that are continuous or batch-oriented, such as oil refining, petrochemicals, central station power generation, fertilizers, pharmaceuticals, food and beverage manufacturing, cement production, steelmaking, and papermaking. DCSs are connected to sensors and actuators to control the flow of materials through the plant. Modern DCSs also support neural networks and fuzzy applications.

DCSs are usually designed with redundant processors to enhance the reliability of the control system. Most systems come with canned displays and configuration software that enables the end user to set up the control system without a lot of low-level programming. This allows the user to better focus on the application rather than the equipment, although a lot of system knowledge and skill is still required to support the hardware and software as well as the applications.

engineers tell what they want. Programmers design and code. In milliseconds need sensors and decides what to do.

generally plant when we hear DCS. Controlling controllers: Coordination of different parts.
© 2019, Justin Searle as little input as possible → to safety and security.

DCS EXAMPLE: PETROLEUM REFINERY



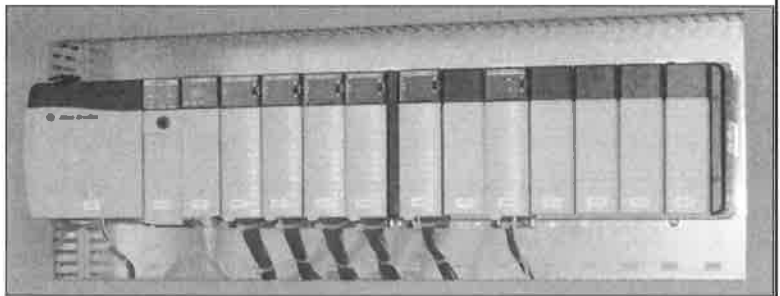
Energy extraction, production, and delivery relies upon various types of ICS to extract, collect, transport, and produce petroleum-based products. An oil refinery, for example, will use large-site DCS and specific field control and SIS to monitor and manage individual units and stages in the process of refining petroleum.

PROGRAMMABLE LOGIC CONTROLLER (PLC)

PLCs were created in an effort to increase flexibility and decrease the proprietary nature of custom DCS plant solutions

General-purpose controller for real-time mechanical processes

- Programmable by the engineer
- Resistant to physical stress
- Expandable connections for various sensors and actuators
- Deterministic logic executed based on state



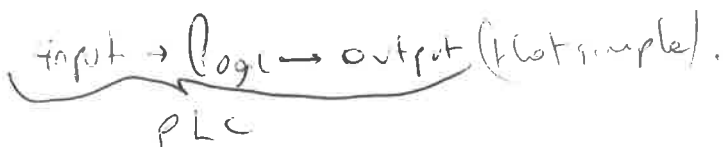
The picture shows an Allen-Bradley PLC. You can see the multiple modules that we'll discuss next.

NIST definition: PLCs are computer-based, solid-state devices that control industrial equipment and processes. Although PLCs are control system components used throughout SCADA and DCSs, they are often the primary components in smaller control system configurations used to provide operational control of discrete processes, such as automobile assembly lines and power plant soot blower controls. PLCs are used extensively in almost all industrial processes.

As we discussed, the logic for running an automated process isn't present on the HMI. That's left up to the PLC to do. PLCs are physically hardened real-time computers. That means a couple of things. First, these components are tested against extreme physical conditions such as very high temperatures, very low temperatures, physical vibrations, electromagnetic interference, or all of the above, plus more. This is one of the reasons the equipment is expensive relative to the processing capabilities. Second, a PLC has to read inputs and respond by adjusting outputs directly. This process has to be real-time or unintended behavior may occur, resulting in a problem or, in the worst case, personal injury. Considering ICS security, an attack that causes a PLC to be unable to respond in real-time can be very effective. We'll go into more detail on this later.

A PLC will take a given set of inputs that it uses to calculate what state to set its outputs in. Modular PLCs have expansion modules to allow for more I/O connections. The processor in the PLC will scan through the set of inputs (sometimes called an "I/O Image Table") and use the programmed logic to set the necessary outputs.

A practical example: In a chemical process, a PLC may be used to automate the task of mixing chemical batches. At one stage in the process, the PLC needs to pump two chemicals into a mixer. Sensors indicating the tank level, temperature, or other sensors are used as inputs. Actuators (in this case, our chemical pumps) are the outputs, which are turned on in order to pump chemicals into the mixing tank until the inputs indicate the tank level is high enough.



BASICS OF CONTROLLER LOGIC

Factors contribute to desired action

- Setpoint (SP) \rightarrow target value
- Process Value (PV)
- Manipulated Variable (MV)
- Error (E = difference between PV and SP)

Example of water faucet with both hot and cold water valves

- 2 setpoints (SP) = the desired temperature and desired flow
- 2 process values (PV) = current water temperature and current flow
- 2 manipulated variables (MV) = valve position for each faucet
- 2 error conditions (E) = temperature difference and flow difference

Control loop theory is used for calculating and controlling an environment or process based on feedback. PID (Proportional, Integral, and Derivative) controller theory is used to optimize tuning. We'll use the classic water faucet example:

An example control loop is used to control the water temperature from a faucet. The hot and cold faucet valves are adjusted and a person touches the water to measure the temperature. Based on the error, he continues to adjust it until the process settles on the desired value.

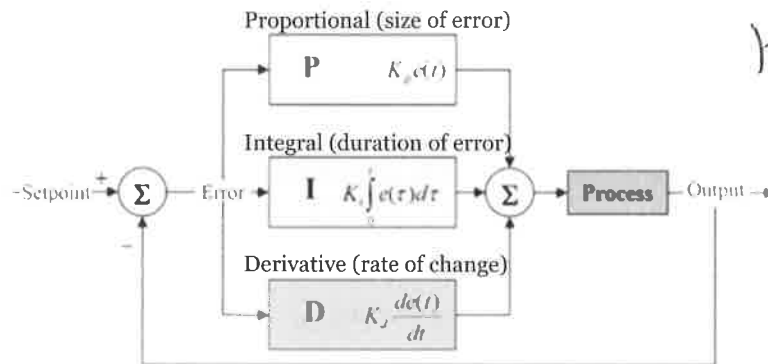
In control loop terms:

- Process Variable (PV): Water temperature
- Manipulated Variable (MV): Valve position for each faucet, two total
- Setpoint (SP): The desired temperature (lukewarm, hot, cold, etc.)
- Error (E): The temperature difference between PV and SP

The process seems simple at first, but a lot of room exists for tuning.

MEASURING ERROR CONDITIONS

PID Controller Theory is used to determine how to respond to the error



*Hota nasil
yarehleh,*

This is a broad topic in itself but we'll breach it here. The three parts to the PID algorithm:

- **Proportional Term:** Proportional to the current error value. This is the part that most reflects change based on error. If this is cranked up too high, there will be large adjustments in response to small errors. The reverse is also true. Think of this as the current error. This contributes most of the output change based on error.
- **Integral Term:** Duration of the error, not just how far off it is. Think of this as the historic error.
- **Derivative Term:** Slope of error over time multiplied by the derivative gain. The point is to slow the rate of change of output and to reduce the overshoot. Think of this as the future error.

Loop tuning is the discovery and experimentation of optimal parameters for a given loop. Manipulation of the three terms and their weights is done to discover the ideal process control.

PID tuning is a non-trivial problem but results in important gains for a process. You don't want your home furnace oscillating constantly between hot and cold. It would be much preferred if the furnace sat in a 1–2 degree range constantly, at least while you are awake and at home. Further tuning exists in time slices: Your house could let itself get colder while you are away at work and warm up when you are expected to return.

Reference:

<http://benrobotics.wordpress.com/>

STARTING TO DEFINE YOUR LOGIC

Fuzzy logic cares about degrees of truth

- Truth value between 0 and 1 for various factors
- Fuzzy values used to determine final logic

Consider the following cold/hot faucet logic:

- **IF** flow is too low ($PV < SP$), **THEN** increase hot and cold water valves
- **IF** flow is too high ($PV > SP$), **THEN** decrease hot and cold water valves
- **IF** temperature is too low ($PV < SP$), **THEN**
 - Increase hot water valve **AND**
 - Decrease cold water valve
- **IF** temperature is too high ($PV > SP$), **THEN**
 - Decrease hot water valve **AND**
 - Increase cold water valve

Fuzzy logic is a term for taking logic statements and distilling them down in order to control machinery. Fuzzy logic can also use linguistic variables for configuring a system.

The logic is easy for a person to read and identify what the system is supposed to do. These logic statements are translated into fuzzy logic for mathematical evaluation.

In doing so, values may have a degree of truth that may be between 0 and 1, rather than simply true (1) or false (0). For instance, referring to the temperature of a room, the assessment of the statement, "The room is too hot," might vary. If the expected temperature is 70°F and the current temperature is 71°F, that's a small degree of truth for the statement, as opposed to the room temperature being 92°F, which is *very* true.

PROGRAMMING FINAL LOGIC

Special purpose computers running real-time operating system

Rely on firmware and system applications

IEC61131-3 defines five programming approaches for controllers:

- Ladder Logic
- Function Block
- Structured Text
- Instruction List
- Sequential Function Chart



The configuration of field devices implemented in the PLC can use several programming approaches and languages. The IEC61131-3 publication covers programming languages and has contributed to the following becoming the most common:

- **Ladder Diagram (also referred to as *Ladder Logic*) (LD):** This approach was developed off of relay logic displays and drawings and was written to be understood by technicians. There are subtle implications related to how the processor scans for reading and programming ladder logic. It is probably the most widespread language, but limitations dealing with more complex handling have resulted in the adoption of other languages.
- **Function Block Diagram (FBD):** This approach uses a series of blocks with a purpose and inputs and outputs strung together. The order by which the blocks execute depend on the vendor's implementation. This programming language had experienced significant growth in implementations and is probably the next behind LD.
- **Structured Text (ST):** A series of instructions to include statements implemented by a vendor's use of an editor (can be similar to high-level computer programming languages such as C). This approach can be difficult to troubleshoot. This language is seeing much greater increases in use.
- **Instruction List (IL):** Similar to assembly language, but not common in certain geographies. This is a very low-level language and it executes very quickly on a PLC and takes much less space in memory (it is very popular in Europe).
- **Sequential Function Chart (SFC):** This automation approach uses steps and transitions from top to bottom. Steps are used to execute commands, and transitions wait for conditions to be true to move forward.

Some hardware limits which language you can implement (this is typically true for some microcontrollers). Regardless of the programming language, the field components (PLCs, controllers, RTUs, loop controllers, field controllers, etc.) organize the inputs and outputs and pass information or perform control functions based on "rules." The configuration for these controllers is often kept in a database within the engineering workstation. Regardless of the programming approach, cyber impacts to field components can include disrupting communications to the device, consuming processes and resources, changing the configuration for accessing the device, rebooting the device, modifying the programming/setpoints, changing group settings, manipulating the firmware (low-level machine code), modifying applications on the device, etc.

LADDER LOGIC EXAMPLE

Based on traditional relay logic diagrams

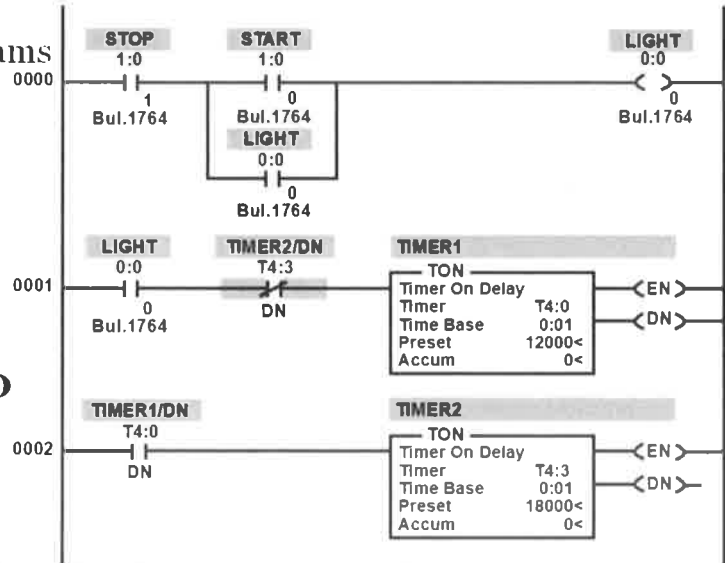
Most common programming style in most existing ICSs

Scans each rung at a known scan rate

- Conditions checked on the left
- Changes are indicated on right

Example 0000 rung:

- **IF STOP** and **START** are **CLOSED**
- **OR IF STOP** and **LIGHT** are **CLOSED**
- **THEN** turn **LIGHT ON**



Ladder Logic is a type of programming language used in many PLCs. It's based on the circuit diagrams of relay logic hardware previously used in these types of systems. It had traditionally been the most common language.

Ladder Logic (so named because of its appearance) is a programming language operating on rules rather than procedures or functions. The loop of rules is passed over sequentially and continuously. Each rule is executed as a logical test to determine the firing of an actuator (for example, activating on a pump, closing an electrical relay, or opening a pressure release valve). The logic is read top down from left to right on a "rung."

The ladder is passed over many times per second (called a scan). The scan time of the system has to be low enough to respond to input changes and affect the process correctly. If the scan time is too long or unstable, the behavior of the system is at risk.

Reference:

<http://www.famictech.com/pro/plc-ladder-logic.html>

FIELD DEVICES

Controllers are wired to various instrumented devices

- **Sensors:** Temperature, humidity, vibration, sound, pressure, etc.
- **Actuators:** Valves, solenoids, pumps, agitators, burners, compressors, etc.

Communication to these devices is referred to as I/O (input/output)

- **Basic I/O:** Single digital or analog lines
- **Smart I/O:** Fieldbus protocols over specialized networks

Lines can blur between controllers and field devices

- Smart sensors and actuators have built-in logic
- They may have their own basic field device below them



Instrumented devices are all of the devices that are ultimately responsible for monitoring, measuring, or controlling the bits and pieces of an industrial system or process.

Sensors can be important pieces of a control system. Many ICS configurations will either directly control these types of environmental qualities or otherwise depend on them being at a certain level.

An HVAC, for example, exists to control the temperature, humidity, and other factors for a given spatial area. Having a sensor network collecting data and making it available to the SCADA/RTU or PLC is needed to complete that task.

Other types of ICS may depend on an environmental factor being at a certain level. For example, a chemical process may require a room temperature between 60°F and 70°F. The HVAC (a separate control system) is responsible for controlling that temperature, but the chemical process also requires sensors to identify if the proper conditions for execution are met.

Among the actuators, these devices could be valves, switches, level monitors, pumps, compressors, levers, pressure sensors, thermocouples, etc. Communication with a hardware device is generally referred to as I/O. The I/O could be digital or analog, and its physical connectivity could be one of many options—we'll discuss this in a few minutes.

BASIC I/O

Digital (discrete) I/O

- Voltage present or not present on a wire
- Sensors: Switches, object positioning, state, etc...
- Actuators: Alarms, stepper motors, relays, pumps, lights, etc...
- Represented in logic as single binary bit (0/1) or Boolean values (True/False)



Analog I/O

- Value is a range of voltage (0–5 V, 0–24 V) or current (4–20 mA) of a signal
- Sensors: dials, temperature, pressure, flow, speed, etc...
- Actuators: valves, variable speed motors, mixers, burners, etc...
- Represented in logic as 8/16/32/64 bit integers (410) or floating points (4.10)



Digital I/O is an input or output where the specified value is communicated as simple on-or-off signals. Common examples of digital I/O include relays, switches, and status reporting. *Typically a bit*

Analog I/O is an input or output where the specified value is communicated by varying the voltage or current of a signal. Common examples of analog I/O include temperature, pressure, flow, or speed measurements.




SMART I/O WITH FIELDBUS PROTOCOLS

Sometimes our controllers are connect to other controllers, or to smart sensors and actuators

- Smart I/O is often performed through protocols classified as fieldbus
- Fieldbus protocols like HART can use traditional 4-20mA analog lines
- Most fieldbus protocols use standard or proprietary serial buses

Controller can talk to each other over a serial line

The RS standards are very common, now maintained by TIA

- TIA-232: Point-to-point serial 
- TIA-422: Multi-drop, single-master bus 
- TIA-485: Multi-drop, multi-master bus 

over a bus

Modern fieldbus protocols can also work over Ethernet

Fieldbus protocols will be discussed in more detail later in the course

References:

HART Protocol – https://en.wikipedia.org/wiki/Highway_Addressable_Remote_Transducer_Protocol
TIA – https://en.wikipedia.org/wiki/Telecommunications_Industry_Association
TIA-232 – <https://en.wikipedia.org/wiki/RS-232>
TIA-422 – <https://en.wikipedia.org/wiki/RS-422>
TIA-485 – <https://en.wikipedia.org/wiki/RS-485>

writes the electrical wave, control and find here it a sync. error. single purpose device.

SMART FIELD DEVICES

Smart field devices can also be categorized as

- Intelligent Electronic Devices (IED)
- Industrial Internet of Things (IIoT)

Examples

- Digital protective relay (DPR) → if a point is reached give a break. look for this do this...
- Phasor measurement unit (PMU)
- Smart meters

Smart field devices compared to controllers

- More limited purpose and function
- Usually microcontroller-based
- Contains its own control logic that cannot be changed



SANS

Smart field devices (also known as IEDs or IIoTs in some industries) are special field devices that are primarily used in the electric power industry. They usually are limited in their purpose or function, unlike the general-purpose PLCs and RTUs. Smart field devices contain their own control logic and are usually microcontroller-based. Examples from the electric power industry include digital protective relays (DRPs) and phasor measurement units (PMUs).

A digital protective relay (DPR) is a device containing a microcontroller with the specific purpose of measuring voltages and currents to determine whether a fault in the system exists. A DPR is an example of an IED, which is any end device that has that kind of capability. RTUs may be configured to communicate with end devices to collect status information for reporting back to the supervisory system.

Synchrophasors measure voltages and currents at principal intersecting locations (critical substations) on a power grid and can output accurately timestamped voltage and current phasors. Because these phasors are truly synchronized, a synchronized comparison of two quantities is possible in real-time. These comparisons can be used to assess system conditions, such as frequency changes, MWs, MVARs, kVolts, etc. The monitored points are preselected through various studies to make extremely accurate phase angle measurements to indicate shifts in the system (grid) stability. In North America, the phasor data is collected either on-site or at centralized locations using Phasor Data Concentrator technologies. The data is then transmitted to a regional monitoring system that is maintained by the local independent system operator (ISO). These ISOs will monitor phasor data from individual PMUs or from as many as 150 PMUs—this monitoring provides an accurate means of establishing controls for power flow from multiple energy generation sources (nuclear, coal, wind, etc.).

Industrial Internet of Things (IIoT) is more of a marketing term than a concrete category of devices. But some common features usually (but not always) seen in products labeled IIoT are a large number of simple devices, usually relatively inexpensive, often using wireless communications, and probably controlled centrally from a management server or cloud solution. An example of IIoT devices in the Electricity sector are Smart Meters.

TAKEAWAYS AND RECOMMENDATIONS

Section takeaways

- There are many types of controllers, not always clearly defined
 - Some controllers such as DCSs are custom designed for a plant
 - Other controllers such as PLCs, can be programmed and reprogrammed with different process logic by the engineer
 - Smart field devices can also be considered controllers, but not reprogrammable
 - IIoT and IED are types of smart field devices → not programmable
- Controllers talk to field devices via basic I/O or fieldbus protocols
 - Sensors allow controllers to measure elements of the process
 - Actuators provide a means for controllers to change the process
 - When controllers receive input and make decisions is often critical

— serial
ethernet

This page intentionally left blank.

Course Roadmap

Day 1: ICS Overview

Day 2: Field Devices and Controllers

Day 3: Supervisory Systems

Day 4: Workstations and Servers

Day 5: ICS Security Governance

1. Introduction
2. GICSP Overview
3. ICS Concepts
 - Processes and Roles
 - Industries
 - **Exercise 1.1: Learning from Peers**
4. Purdue Levels 0 and 1
 - Controllers and Field Devices
 - Programming Controllers
 - **Exercise 1.2: Programming a PLC**
5. Purdue Levels 2 and 3
 - HMIs, Historians, Alarm Servers
 - Special Applications and Master Servers
 - Control Rooms and Plants
 - SCADA
 - **Exercise 1.3: Programming an HMI**
6. IT and ICS Differences
 - ICS Life Cycle Challenges
7. Physical and Cybersecurity
8. Secure ICS Network Architectures
 - ICS410 Reference Model
 - Design Example
 - **Exercise 1.4: Architecting a Secure DCS**

This page intentionally left blank.

EXERCISE 1.2: PROGRAMMING A PLC

DURATION TIME: 60 MINUTES

We are going to program our Velocio PLC to control a chemical mixing process.

- 3 chemicals need to be mixed together in a specific order and at specific times
- 2 digital inputs: **RunProcess** switch and **EmergencyStop** switch
- 6 digital outputs: **3 x IntakePumps**, a **Mixer**, an **OuttakePump**, and an **EmergencyFlush** pump

OBJECTIVES

- To gain a better understanding of how controller logic works and interacts with the process.
- Understand what can happen if an attacker can influence the controller.
- Become familiar with PLC programming software and how attackers who access it can do anything they want with the process.

PREPARATION

- Start the Windows 10 virtual machine.
- Prepare your Velocio PLC for use.
- Read note below about a possible graphical issue you might see during the exercise

SANS

ICS410 | ICS/SCADA Security Essentials

vBuilder doesn't always work well with resolution scaling (the 130% or 150% settings) of high-resolution displays. This is a common issue with a lot of Windows software, so you there are good chances you already have resolution scaling turned off. For this exercise, if you experience this issue, it will only affect one step, when you set "Pause" on the Timer block. To complete the lab, you have two choices.

- 1) Ignore the graphical anomaly by selecting the start option and then pressing the down arrow once to select the invisible "Pause" option or twice for the invisible "Stop" option. I personally recommend this approach
- 2) Remove resolution scaling, which will make all your icons and text really, really small. You can do this by setting resolution scaling in Windows to 100% if enabled. If you are running Windows on your host (your laptop), change it there as your host will pass that setting along to the virtual machine. You can usually make it larger (and readable) again by changing the actual display resolution on your host to something like "1440 x 900" based on your preference.

If you are running Mac or Linux, you shouldn't experience this issue, but on occasion, this may be an issue with some Retina displays on MacBook Pros as well. If you do have this issue, the solution is for you to change your VMware Fusion **Display** settings for your Windows 10 VM to "Use full resolution for Retina display" and the single window and full screen settings to "Use Fusion Display Preferences".

EXERCISE 1.2: PROGRAMMING A PLC

PROCESS REQUIREMENTS

Process Requirements:

- One part chemical 1 and 2 to two parts chemical 3
 - Chemical 1 and 2 intake pumps must run 2 seconds
 - Chemical 3 intake pump must run 4 seconds
- Chemical 1 must be added before the others
- Chemical 2 and 3 must be added while mixer is running
- After all chemicals fully added, they must be mixed for 4 seconds
- Once mixed, outtake pump must run 5 seconds to process batch
- **RunProcess** switch can start and pause the process
- **EmergencyStop** switch must run emergency flush pump for 5 seconds before resetting system

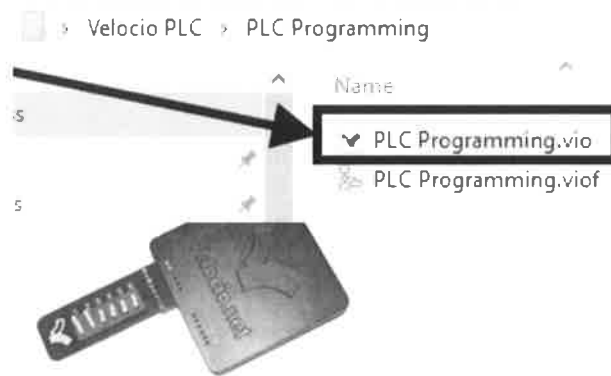


In this exercise, we will program our Velocio PLC to control a simple chemical mixer program. You can see what the process requires in the above slide. This exercise will step you through the process of building this program and programming your PLC with it. After all, with only two inputs, a start/pause switch and an emergency stop switch, how hard can it be?

EXERCISE 1.2: PROGRAMMING A PLC

CONNECTING TO YOUR PLC

1. Open the "Velocio PLC" folder on the desktop of your Windows 10 VM
2. Open the "PLC Programming" folder
3. Open the "PLC Programming" file that has the Velocio logo
4. Answer "Yes" to the prompt about vBuilder modifying your computer
5. Turn off all the switches on the input switchboard (away from the numbers)
6. Connect your input switchboard on the **LEFT SIDE** of your PLC
7. Use the USB cable to connect your PLC to the computer



To do this exercise, we have already created a template as a starting point. To access this template, follow these steps:

1. Open the "Velocio PLC" folder on the desktop of your Windows 10 VM.
2. Open the "PLC Programming" folder.
3. Open the "PLC Programming" file that has the Velocio logo.
4. Answer "Yes" to the prompt about vBuilder modifying your computer.

Velocio needs to be able to install drivers to access your PLC and program your PLC, so it needs a bit more permission than most programs; thus, you will see this prompt every time you start Velocio. Now we need to prepare our PLC to connect to our computers.

5. Turn off all the switches on the input switchboard (away from the numbers).
6. Connect your input switchboard on the **LEFT SIDE** of your PLC.
7. Use the USB cable to connect your PLC to the computer.

Once you have it connected to your computer, there is no visual indicator in Velocio to show it is connected. We will just assume it is connected until we try programming it at the end of the lab. However, please make sure inside of VMware that your PLC is connected to your Windows 10 virtual machine, not your host machine. You can do this by opening your VM settings, going to the USB configurations, and making sure the **Luminary Micro** device is connected. Not having your PLC connected to the correct VM is the most likely reason that the programming step at the end of this exercise fails.

Connect USB devices:

Name	Plug in Action
LG USA USB Audio	Ask what to do
LG USA USB Controls	Ask what to do
 Luminary Micro Virtual COM Port	Ask what to do

EXERCISE 1.2: PROGRAMMING A PLC

BUILDING A STATE MODEL

The PLC Programming file you just opened is a template for you to fill in

- Template is a basic state model, checking for each state
- There is a single timer that is used to determine which state we are in
- The whole batch process will take 15 seconds to complete, then process starts over
- You will complete the logic for each state in this exercise

The easiest method for programming a PLC is to build a state model. Here is our model:

- State 0: Initialization of tags (outputs and variables)
- State 1: Adding the first chemical to the mixer (finished at 2 seconds into batch timer)
- State 2: Adding the second and third chemicals (finished at 4 seconds into batch timer)
- State 3: Finish adding the third chemical (finished at 6 seconds into batch timer)
- State 4: Mix the batch (finished at 10 seconds into batch timer)
- State 5: Process the batch to the next system (finished at 15 seconds into batch timer)

Now you have to do the rest... with the help of our walkthrough of course...

Once you get the PLC connected, look at the template we provided you. There is a start icon that has connections with arrows showing you the flow of the program. This is a sequential function block program. The logic in this program follows a state machine design. At any given point, our program will be assigned to one of the six states above numbered zero through five. Each state has a specific action that is performed. The PLC will continually loop through the program for each state until some requirement for that state is completed. When the PLC reaches the end of a program branch (each state will become a branch), the PLC will automatically loop back to the Start object.

All of the objects we have in our template are either testing which state we are in or performing special actions for the Start and Stop switches. We only have the basic decision blocks created for you to help keep your program organized. The rest of this exercise will help you program each state.

EXERCISE 1.2: PROGRAMMING A PLC

VIEWING PRE-CONFIGURED TAGS



Tags are human readable names for inputs, outputs, and other needed variables

We preconfigured the tag names for you

- **Please do not change these**
- Any changes may break this exercise
- This step is just to give you context

View all the I/Os and Register categories on the left

Input / Output	Name	Signal	Used	Modbus
Input bit	RunProcess	C6	<input checked="" type="checkbox"/>	<input type="checkbox"/>
Input I16	InBrC2	C2	<input type="checkbox"/>	<input type="checkbox"/>
Input Float	InBrC3	C3	<input type="checkbox"/>	<input type="checkbox"/>
Output bit	InBrC4	C4	<input type="checkbox"/>	<input type="checkbox"/>
Output u16	InBrC5	C5	<input type="checkbox"/>	<input type="checkbox"/>
Register bit	RunProcess	C1	<input checked="" type="checkbox"/>	<input type="checkbox"/>
ui8				
i16				
ui16				
i32				
Float				

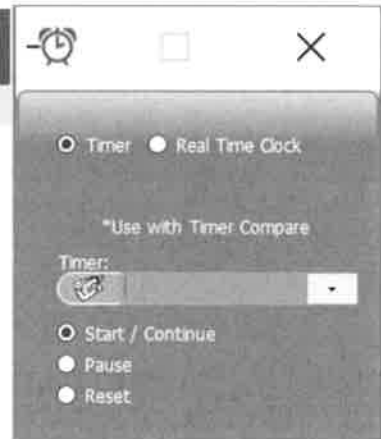
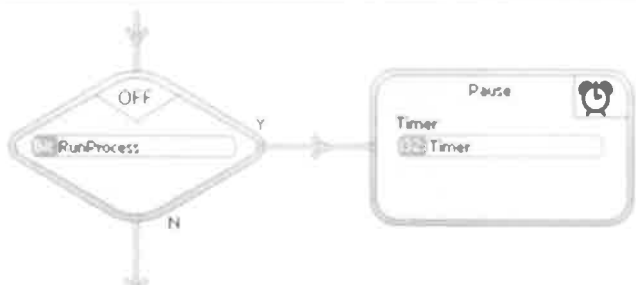
One key component we need in programming our PLC is a way to read our inputs and write to our outputs. We also need a variable to keep track of internal logic, such as the state our program is in and the current value of our process timer, neither of which are inputs nor outputs. We can assign names to all of our inputs and outputs and create other needed variables for our programs with the Tag tool. In ICS programming logic, tags are human-readable names for inputs, outputs, and variables in our programs. These tag values make up most of the time-series data that we store in our historians.

Take a moment to look at the tags we have already set for you. Use the tag tool icon indicated above to open the tag editing window. On the left side of the tag window are various buttons representing different types of tags. Review the tags we set under the following sections: Input Bit, Output Bit, ui16, and i32. The first two categories apply to our inputs and outputs on the PLC. For the other two categories, the "u" stands for unsigned and the "i" stands for integer. We need an unsigned integer to keep track of our **State** (we chose a 16-bit but could have chosen 8-bit as well) and a 32-bit integer for our **Timer**.

Please be careful not to edit these values or it could impact our exercise. Close the tag window once you are finished inspecting these values.

EXERCISE 1.2: PROGRAMMING A PLC

RUNPROCESS SWITCH



8. Add a **Timer / Clock** object with the tag **Timer** set to **Pause** so if the switch is off, the timer (and thus, the state) will freeze in place until the switch is turned on. If you do not see the Pause and Reset options as shown above, refer to the intro slide for this exercise
9. Connect the provided **RunProcess = OFF** decision block to the new block you just added, as seen in the diagram above

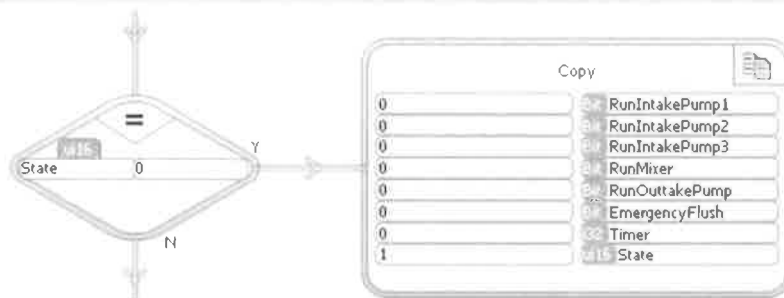
To **Start** our process, we will use input switch number one. We have already assigned this the human-readable name of **RunProcess** as you should have seen in the previous step of this exercise.

Our process requirements say this switch must be able to start and stop/pause the process. So, for our logic, we want to detect if the switch is off. If it is off, then we should pause the process timer. This will make the PLC continuously loop through this branch and continuously pause the process timer. In reality, the timer just stays paused, but often, in control logic, the program continuously takes the same path until something tells it to take a different path. For us, we want it to pause **Timer**, and then because it is the end of a branch, it will go back to **Start**, continue until we see the **RunProcess** switch still off, go to the pause **Timer**, go to **Start**, detect the **RunProcess** switch is off, pause **Timer**, go to **Start**, etc. Again and again, the PLC will do this, approximately more than 1,000 times per second—at least on this PLC. Once we move the **RunProcess** switch, the route will change to continue down the functional logic program to run the appropriate state logic.

Find the diamond-shaped decision block in the template program logic that looks like the diamond-shaped block above. This should be the second decision block after the **Start** object. We will skip the first decision for **EmergencyStop** for now. Next to the **RunProcess** decision block, you will need to add the appropriate rectangle block to the right of it. To do this, perform the two actions indicated in the slide above.

EXERCISE 1.2: PROGRAMMING A PLC

STATE 0 – INITIALIZATION



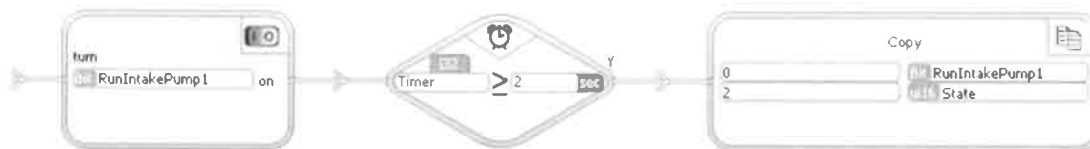
10. On the **State = 0**. branch, add a **Copy** object to initialize each tag
 - Copy a **zero** (just type a **0** in the field) to the **six output tags** and **Timer**
 - Copy a **1** to the **State** variable (to move to the next state)
11. Build a connector from the provided **State = 0** decision block to the new copy block you just added

When the program **Starts** the process for the first time, or when we finish one batch and need to start the process again, we start at **State 0**. **State 0**'s goal is simple: Reset and initialize all tags to begin the process. To do this, we set all tag values to **0**. While there is a **Turn On/OFF** object to set tags on and off, that object allows you to change only one tag per block. So, we will instead use a **Copy** object to "copy" the needed values to each tag. The only tag we set differently is the **State** tag. Because we need to run this initialize branch only once per batch process, we'll set **State** to **1**, which will make the program go to the next state of the program when it loops back to **Start**.

Perform the actions in the slide above to complete the logic of this state. Note that the screenshot above shows the final object once it is saved. The edit window while you are configuring the object looks slightly different, but the values on the left and right side of the edit window are the same as the copy object shown above.

EXERCISE 1.2: PROGRAMMING A PLC

STATE 1 – ADD CHEMICAL 1



12. On the **State = 1** branch, add a **Turn On/Off** object to activate **RunIntakePump1**
13. Add a **>=** object to test if the **Timer** is greater than **2 seconds**
14. Add a **Copy** object to turn off the **RunIntakePump1** (copy a **zero** to it) and increment **State** to the next state of **2**
15. Connect the provided **State = 1** decision block to the three new blocks you just added, as seen in the diagram above, selecting **YES** when connecting the last two blocks

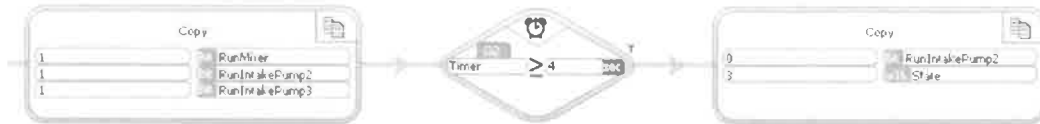
Our process requirements say that chemical 1 must be added before the other two chemicals, so we will make this **State 1**. That means we need to add the chemical using intake pump 1, wait 2 seconds while it is added, then stop adding chemical 1 before moving on to the next **State**. To do this, we will use an **ON** action block to turn on **RunIntakePump1**, use a **Timer** to wait 2 seconds, then use a copy block to turn off **RunIntakePump1** and increment the **State**.

Perform the actions in the slide above to complete the logic of this state. This string of blocks should be added to the right of the **State = 1** block in your template program.

Now look closely at your logic. Notice that the timer is diamond-shaped. This means it is a decision block. When your PLC reaches this block, it will test if the **Timer** variable has reached 2 seconds yet. Once that time is met, it will go out the right side as indicated by the **Y** next to the path. If it has not reached 2 seconds yet, then it will take the **N** or No path, which is not defined in our flow diagram. This means the PLC has reached the end of this programming branch and goes back to the **Start** object. It is important to know that the PLC does not stay in this block until it reaches 2 seconds, otherwise it would have no way of checking if an emergency stop has occurred during those 2 seconds.

EXERCISE 1.2: PROGRAMMING A PLC

STATE 2 – ADD CHEMICAL 2 AND 3



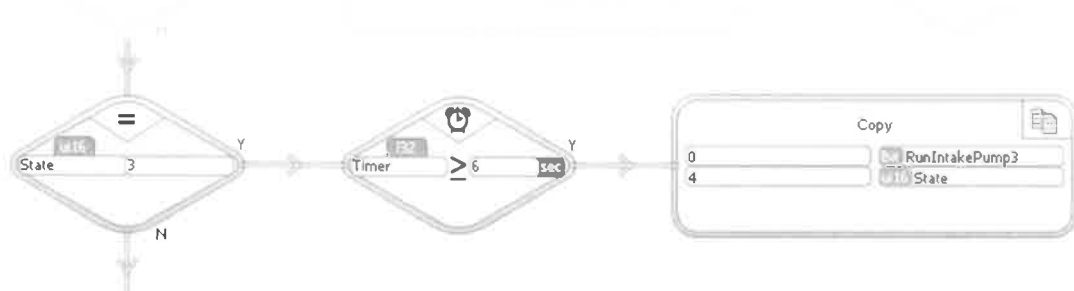
16. On the **State = 2** branch, add a **Copy** object to activate **RunMixer**, **RunIntakePump2**, and **RunIntakePump3** (set them all to **1**)
17. Add a **>** object to test if the **Timer** is greater than **4 seconds**
18. Add a **Copy** object to turn off the **RunIntakePump2** (copy a **zero** to it) and increment **State** to the next state of **3**
19. Connect the provided **State = 2** decision block to the three new blocks you just added, as seen in the diagram above, selecting **YES** when connecting the last two blocks

Our process requirements say that chemicals 2 and 3 can be added at the same time; however, chemical 3 must be added for twice as long as chemical 2. They also require that the mixer run while we add these chemicals. So, our logic will first use a copy block to turn on **RunMixer**, **RunIntakePump2**, and **RunIntakePump3**. Once we turn those three outputs on, our program will continually loop until the **Timer** reaches 4 seconds, at which point we will turn off **RunIntakePump2** and increment **State**. This will allow **RunIntakePump3** and **RunMixer** to keep running into the next State.

Perform the actions in the slide above to complete the logic of this state.

EXERCISE 1.2: PROGRAMMING A PLC

STATE 3 – FINISH ADDING CHEMICAL 3



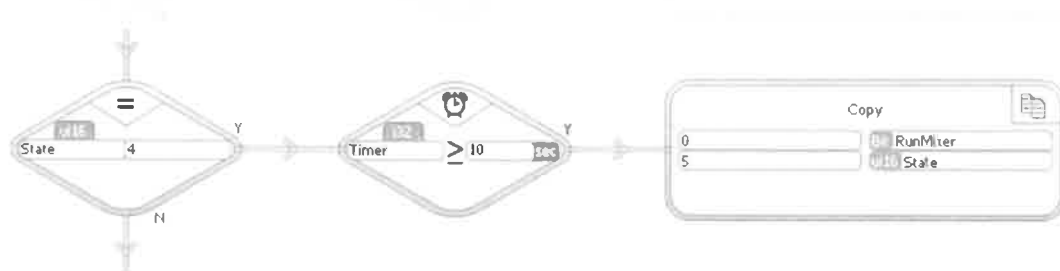
20. On the **State = 3** branch, add a \geq object to test if the **Timer** is greater than **6 seconds**
21. Add a **Copy** object to turn off the **RunIntakePump3** (copy a **zero** to it) and increment **State** to the next state of **4**
22. Connect the provided **State = 3** decision block to the two new blocks you just added, as seen in the diagram above, selecting **YES** when connecting the last two blocks



Our process requirements say that chemical 3 needs to be added for 4 seconds. The previous state had it running for 2 seconds, which means we need to run it for 2 more seconds. Because **RunIntakePump3** is already on, we just need to check the **Timer** to see if it has reached 6 seconds. Once it has, we need to turn off **RunIntakePump3** and increment state. We will leave **RunMixer** on for the next state.

Perform the actions in the slide above to complete the logic of this state.

EXERCISE 1.2: PROGRAMMING A PLC

STATE 4 - MIX BATCH



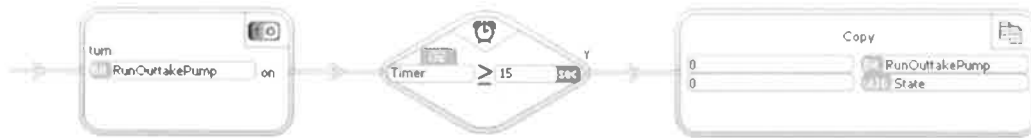
23. On the **State = 4** branch, add a  object to test if the **Timer** is greater than **10 seconds**
24. Add a  object to turn off the **RunMixer** (copy a **zero** to it) and increment **State** to the next state of **5**
25. Connect the provided **State = 4** decision block to the two new blocks you just added, as seen in the diagram above, selecting **YES** when connecting the last two blocks

Our process requirements say that after all the chemicals are added, we need to mix them for 4 seconds. Because **RunMixer** is already running, we just need to check to see if the **Timer** has reached 10 seconds. Once it has, we need to turn off **RunMixer** and increment **State**. This should leave us with all outputs off going into the next state.

Perform the actions in the slide above to complete the logic of this state.

EXERCISE 1.2: PROGRAMMING A PLC

STATE 5 – PROCESS BATCH



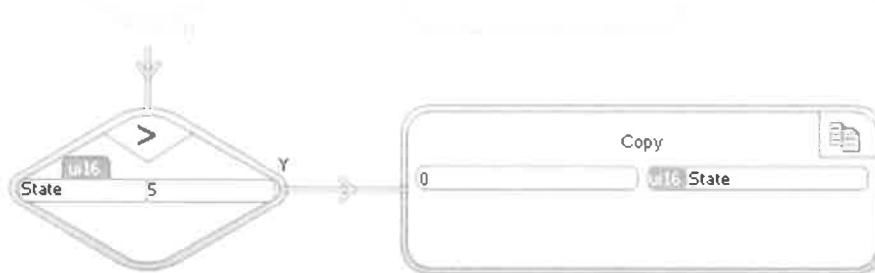
26. On the **State = 5** branch, add a **Turn On/Off** object to activate **RunOuttakePump**
27. Add a **> ≥** object to test if the **Timer** is greater than **15 seconds**
28. Add a **Copy** object to turn off the **RunOuttakePump** (copy a **zero** to it) and reset **State** to **0** to restart the process
29. Connect the provided **State = 5** decision block to the three new blocks you just added, as seen in the diagram above, selecting **YES** when connecting the last two blocks

To finish our batch, our process requirements say we need to run the outtake pump for 5 seconds to move the mixed chemical onto the next process. To do this, we need to turn on **RunOuttakePump** and wait until the Timer reaches 15 seconds, at which point we can turn off **RunOuttakePump** and reset **State** back to 0.

Perform the actions in the slide above to complete the logic of this state.

EXERCISE 1.2: PROGRAMMING A PLC

PREVENTING UNDEFINED STATES



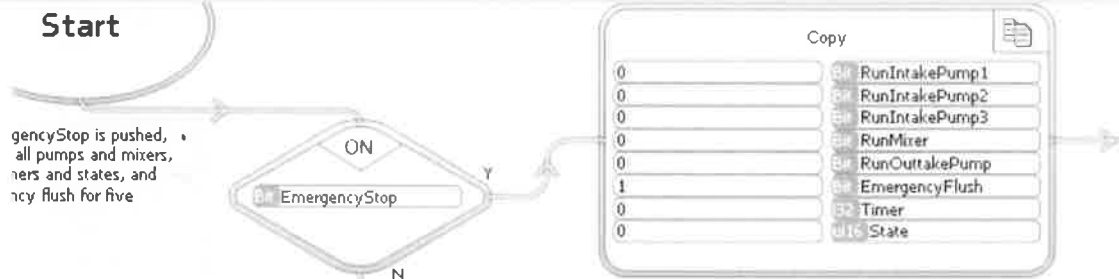
30. On the **State > 5** branch, add a **Copy** object to reset **State** to **0** just in case we ever have an undefined state, which would be anything greater than the 5 states we have just defined
31. Connect the provided **State > 5** decision block to the new block you just added, as seen in the diagram above

Now our programming logic should never change the State variable to anything greater than 5; however, just in case, we should build a safety net forcing **State** to reset to 0 if it ever occurs.

Perform the actions in the slide above to complete the logic of undefined states.

EXERCISE 1.2: PROGRAMMING A PLC

DEALING WITH EMERGENCY STOPS (PART 1 OF 2)



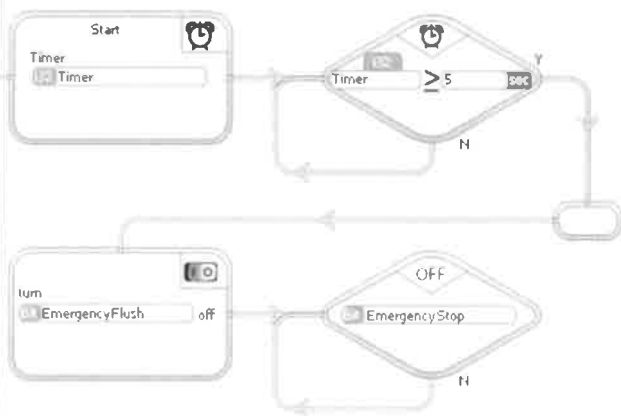
32. Add a **Copy** object to initialize each tag (example above)
 - Copy a **zero** to the first **five output tags**
 - Copy a **1** to **EmergencyFlush** to activate the emergency flush pump
 - Copy a **zero** to **Timer** so we can reuse it for the flush process
 - Copy a **zero** to **State** to reset the process once this routine is finished
33. Connect the provided **EmergencyStop = ON** decision block to the new copy block you just added, as seen in the diagram above

Now that all the state logic is completed, we have to deal with emergency events. The process requirements say that if a user triggers the emergency stop switch, the process should immediately stop its current action and run the emergency flush pump for 5 seconds to discard the batch. This means the first thing we need to do is turn off all outputs, reset all variables, and turn on the **EmergencyFlush** output.

Perform the actions in the slide above to start the logic of the Emergency Stop action.

EXERCISE 1.2: PROGRAMMING A PLC

DEALING WITH EMERGENCY STOPS (PART 2 OF 2)



34. After the last **Copy** block, add a **Timer / Clock** object with tag **Timer** set to **Start**
35. Add a **>=** object to test if **Timer** is greater than **5 seconds**
36. Add a **Wire Router** object to help separate our connectors
37. Add a **Turn On/Off** object to turn off **EmergencyFlush**
38. Add a **OFF?** object to loop if **EmergencyStop** is on
39. Connect the **Copy** block to the new blocks as in the diagram, making sure the **Y** and **N** paths match those in the diagram

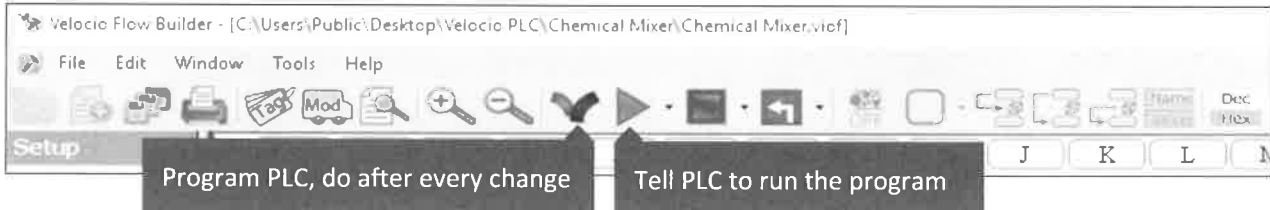
5/15

We are going to continue to the right of the copy block we just created to finish the emergency stop process. Now that we have the **EmergencyPump** running in the copy block, we need to start **Timer** and wait until it reaches 5 seconds. Previously, when we checked timers and they hadn't reached the specified values, we just let the branch end with **N** and then loop back through the process continually until it is completed. We do not want this behavior for our **EmergencyStop** action. We want the PLC to stay in this branch until it is finished and everything is reset. So, we are going to loop the **N** option back to the timer check input so it stays in this object until the **Timer** reaches 5 seconds. Once it does, we will turn off **EmergencyPump** and check to see whether the **EmergencyStop** switch is still on. We want to continue to loop through this object until the user disables the **EmergencyStop** switch, at which point we can start a new batch process.

Perform the actions in the slide above to finish the logic of the Emergency Stop action.

EXERCISE 1.2: PROGRAMMING A PLC

RUNNING THE PROGRAM



40. Program the PLC by clicking on the Velocio icon
41. Run the program by clicking the Play button
42. Flip switch 1 to on, and watch the output LEDs on the PLC which should look like this:
Seconds 0–2: Output LED 1 ● ○ ○ ○ ○ ○
Seconds 2–4: Output LED 2, 3, and 4 ○ ● ● ● ○ ○
Seconds 4–6: Output LED 3 and 4 ○ ○ ● ● ○ ○
Seconds 6–10: Output LED 4 ○ ○ ○ ● ○ ○
Seconds 10–15: Output LED 5 ○ ○ ○ ○ ● ○

SLWS

ICS410 | ICS/SCADA Security Essentials

60

Now that we have our program written, we need to program our PLC and see how it runs. To do this, follow the steps above.

When you run the program, and turn on the **RunProcess** on your PLC input board (switch 1), you should see the output lights start cycling through different patterns. Here are the patterns you should see:

- Seconds 0 – 2: Output LED 1 (the one closest to the input board)
- Seconds 2 – 4: Output LED 2, 3, and 4
- Seconds 4 – 6: Output LED 3 and 4
- Seconds 6 – 10: Output LED 4
- Seconds 10 – 15: Output LED 5

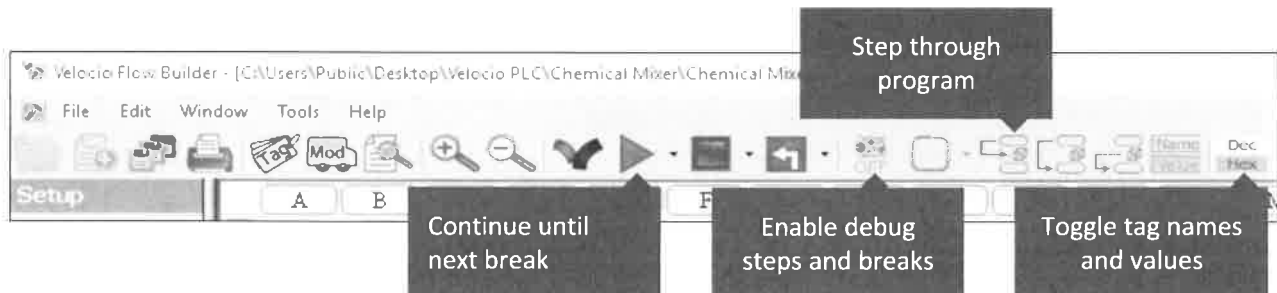
Turning off **RunProcess** in the middle of the batch process should pause all output LEDs until you turn on **RunProcess** again, which should then continue where it left off.

When you flip the **EmergencyStop** switch (switch 6), you should see all output LEDs turn off and LED 6 turn on for 5 seconds, and then all output LEDs should turn off until you disable the **EmergencyStop** switch.

If the process does not seem to work properly, go back and ensure your functional diagram looks like the functional diagram screenshots in this exercise. Alternatively, you can run the debug tool to step through the process as described in the next slide.

EXERCISE 1.2: PROGRAMMING A PLC

DEBUGGING THE PROGRAM



43. If things don't seem to be working right, try enabling debug and stepping through the program
44. If you cannot get your program working correctly, there is a completed project you can use in the Chemical Mixer folder

About at this point in the exercise, your instructor should do a demonstration of how to use the debug functions. If he hasn't done so yet, let him know you are at this point and need the demonstration. If you are taking the course via OnDemand, there should be a video showing this demonstration.

If you do this, note that your Velocio PLC processes each block in your program at sub-millisecond rates, so if your timer is waiting for 2 seconds as it does in State 1, you will have to step through the blocks in the branch well over 2,000 times unless you use breakpoints. To fix this, once debug is running, click on the final block of each state to set breakpoints. Use the Step-In tool to work through each block until you get into a loop, then use the Play button to play through that loop until it reaches the next breakpoint. Your instructor will demonstrate this.

EXERCISE 1.2: PROGRAMMING A PLC

QUESTIONS FOR THOUGHT

When we first start the PLC, what assumptions do we make about the initial **State**?

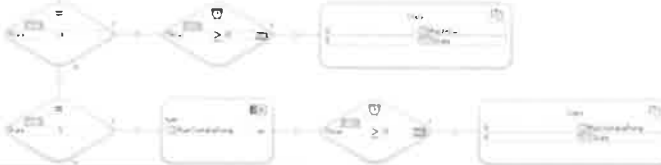
Notice we pass through the **Start Timer** object thousands of times per second on every loop through the logic instead of once per batch.

- Why does this not restart the **Timer** each loop?
- What is the benefit to the way we did it here?



Some of our state branches set outputs before the **Timer** (states 1, 2, and 5) while other states (3 and 4) assume outputs are on and go straight to a **Timer**.

- Can an attacker take advantage of the assumptions we are making on states 3 and 4?



When we first start the PLC, what assumptions are we making about State?

Notice that we initialize all the outputs and variables during State 0, but how did we get to State 0? We assume that when the PLC starts that State = 0. If somehow it starts as State 5, then we might burn up our outtake pump because there wouldn't be any fluid in the vat to pump out. Although we try to avoid any assumptions, it seems we sometimes have to make one or two assumptions on occasion because programming around it either isn't possible or would make the program too complex.

In this example, if the attacker sets starting state to 6 → it never starts.

Why does this not restart the timer each loop?

If you look at the settings for the Start Timer object, you will notice that the Start option is actually a start/continue action, so if the timer is already started and the block is reached, it will just continue running the timer.

What is the benefit to the way we did it here?

If our timer is accidentally stopped during one of our state branches, this ensures that the timer will be restarted.

Can an attacker take advantage of the assumptions we are making on steps 3 and 4?

If an attacker turns off one of our outputs during States 3 or 4, we won't catch it; however, if an attacker turns off an output in States 1, 2, or 5, the next loop (every millisecond) will re-enable it. Of course, none of our states will catch an output that is turned on that isn't supposed to be turned on. Perhaps that is something you can add to your program if you have time remaining in this lab.

if attacker steal the project, every logic can be known and easily attacked (Ethernet)
Generally no auth. to connect and send command to field devices.

via file → open with notepad → Just a simple XML FILE!!!

62 → process logic in XML *Attacker can change anything!*

EXERCISE 1.2: PROGRAMMING A PLC

TAKEAWAYS AND RECOMMENDATIONS

Section takeaways

- The PLC logic you created is generically called your "project files"
 - Project files reveal EVERYTHING about your process
 - Attackers will search for these, often left on engineering workstations
- vBuilder is Velocio's software to program their PLCs
- If Stuxnet had attacked Velocio's PLC, it would have used vBuilder and the project files you just created

Recommendations to owner/operators

- Keep attackers away from your controllers, project files, and software

Recommendations to vendors

- Explore methods to further limit who can program controllers

This page intentionally left blank.

Course Roadmap

Day 1: ICS Overview

Day 2: Field Devices and Controllers

Day 3: Supervisory Systems

Day 4: Workstations and Servers

Day 5: ICS Security Governance

1. Introduction
2. GICSP Overview
3. ICS Concepts
 - Processes and Roles
 - Industries
 - **Exercise 1.1: Learning from Peers**
4. Purdue Levels 0 and 1
 - Controllers and Field Devices
 - Programming Controllers
 - **Exercise 1.2: Programming a PLC**
5. Purdue Levels 2 and 3
 - HMIs, Historians, Alarm Servers
 - Special Applications and Master Servers
 - Control Rooms and Plants
 - SCADA
 - **Exercise 1.3: Programming an HMI**
6. IT and ICS Differences
 - ICS Life Cycle Challenges
7. Physical and Cybersecurity
8. Secure ICS Network Architectures
 - ICS410 Reference Model
 - Design Example
 - **Exercise 1.4: Architecting a Secure DCS**

This page intentionally left blank.

Purdue Levels 2 and 3

Supervisory via HMIs, Historians, and Alarm Servers

Specialized applications for special purposes

Master servers provide control system management

This page intentionally left blank.

SUPERVISORY SYSTEMS AT LEVEL 2 AND 3

There are a number of different systems used to supervise (monitor and control) the ICS processes

- Human Machine Interfaces (HMIs)
- Historians
- Alarm Servers
- Engineering Workstations

All of these supervisor systems may exist in both Purdue Level 2 and 3

- Purdue Level 2 for supervisory systems for each process, manufacturing line, or manufacturing cell
- Purdue Level 3 for plant-wide or regional-wide supervisory

This page intentionally left blank.

Extra level of visuality
→ visualize the project
File, Very useful
for attachments

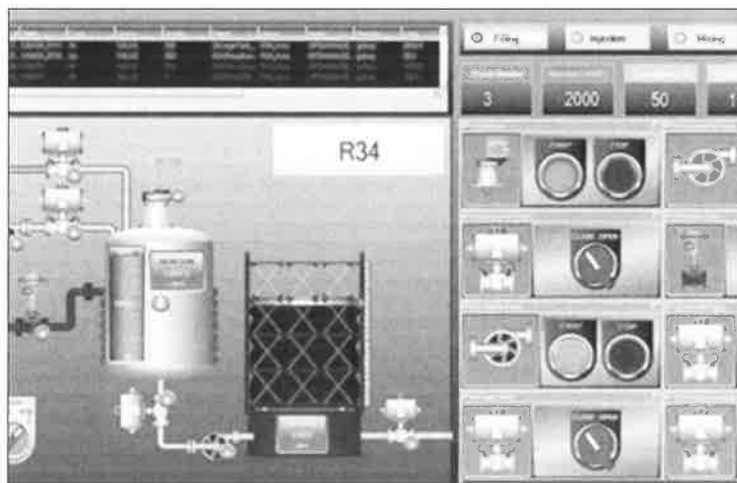
HUMAN MACHINE INTERFACES (HMIs)

Presents process data to human operator

- Visual model created by integrator
- Displays alerts for operators
- Manual control of the process

Despite the seeming importance, this is an optional component

- Control systems can be designed without HMIs
- Some control systems can function for limited amounts of time
- Can be a great defense to disconnect compromised HMIs
- Disconnection can impact safety and reliability if Engineer doesn't design the system for this



SANS

ICS410 | ICS/SCADA Security Essentials

67

The Human Machine Interface is what most people think of first when considering a control system (much like your grandmother who brings her flat-screen monitor in to get a virus removed and leaves the tower at home). This is the GUI for the process, and yes, most of them look that old (it's a recurring theme you'll see throughout the class).

The HMI is usually organized as a model diagram of the process. If you are looking at a chemical system, the screen is going to contain pump icons, tanks and levels, flow indicators, and agitator indicators to let the operator know what's happening with the process. This diagram was created by the process integrator or operator when the system was being assembled.

Additionally, an HMI is responsible for displaying important information to the operator. If a chemical tank is about to overflow, for example, the operator probably knew about that immediately (in addition to the system exacting safety logic and automatically shutting down pumps).

By the description, one would think the HMI is a primary component of a control system, but that's not necessarily true. The HMI is an optional component that just makes things easier. We'll discuss where later, but the logic for the automated process itself (such as which pumps to turn on when, how long to mix the chemicals, etc.) is located elsewhere in other components. The HMI exists to collect data from other components and display it to the operator. This way, if the HMI fails, the process can continue.

What the HMI may be indispensable for, however, is for manual control of the process. Manual controls may exist on individual components, but it is not uncommon for ICSs to be very large or to have components in remote locations that make actually visiting the devices problematic.

Reference:

<https://www.wonderware.com/>

HISTORIANS

A historical datastore for ICS process data

offer DBs. Not always.

- Contains event logs and time-series data of tag values
- Useful for engineers to troubleshoot and optimize processes
- Could have traditional GUIs, web interfaces, and API access
- Underlying technology can be based on
 - No-SQL databases
 - Graphing databases
 - Relational database (SQL)
 - Basic file servers and almost anything else you can imagine

Business needs access to historian data for financial purposes

- A major driver of connectivity between the business and control networks
- Billing customers (electric/gas/water utilities)
- Track manufacturing numbers, and for many other reasons

Most of the vendors that create controllers also offer their own historians. These historians can be based on a large number of different underlying technologies, so don't make assumptions here. In complex control systems like the electric grid, higher-level aggregate historians are used to gather all the information from lower-level historians and systems. Of these higher-level historians, PI Historian, from OSI Soft, is one of the most popular.

Data in a historian can usually be displayed and analyzed through a GUI, web interface, or set of APIs. The data in the historian may be displayed as a simple, time-ordered log or as a trend graph.

- operational logs, error states, process information → Getting measurement from controllers. Records.

- Historian data is generally linked to SAPs. (ERP)
money comes from data. → attackers love the information inside historians'

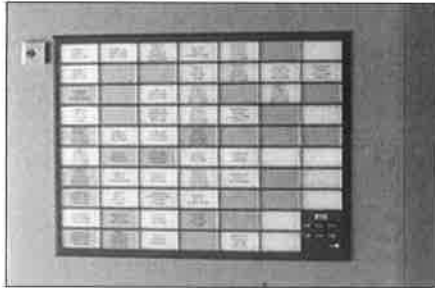
Level 3

ALARM SERVERS

An alarm informs operators of an abnormal event or condition

Alarms may be visual, audible, or digital

Annunciator panels aid in locating problem



An *alarm* is how an operator is informed of any event or condition that must be brought to their attention. Alarms might be communicated to an operator with an audible noise or by a visual indication on either an HMI or a dedicated annunciator panel.

In most ICS environments, alarms will occur regularly and could indicate something that is a significant error or they could simply indicate that something has a small issue or has been disabled for maintenance. Most ICSs will categorize and prioritize alarms so that operators can easily tell which alarms should be reviewed first.

These are together. They can access devices, then control, program, project files store in their PCs...
Lynpice11 1 23

ENGINEERING / OPERATOR / TECHNICIAN WORKSTATIONS

Workstations and/or laptops with software for the specific job function

Engineering workstations are the largest target

- Primarily used for making changes
- Often have greater access to ICS than operator or technician workstations

Workstations often contain

- Device management software
- Project files for those devices
- Runtime libraries that software uses



SANS

ICS410 | ICS/SCADA Security Essentials

70

An engineering workstation is a computer used to make changes or perform maintenance on industrial control systems. The engineering workstation is usually dedicated to the task; however, it could also be used as an operator workstation.

In many environments, due to the tasks that must be performed (reconfiguration of ICS as opposed to monitor and control), the engineering workstation may have greater application or network rights than other devices in an ICS network. It's imperative that the engineering workstation be protected equally to other ICS assets on a control network.

"The engineering workstation is usually a high-end very reliable computing platform designed for configuration, maintenance and diagnostics of the control system applications and other control system equipment. The system is usually made up of redundant hard disk drives, high speed network interface, reliable CPUs, performance graphics hardware, and applications that provide configuration and monitoring tools to perform control system application development, compilation and distribution of system modifications." – US DHS ICS-CERT

RUNTIME LIBRARIES

Installed on servers, workstations, and devices

Provide standard means for interacting with ICS

- Ease integration of disparate ICS components
- Customize their systems in a programmatic way
- Connect to other business systems

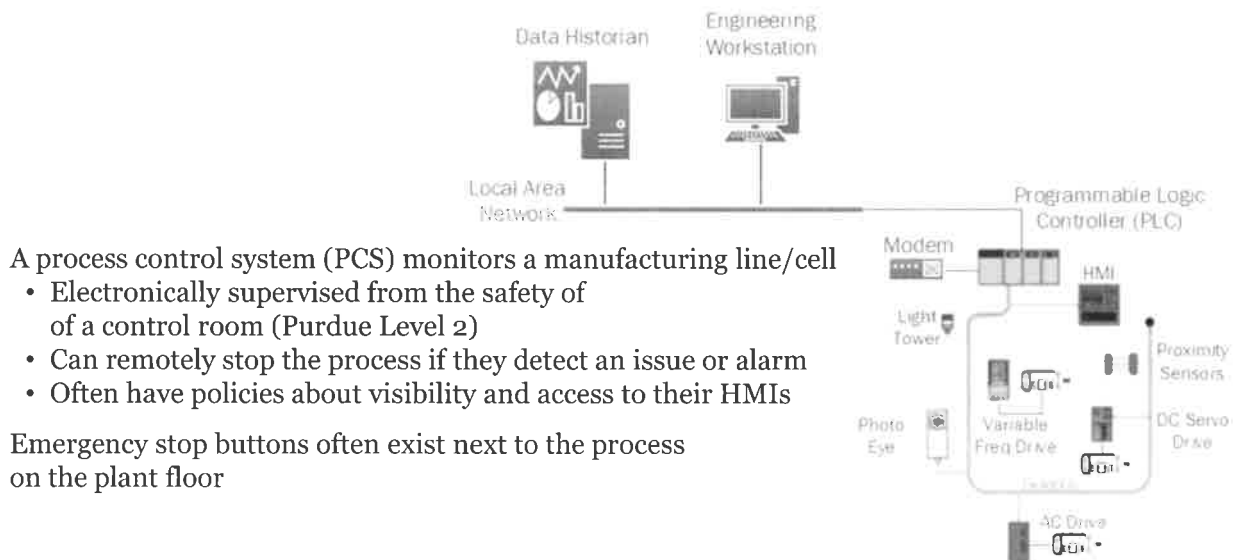
Widely deployed in ICS networks

Example: Stuxnet replaces Siemens Step7 library s7otbxdx.dll with its own malicious version

Runtime Libraries are used to expose and integrate disparate ICS technologies with each other. These runtimes provide standard, programmatic means for interacting with industrial control systems and their connected components, giving organizations a convenient means to customize their ICS and integrate it with other control and business systems.

With this increased ability to customize and interconnect systems comes increased risk. Organizations need to be aware of the increased risk and attack surface that exist when enabling or installing external programs or APIs.

SUPERVISORY WITH CONTROLLERS AND FIELD DEVICES



NIST

ICS410 | ICS/SCADA Security Essentials

73

Process control systems enable automation, thereby allowing a small staff of operating personnel to operate a complex process from a central control room. A process control system monitors the manufacturing environment and electronically controls the process or manufacturing flow based on the various setpoints given by the user.

In a manufacturing setup, there will be different parameters for critical processes that have to be monitored. The real-time values of these parameters will be fed to a central control system. These values are compared with the preset setpoints through feedback systems, and the necessary alerts are output on the display system so that corrective action can be taken.

The diagram above is an example of a simple PCS architecture. Typically, a process-engineered design processes around a Programmable Logic Controller (PLC) with system inputs and outputs all connected and programmed in the PLC.

Image Source: NIST SP800-82

<http://nvlpubs.nist.gov/nistpubs/Legacy/SP/nistspecialpublication800-82.pdf>

MASTER SERVERS

Most large and complex control systems are system of systems

- Have some kind of master server supervising all the lower controllers
- Usually sit in Purdue Level 3

Each industry has specialized names for these systems

- Manufacturing: Manufacturing Execution System (MES)
- Electricity industry: Energy Management System (EMS)
- Building Automation: Building Management System (BMS)

An energy management system (EMS) is a computer-based system used to monitor, control, and optimize the performance of the generation, transmission, and distribution systems. The monitor and control functions are known as SCADA, and the computer technology is also referred to as SCADA/EMS or EMS/SCADA. Energy management systems are also often commonly used by individual commercial entities to monitor, measure, and control their electrical building loads. Energy management systems can be used to centrally control devices like HVAC units and lighting systems across multiple locations, such as retail, grocery, and restaurant sites. Energy management systems can also provide metering, submetering, and monitoring functions that allow the facility and building managers to gather data and insight that allows them to make more informed decisions about energy activities across their sites.

A Building Management System (BMS) is a computer-based control system installed in buildings. It controls and monitors the building's mechanical and electrical equipment, such as ventilation, lighting, power systems, fire systems, and security systems. A BMS consists of software and hardware. The software program can be proprietary using such protocols as C-bus, Profibus, and so on. Vendors are also producing BMSs that integrate using internet protocols and open standards such as DeviceNet, SOAP, XML, BACnet, LonWorks, and Modbus.

Building Management Systems are most commonly implemented in large projects with extensive mechanical, electrical, and plumbing systems. Systems linked to a BMS typically represent 40% of a building's energy usage; if lighting is included, this number approaches 70%. BMSs are a critical component to managing energy demand. Improperly configured BMSs are believed to account for 20% of building energy usage, or approximately 8% of total energy usage in the US.

In addition to controlling the building's internal environment, BMSs are sometimes linked to access control (turnstiles and access doors controlling who is allowed access and egress to the building) or other security systems, such as closed-circuit television (CCTV) and motion detectors. Fire alarm systems and elevators are also sometimes linked to a BMS for monitoring. In case a fire is detected, then only the fire alarm panel could shut off dampers in the ventilation system to stop smoke spreading and send all the elevators to the ground floor and park them to prevent people from using them in the event of a fire.

ANALYTIC APPLICATIONS

Many ICS industries generate analytics from process datasets

- Some sold as a commercial solution by various vendors
- Some custom created in-house

Example: Contingency analysis applications

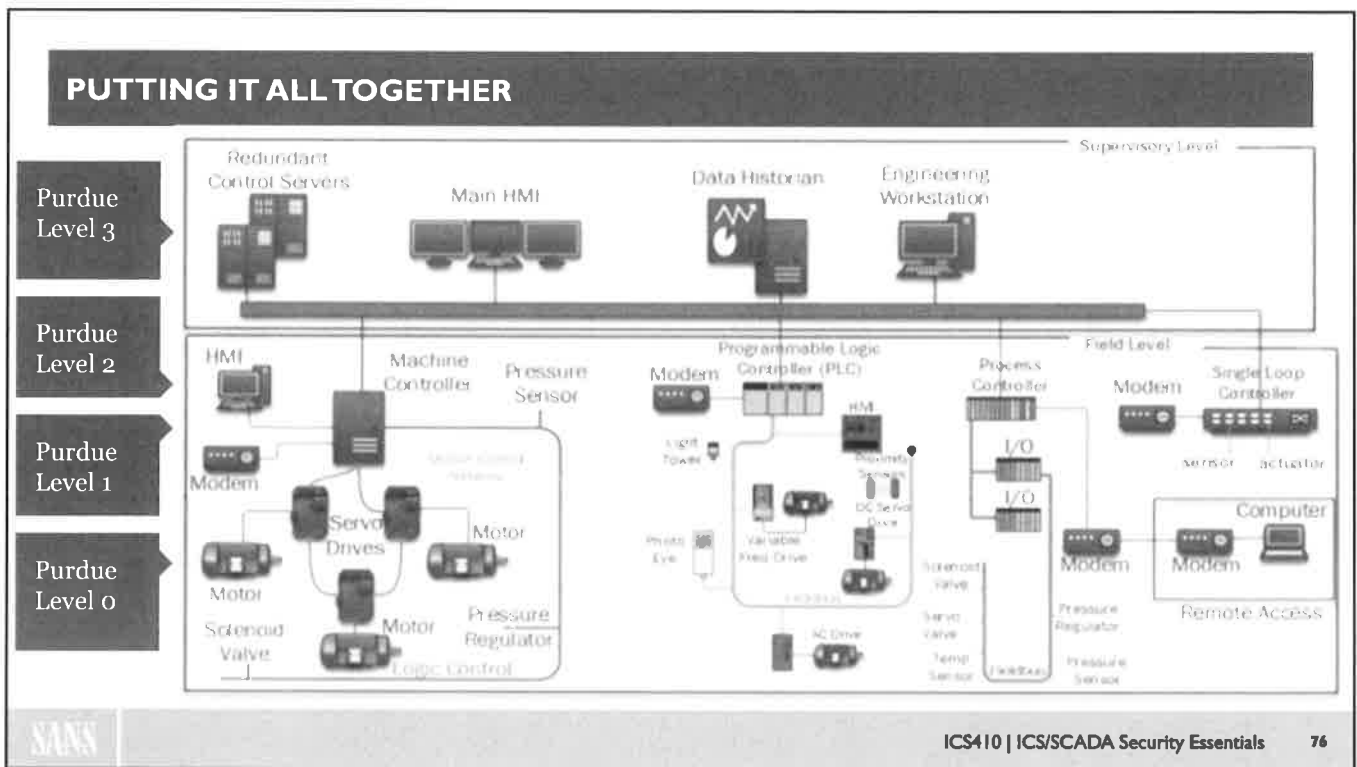
- Identify potentially harmful parts of the process
- Monitor events, thresholds, configurations
- Take actions such as tripping breakers, opening/closing valves, and sounding alarms
- Example: State estimators in power industry

ex/1) very people want sth, raise the price

Contingency analysis is the process where potentially harmful events, thresholds, or configurations are identified so that contingent actions may be taken if the criteria are met.

Actions that may be taken could include:

- Tripping breakers
- Opening/closing valves
- Sounding alarms



NIST definition: DCSs are used to control industrial processes such as electric power generation, oil refineries, water, and wastewater treatment, as well as chemical, food, and automotive production. DCSs are integrated as a control architecture containing a supervisory level of control overseeing multiple, integrated subsystems that are responsible for controlling the details of a localized process. Product and process control are usually achieved by deploying feedback or feed-forward control loops whereby key product and/or process conditions are automatically maintained around a desired setpoint.

To accomplish the desired product and/or process tolerance around a specified setpoint, specific PLCs are employed in the field and proportional, integral, and/or derivative settings on the PLC are tuned to provide the desired tolerance as well as the rate of self-correction during process upsets. DCSs are used extensively in process-based industries.

The slide shows example DCS architecture and components. Note the logical groupings of field-level components, supervisory-level components, and corporate or business devices.

Image Source: NIST SP800-82

<https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-82r2.pdf>

TAKEAWAYS AND RECOMMENDATIONS

Section takeaways

- This is just a small taste of what can be found in Layers 2 and 3
- Attackers' level of interest is often in this order
 - HMIs
 - Engineering workstations with project files
 - Historians and master servers
 - Alarm servers
- This varies based on industry and attacker's intent
- Most ICSs have several layers of control: From master server, to supervisory, to controllers, to field devices

SANS

ICS410 | ICS/SCADA Security Essentials

77

This page intentionally left blank.

→ Sometimes they have ability to control and send data.
→ Sometimes operator doesn't have ability to control. But when alarm goes off, system let operator to push command.

Course Roadmap

Day 1: ICS Overview

Day 2: Field Devices and Controllers

Day 3: Supervisory Systems

Day 4: Workstations and Servers

Day 5: ICS Security Governance

1. Introduction
2. GICSP Overview
3. ICS Concepts
 - Processes and Roles
 - Industries
 - **Exercise 1.1: Learning from Peers**
4. Purdue Levels 0 and 1
 - Controllers and Field Devices
 - Programming Controllers
 - **Exercise 1.2: Programming a PLC**
5. Purdue Levels 2 and 3
 - HMIs, Historians, Alarm Servers
 - Special Applications and Master Servers
 - Control Rooms and Plants
 - SCADA
 - **Exercise 1.3: Programming an HMI**
6. IT and ICS Differences
 - ICS Life Cycle Challenges
7. Physical and Cybersecurity
8. Secure ICS Network Architectures
 - ICS410 Reference Model
 - Design Example
 - **Exercise 1.4: Architecting a Secure DCS**

This page intentionally left blank.

SCADA

The highest level of supervisory monitoring and control

This page intentionally left blank.

SUPERVISORY CONTROL AND DATA ACQUISITION (SCADA)

SCADA is a term that varies in use between different ICS industries

- Manufacturing industries use the term for their supervisory systems
- Electricity, oil, gas, and water industries use the term for regional supervisory and control that typically span large geographic distances
- The same vendors often sell the same solutions to both groups

Common features of SCADA between the industries

- SCADA technologies are usually TCP/IP based — almost always. it needs routing
- SCADA solutions tend to be more vendor neutral with standardized protocols
- DCSs usually have their own integrated supervisory solution not called SCADA
- A plant-wide SCADA solution may sit above lower DCSs

In regional SCADA and plant SCADA, both are usually considered Purdue Level 3

SCADA systems historically distinguish themselves from other ICSs by being large-scale processes that can include multiple sites and large distances. These processes include industrial, infrastructure, and facility-based processes, as described below:

- Industrial processes include those of manufacturing, production, power generation, fabrication, and refining, and may run in continuous, batch, repetitive, or discrete modes.
- Infrastructure processes may be public or private and include water treatment and distribution, wastewater collection and treatment, oil and gas pipelines, electrical power transmission and distribution, wind farms, civil defense siren systems, and large communication systems.

Broad ~~range~~ control. That's why generally L3

EXAMPLE OF SCADA SCALABILITY

Midcontinent Independent System Operator, Inc. (MISO)

- Non-profit, member-based organization
- Administer wholesale electricity markets
- Does not own any generation, transmission, or distribution assets
- Reliability Coordinator for their members
- Provides the wide area reliability view over their region

MISO example from Electricity

- 15 states and 1 Canadian province
- 205,759 MW generation
- 65,250 miles transmission
- 282,163 SCADA data points



The Midcontinent Independent System Operator, Inc. (MISO) is a not-for-profit, member-based organization administering wholesale electricity markets that provide their customers with valued service, reliable, cost-effective systems and operations, dependable and transparent prices, open access to markets, and planning for long-term efficiency. From a Bulk Electric System perspective, MISO does not own any generation, transmission, or distribution assets. MISO acts as the Reliability Coordinator for their members and provides the wide-area reliability view over their region. To perform this task MISO has connectivity and receives real-time data points from member electric system networks. This data is used in contingency analysis, state estimation, simulation studies, outage scheduling, and transmission planning activities. Communications from member companies to MISO typically occur over ICCP Inter-Control Center Communications Protocol, which will be discussed later in the course.

Reference:

<https://www.misoenergy.org>

DCS → sth that does sth you want for a specific region
RTU → similar but more general purpose. Orchestration might be done.
if can have PLC capabilities. It controls PLCs and runs together.
Commands

REMOTE TERMINAL UNIT (RTU)

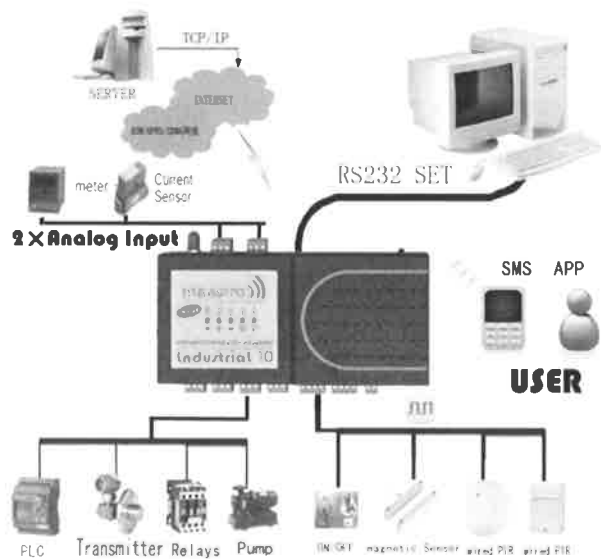
Computer-based devices that act as a control and gateway in SCADA networks

- Usually used with WAN connections
- Can have large geographic distances between RTU and main systems
- Multiplexes signals from master terminal unit (MTU) to controllers and field devices behind the RTU

RTU can also stand for remote telemetry unit

RTUs different from PLCs in some ways:

- Usually only runs simple logic
- More suitable for large geographical areas
- Differences are decreasing each day



SANS

ICS410 | ICS/SCADA Security Essentials

82

Above is an image of a remote terminal unit (RTU). RTUs come in a variety of sizes and capabilities. Most RTUs are cabinet mounted with hardwired points to interface with field devices.

An RTU is installed to report local system information and otherwise communicate with the upstream supervisory system. In a Distributed Control System, you may have many separate automated process areas. For example, a typical power company will have a central supervisory system where operators sit on an HMI and each substation will be connected to the central system by utilizing RTUs. RTUs need to communicate with the central system and potentially with Intelligent Electronic Devices (IEDs). This communication is typically done via serial (RS-232 or RS-485), fiber, or wireless comms like cellular networks. Communications are covered in more detail in another module.

I/O Connections

Like a PLC, an RTU has an array of input/output connections. These could be digital inputs (like a sensor detecting the closed/open state of a container), analog inputs (such as temperature data), or digital outputs (like an electric relay).

Some Logic

Though generally "dumber" than a PLC, RTUs usually run simple autonomous programs to provide redundancy and safety. For example, an RTU at a power substation will have logic to change the behavior when a technician is working on a particular line and has manually disabled a line. This prevents someone at the operator console from closing a relay while technicians are performing maintenance, thus avoiding injury.

Differences from PLC

RTUs are more suitable for large geographical areas. They utilize wireless communications (e.g., cellular networks) to communicate back to the supervisory system. PLCs are generally more suited for a local control system that exists all in one location and uses physical media for communications. More recently, however, the line has been blurring. The extreme drop in microcontroller prices has caused the two devices to share more and more qualities.

Image Source: goo.gl/DeuMkv

Another level of multiplexer to WAN.

FRONT-END PROCESSOR (FEP)

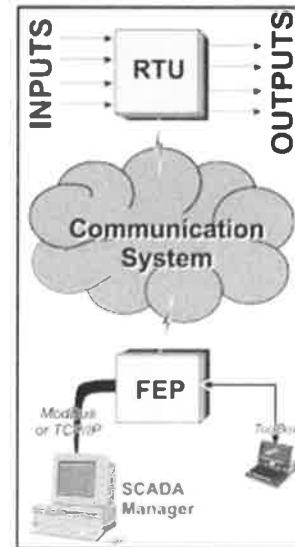
In complex SCADA systems, FEPs replace and extend a number of traditional systems

- Data-communication
- Alarm dispatching
- Automatic data collection
- Backup operation
- Diagnostics
- Event registration

} can be any purpose

Facilitate communications between control center and most SCADA outstations

- Communication actions
- Supervision of administration
- Informing the central system



SANS

ICS410 | ICS/SCADA Security Essentials

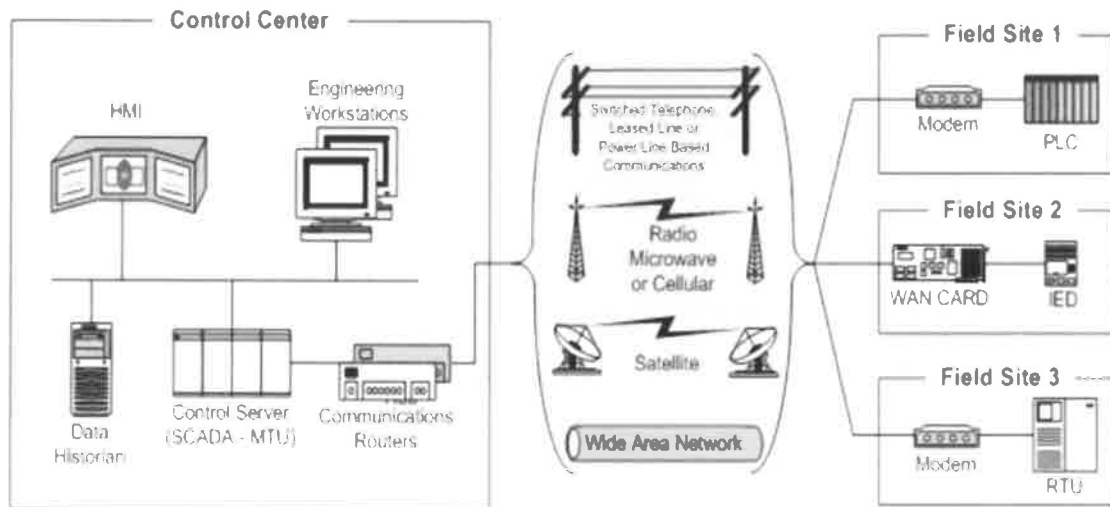
83

The FEP takes care of all communication actions, supervises the communication with associated administration, and informs the central system. It integrates functions that are otherwise only found in separate applications such as data communication, alarm dispatching, automatic data collection, backup operation, diagnostics, and event registration. To build these functions in a SCADA or another host application requires extensive engineering, especially for larger systems.

Reference:

<http://scadaautomation.blogspot.com/2009/07/moscad-system-overview.html>

REGIONAL SCADA

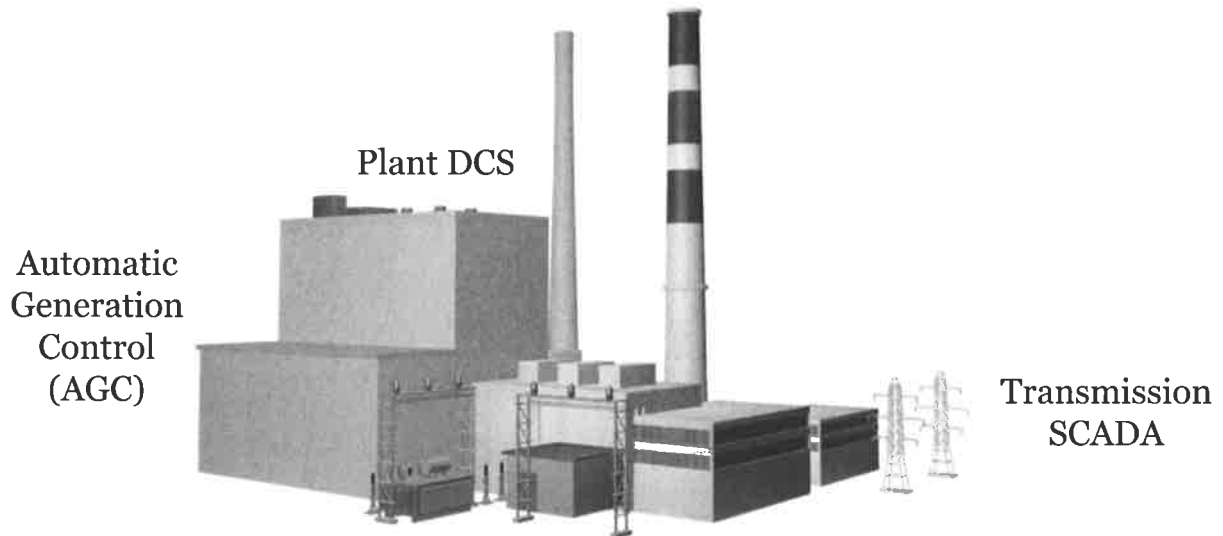


SCADA can span large geographic areas. RTUs may send telemetry data over various communication channels, including cellular, telephone/modem, microwave, serial, Ethernet, or other radio communications. These communications may have to travel over many miles (such as equipment on an oil pipeline, disparate power substations, or ocean drilling operations). This communication may be disrupted or sporadic. If you wish to communicate with a field device that uses a modem to dial into the main system, for instance, you may have to wait for it to initiate the communications, resulting in limited windows of opportunity, or you may consider implementing a modem with call back or dial back security features that allow for initiated requests and some security benefits. Weather may also affect communication ability, causing temporary disruption or a long-term outage requiring a technician to travel out to the equipment site.

Image Source: NIST SP800-82

<https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-82r2.pdf>

DCS PLANTS CAN BE SCADA ENDPOINTS TOO



SANS

ICS410 | ICS/SCADA Security Essentials

85

Multi-unit thermal power plants rely upon DCS technology and PLC-based subsystems to manage the process of generating electricity. Plants are typically connected to the transmission system through switchyards that contain Transmission SCADA systems. Power plants are dispatched from balancing authorities using Automatic Generation Control systems.

TIMING/CYCLES AND SYSTEM SIZE

Describing a specific control system at a high level can be misleading. There are some accepted general descriptions used to differentiate types of systems/applications and provide a gross idea of the size

Type of ICS	Polling Rate
Regional SCADA	1 second to 1 minute
Plant SCADA	.05 to 1 second
Controllers	1ms to 1 second
Field Devices and SIS	1ms to 1 second

We can survive without SCADA but not without Field devices.

depends on polling frequency

It is difficult to compare and generalize industrial systems, but we can consider key attributes such as the timing cycle, scan rate, or SCADA polling rate that is acceptable or the number of components or I/O points to provide a general description. Systems will be configured to collect a signal at some scan/polling rate; the system could store the results of every scan/poll and perform necessary calculations and monitor state, value, and/or trends. For example, a control engineer will select an appropriate sampling rate for a control loop to give proper control.

Timing/cycles describe the control system application and give someone an immediate idea of what type of process is being monitored and controlled. SCADA systems describe control systems that span over wide geographic areas. SCADA systems need to acquire data by having the RTUs or PLCs scan or poll the connected field inputs of wide-area networks. The data collected is completed during an acceptable time cycle and can be configured based on the number of points, the amount of data, and the communication rates available across channels. The SCADA engineer will select an appropriate polling rate for the system. It is very common for electric power SCADA systems to select a 1- to 3-second polling rate.

Distributed Control Systems (DCS) can cover a complex or plant facility and can typically take advantage of higher-speed communications and local networks. They can achieve much faster scan/polling rates. Some control applications, such as motor controls, can require very fast cycles (e.g., in the millisecond range).

Scan rates for instrument readings need to be supported by the hardware and are determined by the application of what is being monitored and controlled. Hardware will often determine the maximum allowable scan rate, and the process will require a desired scan rate or acceptable range. Warning: We don't like generalizations, but they are made to provide descriptions of things. Not all systems will fit general definitions. It is important to note that reliable distinctions between RTUs, PLCs, and terms to describe control systems have faded. Newer technologies have blurred the lines of existing terminology.

Describing the size of an automation and control system is usually referenced by the number of I/O points for key components, such as controllers. A plant may have a control system comprised of a few hundred I/O points to several tens of thousands of points. Some wide-area SCADA systems are monitoring tens to hundreds of thousands of points.

TAKEAWAYS AND RECOMMENDATIONS

Section takeaways:

- SCADA can mean regional or plant-wide control depending on the industry
 - It is usually considered Purdue Level 3 in both scenarios
 - It is usually TCP/IP based using industry standard protocols
- From a cybersecurity perspective, SCADA is more exposed
 - In Plant-wide SCADA, it is the first system attackers usually access
 - In regional SCADA, weak points are WAN providers and unmanned stations, but network defenses can aid us here

This page intentionally left blank.

Course Roadmap

Day 1: ICS Overview

Day 2: Field Devices and Controllers

Day 3: Supervisory Systems

Day 4: Workstations and Servers

Day 5: ICS Security Governance

1. Introduction
2. GICSP Overview
3. ICS Concepts
 - Processes and Roles
 - Industries
 - **Exercise 1.1: Learning from Peers**
4. Purdue Levels 0 and 1
 - Controllers and Field Devices
 - Programming Controllers
 - **Exercise 1.2: Programming a PLC**
5. Purdue Levels 2 and 3
 - HMIs, Historians, Alarm Servers
 - Special Applications and Master Servers
 - Control Rooms and Plants
 - SCADA
 - **Exercise 1.3: Programming an HMI**
6. IT and ICS Differences
 - ICS Life Cycle Challenges
7. Physical and Cybersecurity
8. Secure ICS Network Architectures
 - ICS410 Reference Model
 - Design Example
 - **Exercise 1.4: Architecting a Secure DCS**

This page intentionally left blank.

EXERCISE 1.3: PROGRAMMING AN HMI

DURATION TIME: 25 MINUTES

We are going to create an HMI for our PLC process we created in the last exercise.

We are going to reprogram our PLC with a slightly modified version of our program, one with write access to tags enabled.

OBJECTIVES

Understand how engineers create and modify HMIs.

Learn what attackers can do if they access HMI software and the HMI.

PREPARATION

Start the Windows 10 virtual machine

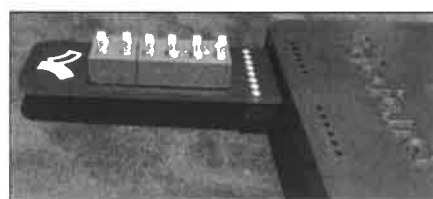
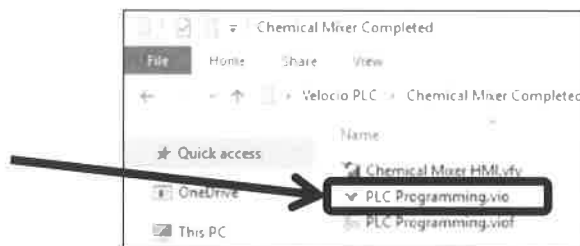
Prepare your Velocio PLC for use

This page intentionally left blank.

EXERCISE 1.3: PROGRAMMING AN HMI

PREREQUISITE PROGRAMMING

1. Open the "Velocio PLC" folder on the desktop of your Windows 10 VM
2. Open the "Chemical Mixer Completed" folder
3. Open the "PLC Programming" file that has the Velocio logo
4. Answer "Yes" to the prompt about vBuilder modifying your computer
5. Turn off all the switches on the input switchboard (away from the numbers)
6. Connect your input switchboard on the **LEFT SIDE** of your PLC
7. Use the USB cable to connect your PLC to the computer



We are going to use a slightly modified version of our previous program that I've prepared for you. This one has all tags remotely writable so our HMI can change their values. It also has Modbus RTU enabled for a later lab. The program you want to program to the PLC is the "Chemical Mixer Completed" folder in the "Velocio PLC" folder on your desktop.

- Open the "Velocio PLC" folder on the desktop of your Windows 10 VM.
- Open the "Chemical Mixer Completed" folder.
- Open the "PLC Programming" file that has the Velocio logo.
- Answer "Yes" to the prompt about vBuilder modifying your computer.

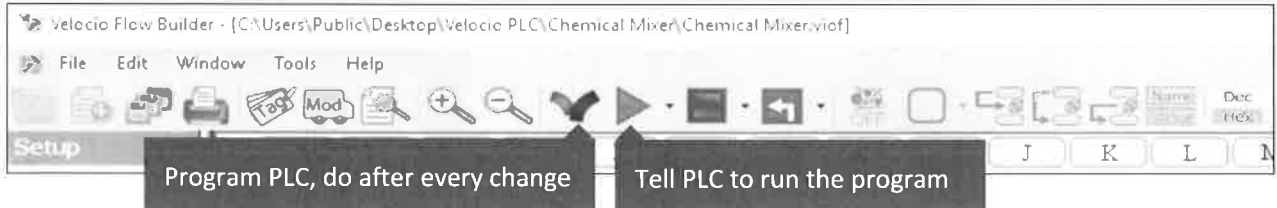
Velocio needs to be able to install drivers to access and program your PLC, so it needs a bit more permission than most programs; thus, you will see this prompt every time you start Velocio. Now we need to prepare our PLC to connect to our computers.

- Turn off all the switches on the input switchboard (away from the numbers).
- Connect your input switchboard on the **LEFT SIDE** of your PLC.
- Use the USB cable to connect your PLC to the computer.

Once you have it connected to your computer, there is no visual indicator in Velocio to show it is connected. We will just assume it is connected until we try programming it at the end of the lab. However, please make sure inside of VMware that your PLC is connected to your Windows 10 virtual machine, not your host machine. Not having your PLC connected to the correct VM is the most likely reason that the programming step at the end of this exercise fails.

EXERCISE 1.3: PROGRAMMING AN HMI

PREREQUISITE RUNNING THE PROGRAM



8. Program the PLC by clicking on the Velocio icon

9. Run the program by clicking the Play button

10. Flip switch 1 to on, and verify the output LEDs on the PLC are correct:

Seconds 0–2: Output LED 1	● ○ ○ ○ ○ ○
Seconds 2–4: Output LED 2, 3, and 4	○ ● ● ● ○ ○
Seconds 4–6: Output LED 3 and 4	○ ○ ● ● ○ ○
Seconds 6–10: Output LED 4	○ ○ ○ ● ○ ○
Seconds 10–15: Output LED 5	○ ○ ○ ○ ● ○

If everything is working, close vBuilder

MAX

Now that we have our program written, we need to program our PLC and see how it runs. To do this, follow the steps above.

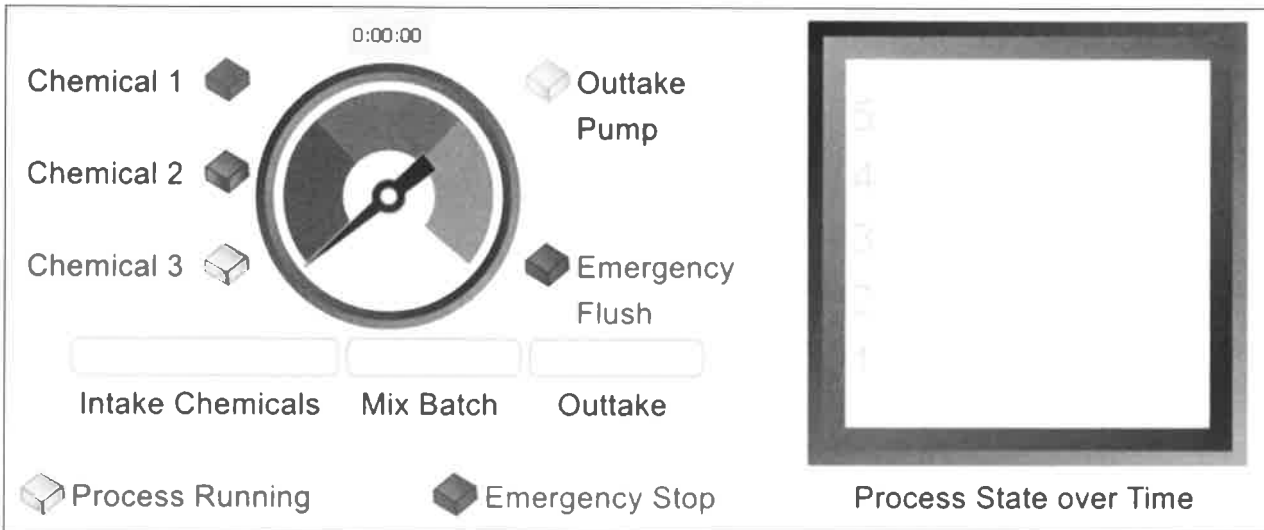
When you run the program, and turn on the **RunProcess** on your PLC input board (switch 1), you should see the output lights start cycling through different patterns. Here are the patterns you should see:

- Seconds 0–2: Output LED 1 (the one closest to the input board)
- Seconds 2–4: Output LED 2, 3, and 4
- Seconds 4–6: Output LED 3 and 4
- Seconds 6–10: Output LED 4
- Seconds 10–15: Output LED 5

Once this is working, close vBuilder without stopping the PLC. You should see the PLC outputs still cycling through the settings.

EXERCISE 1.3: PROGRAMMING AN HMI

THIS IS THE HMI WE ARE GOING TO CREATE



SANS

ICS410 | ICS/SCADA Security Essentials

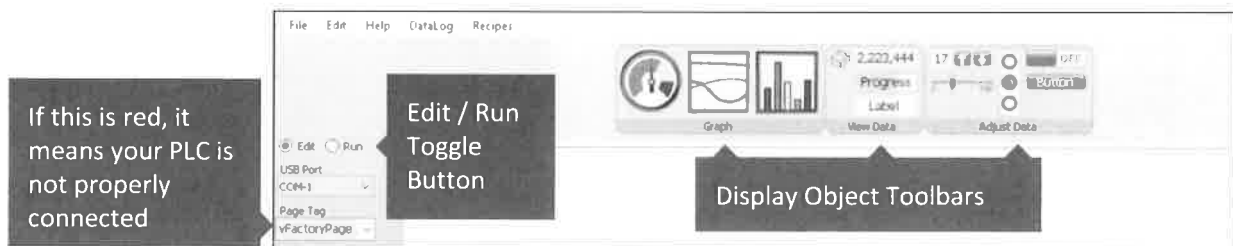
92

For this exercise, we are going to create a Human Machine Interface (HMI) to display and record the state of the chemical mixer process we programmed in the previous lab. Our final product should look similar to the screenshot above.

EXERCISE 1.3: PROGRAMMING AN HMI

OPENVFACTORY

11. Confirm you performed the prerequisite steps in the first two slides of this exercise
12. Verify the PLC is running the Chemical Mixer program with output LEDs changing
13. Quit vBuilder if it is running
14. Double-click on the vFactory shortcut link on the desktop to start the program
15. Once the program is open, create a new project and give it a name
16. Use the tools indicated in the diagram below for the following steps



SANS

ICS410 | ICS/SCADA Security Essentials

93

vFactory is the software from Velocio used to create HMIs for their PLCs. We will be building a simple single-page design; however, this tool can be used to create more complex multi-page HMIs if you need to display different screens based on state or operator selection.

Follow the instructions above to create a new vFactory project for our HMI.

Make sure the Page Tag dropdown box isn't red. If it is, this means your vFactory isn't connected to your PLC. Close vFactory and check your cabling and VMware configurations to make sure it is connected to your Windows 10 virtual machine. Once connected, try opening vFactory again. You should also note that vBuilder and vFactory cannot run at the same time. You will encounter a conflict when both applications try to control the PLC at the same time.

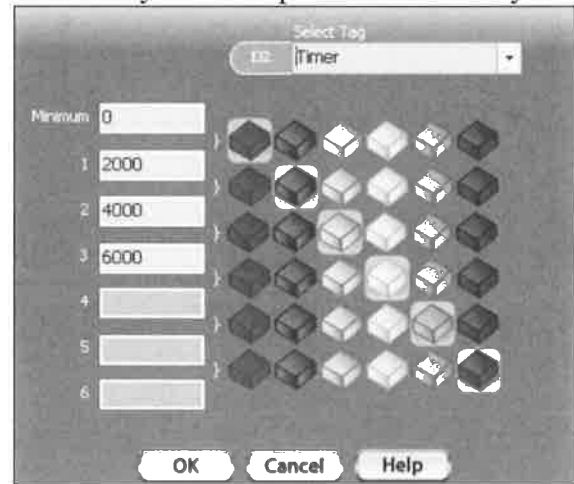
EXERCISE 1.3: PROGRAMMING AN HMI

ADD A METER DISPLAY

17. Drag a meter display from the Object toolbar to the middle of your workspace. You can always move it around later



18. Have it display the **Timer** tag and set four boxes on the left to **0**, **2000**, **4000**, and **6000** to represent the millisecond thresholds for the three intake pumps



Let's create the central display object. This will be a meter display. We will use it to show the intake pump progression for our process. Because we do not have actual sensors telling us fill levels in our design, we will have to use the process Timer to indicate how full the tank is. If you remember, our process had three intake pumps. The first pump ran for 2 seconds, the second two pumps both ran for 2 seconds, and the third pump continued running by itself for 2 more seconds. This gives us a total of 6 seconds of run time. Although it isn't perfect, without level sensors, the Timer value is the best we have to indicate fill level, so we will use that for the meter thresholds between color indicators.

Follow the steps above to add and configure the meter display in your HMI.

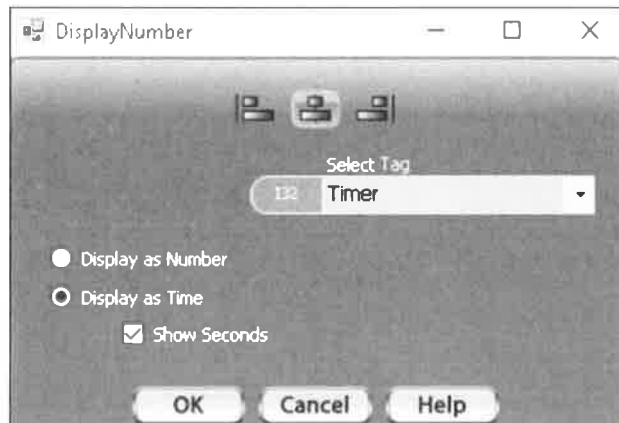
EXERCISE 1.3: PROGRAMMING AN HMI

ADD A TIME DISPLAY

19. Drag a number display that has a number similar to [2,223,484] to the spot above your meter. Once configured, you can adjust the width to center it properly.



20. Have it display the **Timer** tag and configure it to **Display as Time** and **Show Seconds**. At the top of the configure window, you can also have it center the numbers if you wish.



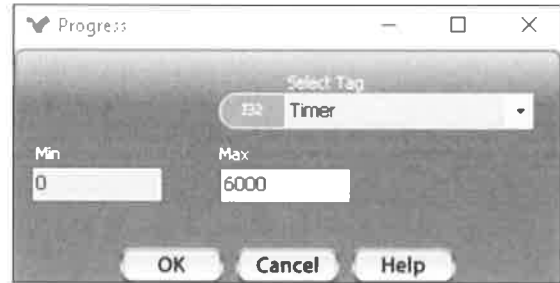
By adding a time display above the meter, we can see the actual value of our timer, helping us to know exactly where in the process we are. This will greatly help if we pause the process. To do this, we will add a number display (the object with the number similar to 2,223,484) and center it above our meter display.

Follow the steps above to add and configure the meter display in your HMI.

EXERCISE 1.3: PROGRAMMING AN HMI

ADD A PROGRESS DISPLAY

21. Drag a progress display to the spot below your meter. If you want, you can use three progress displays, one for each stage as seen to the right. You'll need to adjust width after configuration.
22. Have it display the **Timer** tag and set the range from **0** to **15000**. If you use three displays, set them:
 - 0 to 6000**
 - 6000 to 10000**
 - 10000 to 15000**
23. Drag a label to the spot beneath your progress display(s) and configure appropriately.



By adding a progress display, you can see the visual indicator of the percentage left in the process. If you configure multiple progress displays for different stages, you can increase this visual understanding of the process progress.

Follow the steps above to add and configure the meter display in your HMI.

EXERCISE 1.3: PROGRAMMING AN HMI

ADD LIGHT INDICATORS

24. Drag indicator lights for each of your inputs and outputs and place them around your meters and progress displays. Suggested colors in notes below.



25. Label each of them appropriately, remembering that Chemical 1 is added with IntakePump1 . . .

By adding an indicator light for each input and output, you will immediately know what is on and what is off.

Follow the steps above to add and configure the meter display in your HMI. You can choose any color you wish; however, here are some suggested colors we used in the slide example.

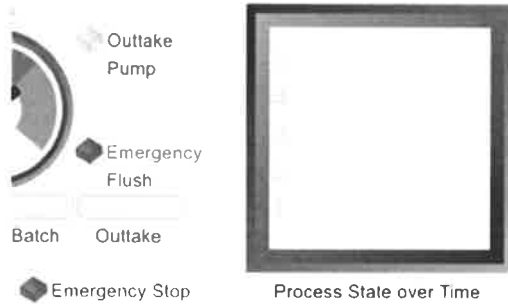
Chemical 1 – Purple
Chemical 2 – Blue
Chemical 3 – Green
Outtake Pump – Yellow
Emergency Flush and Stop – Red
Process Running – Green

When adding labels, you can use the actual Input and Output tags we used in vBuilder, or you can use the more human-friendly references for what that Input or Output is doing as seen above in the screenshot. For example InputPump1 adds Chemical 1, InputPump2 adds Chemical 2, etc.

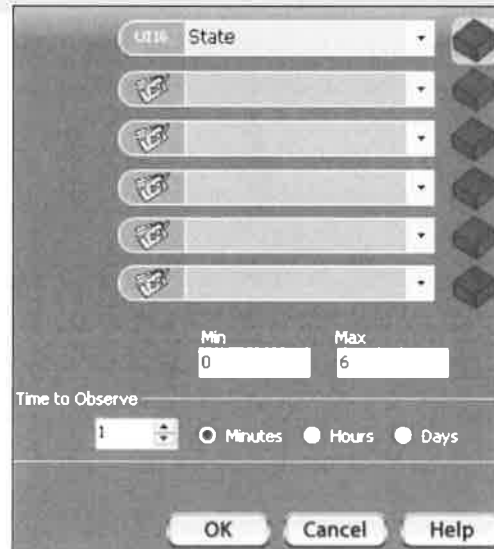
EXERCISE 1.3: PROGRAMMING AN HMI

ADD A LINE GRAPH

26. Drag a line graph display to the spot to the right of your other displays



27. Have it display the **State** tag from **0** to **6** in **1-minute** intervals
28. Add a label below the graph



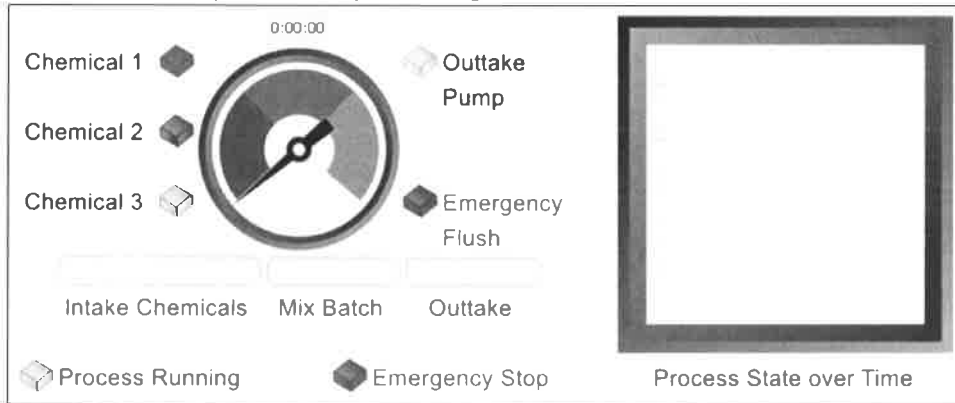
By adding a line graph, you can show historical trends in the process. You can adjust the length of the history as you optimize your process and discover what values the operators believe best suit this process.

Follow the steps above to add and configure the meter display in your HMI.

EXERCISE 1.3: PROGRAMMING AN HMI

RUN AND TROUBLESHOOT

29. Change the radio button to Edit Run and start your process by turning on the RunProcess switch.
30. Troubleshoot any issue you find in your design.



That should be it. Change vFactory from Edit mode to Run mode with the steps above, and as long as we both did our jobs, your HMI should work flawlessly. If you live in the real world and that is not the case, troubleshoot the issue until it is fixed. If you have any time left, try extending your HMI to add other features and displays.

EXERCISE 1.3: PROGRAMMING AN HMI

QUESTIONS FOR THOUGHT

Can you think of anything we could do to make our HMIs help us gain greater insight into our chemical mixer process?

How could we modify our chemical mixer process to provide additional data to our HMI to help operators do their jobs?

- This version of the project files permits the HMI to send control signals
- Experiment with adding objects from the "Adjust Data" tool section

Can you think of anything we could do to make our HMIs help us gain greater insight into our chemical mixer process?

- Answers will vary, but with limited inputs and outputs at our disposal, there are limited options.

How could we modify our chemical mixer process to provide additional data to our HMI to help operators do their jobs?

- Level indicators in the tank would provide greater accuracy to our tank levels.
- Meters on our intake and outtake pumps would help give us accurate measurements of chemicals being mixed and produced.

EXERCISE 1.3: PROGRAMMING AN HMI

TAKEAWAYS AND RECOMMENDATIONS

Section takeaways

- Notice how much easier it was to understand your process with an HMI
- If you were an attacker, an HMI is the easiest and fastest way to modify a process
- We will discuss later in the course how attackers used the HMIs to shut down power in the Ukraine attacks

This page intentionally left blank.

Course Roadmap

Day 1: ICS Overview

Day 2: Field Devices and Controllers

Day 3: Supervisory Systems

Day 4: Workstations and Servers

Day 5: ICS Security Governance

1. Introduction
2. GICSP Overview
3. ICS Concepts
 - Processes and Roles
 - Industries
 - **Exercise 1.1: Learning from Peers**
4. Purdue Levels 0 and 1
 - Controllers and Field Devices
 - Programming Controllers
 - **Exercise 1.2: Programming a PLC**
5. Purdue Levels 2 and 3
 - HMIs, Historians, Alarm Servers
 - Special Applications and Master Servers
 - Control Rooms and Plants
 - SCADA
 - **Exercise 1.3: Programming an HMI**
6. IT and ICS Differences
 - ICS Life Cycle Challenges
7. Physical and Cybersecurity
8. Secure ICS Network Architectures
 - ICS410 Reference Model
 - Design Example
 - **Exercise 1.4: Architecting a Secure DCS**

This page intentionally left blank.

IT and ICS Differences

There are a few: Safety, reliability, security, life cycles, etc.

This page intentionally left blank.

IT AND ICS DIFFERENCES		
IT	ICS	Trend
General skills and competencies available to support	Specific engineering roles with some dedicated technology support	General IT organizations are supporting more ICS applications and functions
Expandable computing power and storage	Computing power and available memory and storage is often constrained in field devices	Field devices are becoming more powerful and capable
High-throughput important, but occasional delays okay	"Real-time," delays are not acceptable, short sense-to-act cycle, deterministic	There are some emerging similarities with mobile and embedded systems
Risk focuses on losses to the business	Risks can exceed losses to the business: - Loss of life and limb - Damage to environment	Static

response in milliseconds and replays every x/records.

Sometimes there is IT backup and supports.

Historical context is important for understanding the culture and security of the ICS space. Historically, ICSs were isolated from any other components or networks. The process control system was designed, built, and tested for one single purpose. With the increased prevalence of internet connectivity, the benefits of remotely controlling and monitoring ICS networks demanded to end the isolation. Connecting ICS devices to the internet allowed for large benefits and cost savings. Not having to send a technician out to get information from field equipment saves labor and vehicle costs. It's much more efficient for the smart grid to self-report on usage statistics that can be immediately analyzed for use in power grid operations than to send someone out to read the meters.

The competencies necessary to support IT traditionally are available in the job market, while the blended need for an ICS environment needs an individual with an understanding of the engineering or operation of the system as well as the cyber components used to operate it.

In today's IT systems, computing power is vast and can be expanded or upgraded with ease; in ICS field devices, the computing power and available memory are often constrained and limited in function. An additional limitation for ICSs is raw processor capability. The processing capability, program space, and memory requirements of encryption algorithms don't have to be very high before they become unreasonable to use on a field device driven by an ultra-low power processor.

The ICS community utilizes a lot of traditional IT assets; however, there are many significant drivers and constraints in ICS environments that should be understood. In addition, there are some trends that can be seen in the IT and ICS communities.

The design of any control facility will be driven by the need to reduce the threat to life and limb at every opportunity. Some environments, such as offshore oil platforms, have such rigorous safety requirements that every square foot is designed to meet a specific need.

ICS AND SAFETY

Most ICS organizations must meet safety standards

- Accepted safety and risk analysis methods
- Safety regulations enforced by governments and industries
- Example: US Occupational Safety and Health Administration (OSHA)

Many ICS organizations have a safety culture

- Personal Protective Equipment (PPE) requirements
- Safety training and drills
- Safety meetings and Stop Work protocols
- Strive to install inherently safe devices and environments

How does your work behavior and work product impact the safety of yourself?
Others? Operations?

The modernization of society claimed many lives and ultimately led to workplace safety standards. Many professionals working across a wide variety of work types are familiar with the safety share, the safety meeting, PPE (Personal Protective Equipment), and Stop Work protocols. This is indicative of the fact that we live in a safety culture. The world of industrial control is no different. Many of the devices we covered earlier come in configurations known as *inherently safe*, meaning they are carefully sealed in such a way that they can operate in hazardous environments without introducing additional danger. An example of this is a sensor designed to take and report temperature and pressure measurements while inside a pipe carrying explosive gas.

There are several types of accepted safety and risk analysis methods to deal with probabilistic risks in control systems and processes. The key for cybersecurity is that the teams conducting the assessments understand what cyber risks are present under normal operation, emergency operations, and when those cyber components are deliberately misused to create an unknown operating state. Risk analysis methods include:

- Process Hazard Analysis (PHA)
- Hazard Operations (HAZOP)
- Layers of Protection Analysis (LOPA)
- Quantitative Risk Analysis (QRA)

Many governments have regulatory bodies that enforce safety, such as the Occupational Safety and Health Administration (OSHA; <https://www.osha.gov/law-regs.html>).

Understand and learn the safety hazards and Personal Protective Equipment requirements of any environment before you enter. Some examples of PPE include hard hat, eye protection, ear protection, rubber soled shoes, flash-resistant clothing, and other specific protective gear (depending on the facility and location). Consider the work you are about to perform and the impacts that work can have on the safety of yourself, others, and the operating environment you are working in. Ensure proper planning, communication, and necessary approvals have been obtained prior to any actions that you may take when performing work in an ICS environment.

ENVIRONMENTAL ISSUES

Well-being of the environment needs to be strictly monitored and regulated in ICS

- Pollution governance
- Proper design of certain processes

Operators must develop contingency plans for their worst-case scenario

- Disaster planning with local authorities
- Escalation to government aid

These risks must be considered in security

Like many other aspects of operating control systems and processes, the well-being of the environment is strictly monitored and regulated. In addition to the governance of pollution and other direct results of operating certain types of processes, asset owner/operators must develop contingency plans for a worst-case scenario. This is especially important for processes that could impact a nearby population if the worst were to happen. The disaster planning process should be exercised with local authorities when possible, as serious events will almost certainly be remediated by the government.

IT AND ICS DIFFERENCES IN RELIABILITY

IT	ICS	Trend
Common operating systems	Mixture of common operating systems (OS) at Purdue level 3, specialized/embedded OS in lower levels	Common OS and x86-based computers in controllers are reemerging
Generally, in environmentally controlled data centers	Field devices must tolerate extreme temperatures, dust, vibrations, etc.	Mobile devices are being designed to tolerate field conditions
Routine windows often available for changes and maintenance	Availability is paramount! Outages scheduled for changes and maintenance. Managed with highly redundant systems.	ICS has traditionally avoided managing system changes and patches, but progress has allowed for accepted practices and use of dedicated tools

A large number of ICS devices are embedded systems. Although some of these systems do run familiar operating systems such as Linux, they are more often systems incapable of running familiar tools. There's no such thing as antivirus, HIDS, or other host-based security mechanisms for a PLC or RTU. For the systems that are running some brand of Linux, there is a high probability that the processor architecture is not x86. ARM, PIC, AVR, PowerPC, Texas Instruments MSP430, and other architectures are common.

Typical operating environment conditions differ greatly from IT environments, with environmentally controlled data centers and ICS environments experiencing extreme temperature fluctuations, dust, vibration, and other various particulate exposures. Variations can also be seen in the tools used within IT and ICS. Many of the security testing tools that are commonly used in IT environments should not be used without modification in ICS environments, as they may cause harm to the operations. Throughout IT environments, common operating systems exist, whereas ICS environments have a blend of common, special purpose, and embedded operating systems.

IT environments often maintain highly available environments, whereas ICS environments need to support real-time operations and data processing. In many IT environments, dark windows or maintenance windows exist to accommodate system management tasks, whereas in ICS environments, availability is paramount and often results in large delays in performance of system maintenance tasks, oftentimes waiting for scheduled or forced process outages.

ICS RELIABILITY

ICS is expensive

Needs to work in a predictable manner

Failures will occur and should be planned for

- Identify components for replacement before failure
- Paradigm also needs to include cybersecurity
- How reliable is a system that is vulnerable to cyber attack?

Reliability  Security

ICS security tasks must be sequenced with additional steps to coordinate, plan, and test

Communication is KEY! Work with ICS teams!

The reliability of any automation system is of critical importance. Systems fail and operators need to be able to identify and respond to these failures. Automation technology is advancing rapidly and more and more systems are being designed that have the capability to detect failing components and replace them before failures occur. This paradigm also needs to be applied to the concept of security. How reliable can a system really be if it is vulnerable to a cyber attack?

With the production cycles that many ICSs operate at, one of the most challenging constraints is having available access to the systems for a period of time that allows for system maintenance. In ICS environments, security tasks must be sequenced in a manner that allows for adequate coordination and communication between operations and IT security. Additionally, planning for blackouts, reversion, and continued operations must be established and communicated to operations to ensure each individual understands his roles and responsibilities based on a dynamic environment. Prior to implementing complex system management tasks, the changes must be tested in a manner that cannot impact operations. At each step, communication is key!

Frameworks for system management in an ICS environment exist and will be discussed later in this course.

IT AND ICS DIFFERENCES IN SECURITY

IT	ICS	Trend
Bring Your Own Device (BYOD) policies are common	Traditionally confined to dedicated devices	ICS is opening up to mobile and tablet device use . . . EEK!!!
Difficult to whitelist	Greater ability to whitelist	ICS is getting more complex
Most security tools are designed for IT	Security tools need to be adapted and/or tested for ICS	Specialized security tools are being designed for ICS (e.g., data diodes, ICS firewalls, Sophia)
Systems and devices often designed with security in mind	Many field devices designed and programmed by engineers without security backgrounds	Vendors are adding security requirements to their designs
Use of standard protocols, many with security features	Proprietary and standard protocols, many designed by engineers	Static with some versions of ICS protocols adding authentication features

like
HMI
Mobile
app

IT environments have seen an explosion in the need to integrate and support a wide variety of personal computing devices, whereas ICS environments have been relatively constrained to traditional dedicated computing devices.

Application whitelisting has been challenging in some IT environments due to the dynamic nature of the systems and applications in use. In ICS environments, application whitelisting has been easier to implement due to the deterministic nature of the environment.

Other considerations in adapting enterprise IT tools to ICS involve how security logs and data are accessed and monitored. Both realms require people to monitor the security systems and communications to make use of the benefits they provide. In ICS, this may be more difficult for two reasons. Culturally, ICS operators are used to respond to alarms and alerts from systems, but it doesn't take very many false positives for a security system to be ignored. Time spent tracking down and verifying a security alert is time spent not maintaining the system.

Most of the IT systems in use today support common protocols and communicate with each other with ease, whereas in ICS environments there are a number of vendor-specific proprietary protocols that do not readily communicate with other ICS protocols. Another consideration is the historical development environments for these components. Typically, the engineers developing ICS components do not have formal software engineering education or experience. This results in many of the common security pitfalls (such as buffer overflow attacks, lack of stack protections, etc.) to be widely prevalent.

Standards and regulations are an important part of modern automation systems. Well-defined standards allow for a larger selection of components, allowing for a more reliable and robust system. Regulations are imposed on asset owner/operators whose systems can pose a risk to the public's well-being. Examples of standards and regulations that include cybersecurity for ICS include NERC/FERC, CFATS, TSA Pipeline, and NRC.

CIA VS. AIC MODEL

Consider the CIA model in regards to ICS

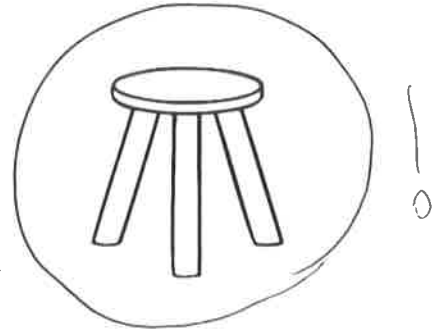
- Confidentiality
- Integrity
- Availability

Is availability king in ICS?

- When it comes to the process, availability is critical
- So, can we reprioritize CIA to AIC?

NO! In cybersecurity, all three legs are needed

- Availability of communications means nothing if the communications are corrupted or malicious (integrity)
- IT doesn't prioritize confidentiality, just listed in that order
- Avoid trying to universally prioritize any leg



Different organizations have different priorities in the CIA model. Many fall into the traditional order of C-I-A:

- **Confidentiality:** Prevention of information disclosure to unauthorized individuals or systems. TLS encrypts a user's web traffic, preventing unauthorized persons from reading the content should they somehow obtain access to the information (such as when using public Wi-Fi at a coffee shop).
- **Integrity:** Prevention of modification of data.
- **Availability:** Prevention of inaccessible systems. This involves ensuring the system is available and without service interruption due to failures or upgrades (or security systems designed to protect confidentiality or integrity). It is important to consider I/O and cable path diversity, redundant physical device location diversity, and in some implementations, a hardwired local alarm panel in case the operators lose access to the control system.

Availability is king?

Availability is the primary concern in a vast majority of ICS environments. Some ICS environments, however, would choose an availability impact if they identified a loss of integrity and an associated safety risk.

Class discussion: How can CIA elements be important under different cyber threat scenarios? For example, integrity is a critical element if you are facing a sophisticated threat actor that is trying to maliciously operate the ICS, or, confidentiality can be the most important element if you are safeguarding a prized recipe and a threat actor intends to steal it.

IT AND ICS DIFFERENCES IN SUPPORT

IT	ICS	Trend
Short life cycles; 3–5 years	Long life cycles; 10–15+ years for hardware	A move toward more software to future-proof solutions and allow for more rapid change
Maintenance and support options available	Maintenance and support typically come from supplier or certified parties	Static

Many IT environments are on a 3- to 5-year hardware and software refresh life cycle due to maintenance and support contracts, as well as system performance, storage improvements, and increased demands. In ICS environments, the systems could have a 10- to 15-year life cycle.

IT environments have seen outsourcing vendors, contractors, and many other forms of hosting and unique support contracts. In ICS environments, support personnel are typically dedicated internal employees, or support from the system supplier, or, in some cases, from a certified party.

ICS LIFE CYCLE

ICSs can have a life span of 10 to 15+ years

- Dependencies on other systems make upgrades challenging
- Takes effort to make equipment live long in harsh environments
- Large part of the system goes through a common life cycle

Factors can lead to new life cycles for sub-components

- Upgrades for improved reliability, optimization, and now security
- Process expansion and cost reduction
- Upgrade-resistant components include wiring, I/O, and hardware

Many new ICS designs are trying to future-proof

- Use hardware commoditization and virtualization
- Facilitates upgrades and changes
- Extensive testing such as FAT and SAT

A typical enterprise IT system will be upgraded, swapped, or otherwise decommissioned every few years. The life span of ICS equipment is, by comparison, over five times longer (sometimes upward of 15-30 years). Much of the same equipment that was created and designed before Metasploit was a thing or security firms were real businesses is still in service today. These systems weren't designed with security mechanisms like TLS in mind because TLS didn't exist at the time.

A second factor toward the extended life span of ICSs is the sheer cost of equipment refreshes. The price is high for designing a new system, configuring it, integrating the components in a test environment, refactoring it to work in production, and finally shifting the system over to the new system and working out bugs and kinks. This puts business pressure on an entity to maintain older equipment that may not have security components. After all, it ain't broke as far as the business case is concerned. It is estimated that over \$50 billion dollars of process automation has reached or surpassed the end of its useful life. The automation and control system supplier industry has been enjoying growth over the general economic slowdowns of the past several years.

Larger DCS/SCADA systems will typically begin with a majority of the components (applications, servers, network components, and hardware) at the same step in the life cycle. There are usually needs to integrate some legacy systems into new deployments. There are very few true greenfield deployments in existing facilities/assets. The cleanest greenfield occurs during the new build process. It is much more common to see some level of brownfield deployment during a modernization effort for the control system. Items that are less likely to be modernized (upgrade-resistant) include wiring, I/O, and hardware (this is where we still see 18- to 20-year life cycles).

The following items can introduce the need to upgrade a sub-element to a deployed SCADA or DCS:

- Operating systems that go past their supported life cycle
- Security guidance, reliability, and cost to maintain have been factors for upgrading parts of an ICS
- Applications that allow for greater optimization or features to improve efficiency or reliability of a system
- Upgrades that reduce the maintenance or operating cost of a process
- Expanding production lines or changing requirements (might require more capable controllers)

FAT AND SAT

Equipment refreshes are cost prohibitive

- Should specify security requirements
- Procurement language is important for managing risk

The Factory Acceptance Test (FAT)

- Performed by vendor
- Verify expected levels of functionality and security
- Could extend over several weeks or months

Site Acceptance Test (SAT)

- Owner/operator repeats a subset of a FAT
- After system installation and integrated functions

Equipment refreshes are cost prohibitive; therefore, procurement language is important when it comes to managing risk. DHS provides guidance on procurement specifications to mitigate the basic system specification issues. References are made to specific timing of deliverable information. All language is agreed upon pre-contract award; proprietary or business-sensitive information will be delivered after the contract is signed (post-contract award).

The vendor's Factory Acceptance Test (FAT) is necessary to verify that security features function properly and provide the expected levels of functionality. Each topic includes FAT tasks specific to that topic. In general, prior to initiation of each FAT, the Vendor shall install all operating systems and application patches, service packs, or other updates certified for use with the provided system by the time of the test and documentation of the configuration baseline. FAT is a process, not an event, and could, in fact, extend over several weeks or months.

The owner/operator Site Acceptance Test (SAT) typically repeats a subset of a FAT after system installation with additional integrated functions. Typically, the SAT is performed before the cutover or commissioning to validate that the site installation is equivalent to the system tested at the factory. Like the FAT, the SAT may extend several weeks or months and may occur at multiple locations.

Reference:

<https://goo.gl/C8nxtg>

VIRTUALIZATION IN ICS

Benefits of virtualization

- Consolidate control center and historian environments
- Create redundancy previously unavailable
- Ease patching and backup management
- Beneficial in testing, development, and training

Limitations

- Constrained by legacy physical connections
- Limited implementations in plant and remote field environments

Security precautions

- Do not architect across different trust zones (Business and Control)
- Ensure management is performed from trusted zone
- Consider the complexity added to troubleshooting and monitoring

In traditional IT data center environments, virtualization has primarily been driven by cost-reduction efforts that leverage a reduction in the data center footprint. This reduction will require less power, less cooling, and fewer physical assets. It will also ease management. In virtualization in ICS environments, there can be existing system dependencies or system interdependencies that require legacy physical connections that either exceed the physical capabilities of the virtual environment or are not supported by the system vendor. A key item to consider when evaluating a potential virtualization approach in an ICS environment is the possibility to overcomplicate the solution, beyond the ability of on-site staff or beyond the system's need and purpose. In some ICS cases, virtualization can create additional complexity in the environment that could impact troubleshooting and ongoing monitoring.

There are many ICS drivers and constraints that make virtualization appealing. These include:

- The need for system test environments that closely mirror the production environment. With virtualization, this can be achieved quite directly, as well as ease of maintaining synchronization and consistency within the test environment and in training environments.
- Routine system maintenance tasks such as patching, backups, restoration, backup validation, site DR planning, and possibly the performance of system upgrades.
- There is also the possibility for organizations to leverage the virtualization technology to achieve system-to-system failover capability that may not have been previously provided or supported by a system vendor or utilized as a means to move an application set over to 64-bit hardware without extensive application migration activity.

Various ICS industries have seen movement to virtualization in larger control center environments that typically contain mid- to large-scale data center environments, fully redundant control center requirements, and advanced network segmentation architectures with appropriate staff to implement and support these solutions. Within plant environments, ICS virtualization implementation efforts have been constrained by complexity concerns, legacy system restrictions, and lack of vendor support. In remote field environments, ICS virtualization pursuits have not typically been justified or implemented.



TAKEAWAYS AND RECOMMENDATIONS

Section takeaways

- ICS security levels are generally lower, life cycles longer, and there is less ability to change existing systems
- Focus security efforts into new projects
- Build defenses around networks of existing systems

Recommendations to owner/operators

- Architect safe methods, business process, and policy to allow minor changes without triggering full FAT/SAT tests

Recommendations to vendors

- Continue exploring methods to allow minor changes such as patches and system hardening to go forward without negatively affecting safety and reliability

This page intentionally left blank.

Course Roadmap

Day 1: ICS Overview

Day 2: Field Devices and Controllers

Day 3: Supervisory Systems

Day 4: Workstations and Servers

Day 5: ICS Security Governance

1. Introduction
2. GICSP Overview
3. ICS Concepts
 - Processes and Roles
 - Industries
 - **Exercise 1.1: Learning from Peers**
4. Purdue Levels 0 and 1
 - Controllers and Field Devices
 - Programming Controllers
 - **Exercise 1.2: Programming a PLC**
5. Purdue Levels 2 and 3
 - HMIs, Historians, Alarm Servers
 - Special Applications and Master Servers
 - Control Rooms and Plants
 - SCADA
 - **Exercise 1.3: Programming an HMI**
6. IT and ICS Differences
 - ICS Life Cycle Challenges
7. Physical and Cybersecurity
8. Secure ICS Network Architectures
 - ICS410 Reference Model
 - Design Example
 - **Exercise 1.4: Architecting a Secure DCS**

This page intentionally left blank.

Physical and Cybersecurity

When one is breached, odds are the other is as well. . .

This page intentionally left blank.

PHYSICAL SECURITY THREAT ACTORS

Disgruntled employees

- Hardest to detect and deter
- Have some level of access to assets
- Technical and procedural controls help

Thieves, espionage, and terrorism

- Different intentions but similar methods
- Perform reconnaissance of facilities and personnel
- Identify opportunities and methods for physical entry
- Attempt to evade detection and blend into the surroundings
- Look and act like authorized persons (social engineering)

Physical security threats to industrial control systems include two common threat actors and one specialized threat actor. Disgruntled employees and thieves are common threats to all organizations, and industrial espionage and terrorism are heightened risks for industrial control systems due to their proprietary configurations and ability to cause widespread damage.

Disgruntled employees are the hardest to detect and deter, as these attackers have some level of legitimate physical and electronic access to an organization's assets. It's highly recommended you tightly integrate Human Resources' employment termination procedures. Additionally, technical or procedural controls to deter piggybacking through secured doors should be established throughout all organizations.

Thieves and terrorists have different intentions behind their attacks on your facilities; however, their methods of penetrating your facilities will be very similar. These attackers will spend time performing reconnaissance of your facilities and personnel to identify opportune times and methods for entering your facility. They will attempt to evade detection or blend into the surroundings by looking and acting like an authorized person. They will attempt to social engineer their way into additional rooms by piggybacking through doors or asking for other assistance from authorized personnel.

Reference:

An FBI brochure that is available online and covers insider threat can be found at https://www.fbi.gov/file-repository/insider_threat_brochure.pdf/view

PHYSICAL SECURITY DEFENSES

Facilities containing ICS may need higher levels of physical security

- **Deter: Reduce the likelihood of attack**
 - Fences with razor wire, bright lighting
- **Delay: Increase the time it takes an attacker**
 - Locks, cement barricade mazes
- **Detect: Sense when a breach has occurred**
 - Tamper sensors, video cameras
 - ID card readers, biometrics
 - Log books, ID badges, visitor badges
- **Respond: Personnel and procedures to act**
 - Alarms, security guards at gates

Defenses should be tested and practiced

Physical security defenses must be taken into consideration for a facility that contains ICS components just like any other facility. However, some organizations feel their requirements for each category below may be far greater for facilities containing ICS:

- **Deterrence** is any security feature that reduces an attacker's likelihood of attempting an attack. In physical security, features that could deter an attacker include high fences with razor wire, visible cameras, bright lighting, and conspicuous security guards.
- **Delay** is the concept of layering defenses in a manner that requires an attacker to spend more time defeating the physical security controls.
- **Detection** is any security features that sense when a security breach may have occurred and identify the potential attacker. One of the easiest ways to detect and identify physical security threats could be security staff watching live CCTV or patrolling the facility looking for suspicious persons. Detection of physical threats may also be electronic in nature, with electronic access control systems that will sound alarms to a central security station if an attempt is made to gain access to a door that a person's access card doesn't have rights to, or if a secured door is forced open.
- **Response** to physical security threats is the last layer of defense in a physical security plan. Response includes any personnel and procedures that will be used to respond to a potential security breach of a facility. Ideally, a response is never required, but an organization must be capable of responding in an appropriate and timely manner to an attack.

Physical security controls can consist of technical and or procedural controls to achieve the most appropriate cost-effective solution for a specific location and access point. Many organizations determine the appropriate combination of technical and procedural controls based on site risk assessments; however, in some industries and at some locations, regulation dictates the appropriate physical security controls.

CYBER DILIGENCE FOR PHYSICAL INCIDENTS

Break-ins to control cabinets/houses are common

- Historically investigated as simple property crimes
- However, physical incidents may have a cyber dimension

Physical access can enable

- Access to a restricted network
- Circumvention of IT security defenses
- Collecting and stealing proprietary information
- Ability to plant unauthorized hardware or software

Many ICS components can be found in field locations, such as electrical substations or remote facilities in lockable control cabinets. Many remote locations experience unauthorized intrusions. These are not typically investigated with an eye toward cybersecurity if investigated at all.

Unauthorized entries at critical infrastructure locations have historically been viewed and investigated as simple property crimes only. This is particularly true for unmanned electrical power substations where an obvious trespass, theft, or vandalism is presumed the sole motive of the intruders. However, the growing number of sophisticated malicious actors and toolsets combined with increased use of conventional networks to remotely monitor and control unmanned sites from centralized facilities has also increased the possibility that traditional property crimes could use these sites to successfully mask less discernible cybercrimes.

In this new era of asymmetric threats, unauthorized entries could just as easily be a front, a cover crime intended to draw the asset owner's attention and investigation away from the real, more insidious motive. Perhaps the true purpose of the entry is to gain surreptitious access to systems and devices within the facility. Once inside, a malicious actor would be free to manipulate equipment, change settings, or plant unauthorized hardware or software.

PLANTING HARDWARE AND SOFTWARE

Thousands of hacking tools exist

- High-capacity keystroke loggers (keyloggers)
- USB keyboard emulators
- Drop boxes
- Other network surveillance products

Hardware is difficult to detect with untrained eye

- Traditionally was deploy-and-recover hardware
- Now options exist for Wi-Fi and cellular connectivity
- Have been used to transcend air gaps in many high-profile cases in both ICS and retail industries

There are literally thousands of hacking tools, high-capacity keystroke loggers, aka keyloggers, and other network surveillance products readily available on today's open market. Keyloggers are also standard features of many popular remote access Trojans (RATs). They can be hardware- or software-based, wired or wireless, extremely easy to use, and very difficult to detect.

Example: <http://www.h-online.com/security/news/item/Hacking-with-USB-keyboard-emulators-1172612.html>

Similar to RATs, malicious use of emulators could pose an even greater threat than keyloggers because they could be used to initiate keystrokes instead of merely recording them.

This is not a theoretical discussion. For example, advanced attacks are said to have been originally introduced via a USB thumb drive, demonstrating the effectiveness of gaining physical access to a system in order to conduct malicious activity, and high-profile incidents have proved that even logically separated or air-gapped networks are not immune to such attacks. Compromising remote facility communications or equipment is not a novel concept either. Security professionals often employ such tactics when conducting sanctioned network penetration tests for corporate clients.

HARDWARE HACKING TOOL EXAMPLES

USB, PS/2, AT
Keyloggers



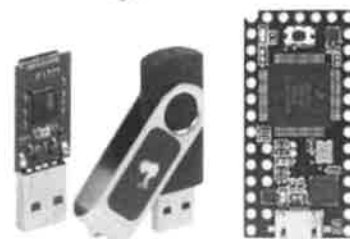
Serial and
Video Loggers



Embedded Keylogger



Keyboard Emulators



Some examples are provided for you here to get an idea of what they may look like and how easily they appear to be a normal part of system cords or normal attachments to the back of industrial computers.

Many of these tools are also useful for performing penetration tests and other cybersecurity assessments. Check out <https://hackerwarehouse.com> if you are interested in purchasing and learning to use some of these tools to test your defenses.

INDICATIONS OF INTRUSION

Physical indications of intrusion

- Unreconciled door and/or cabinet alarms
- Unauthorized personnel or vehicles on facility video camera
- Locks that can no longer be opened with the usual key
- Damage or signs of tampering to locks or barriers

Cyber indications of intrusion

- Inexplicable loss or behavior of communication links
- Unexplained connection between field components
- Inexplicable behavior of control system devices
- Missing or new unaccountable items
- Saturated communications (and large WAN bills)

Similar to social engineering, malicious cyber actors conducting blended attacks depend heavily on their target's daily routine, complacency, and natural tendency to not look beyond the obvious. Cybercrime is probably not a potential objective one would reasonably assign to an intruder, especially if there is any left-behind evidence of their purpose, such as metal theft or vandalism.

When a break-in has been discovered, it is normal to conduct some type of formal or informal investigation by the owner/operator according to their corporate policies and procedures. An inventory and damage assessment can be conducted (care should be taken as not to contaminate the scene and/or damage evidence in the event that local law enforcement is called in to conduct a formal criminal investigation). There are some typical indications of a physical intrusion or attempted breach.

ICS behavior or telecommunication link behavior may be an observable indication that a system is being used for other activity or that it may be infected or impacted. Some of the behaviors that warrant a closer look include loss or strange behavior of communication links, unexpected connections between field devices and other field devices, and strange control system behavior (to include actions not initiated by control room operators).

There have been incidents of malware infections on ICS components that racked up large communication link bills from providers, such as satellite and cellular services, as the infected device tried to propagate the infection.

CYBER INVESTIGATION

Consider evidence and how to further investigate

Report potential ICS incident/event to system owners and consider what information is provided to operators

Plan for minimizing impact to operations

Consider regulatory (if any) responsibility for reporting the incident

Levels of examination:

- Initial examination (walk-down)
- Cyber walk-down (visual inspection) and system and configuration checks
- Detailed examination (files and system binaries)

A systematic series of checks can be conducted to help determine whether or not sufficient evidence exists to indicate that a cybersecurity breach may have occurred in conjunction with the physical breach.

Escalating levels of examination based on increasing indicators of a possible cyber breach provide the decisional framework for balancing the need to gather system information against the risk or impact to operations in doing so. Conversely, one might also consider the risks to systems and operations in choosing not to conduct deeper evaluations that might have otherwise discovered the cyber breach.

Three levels of examination are considered: The initial examination, system and configuration checks, and if necessary, a detailed examination of filesystems and binaries. Each level of escalation is necessarily more invasive and requires greater expertise and closer coordination with facility or company personnel responsible for corporate cybersecurity and production operations. Thus, management needs to be able to assess and balance the risks of escalation with the potential for outages and equipment and personnel safety caused by a cybersecurity breach of a critical asset. As the cyber investigation escalates, it is likely that any significant tests on control components may require specific test plans, procedures, and perhaps even scheduled outages to obtain necessary information.

Remember, you start with a physical walk-down. Before cyber assets themselves are examined, an individual familiar with the site's physical configuration should first examine it for anything that appears missing or out of place (this practice is usually referred to as a "walk-down"). A staff member who is familiar with the facility's control devices should also be present and available to assist in the examination. It is important to log all observations. Photographic or digital video records of the scene can also be of great value because observations deemed insignificant during the initial discovery can sometimes prove otherwise later in the investigation. Again, care must be taken not to disturb evidence should the asset owner elect to involve the local law enforcement agency.

SITE WALK-DOWN PROCESS

Initial Walkthrough

- Look for signs of entry into the building or cabinets.
- Look for evidence of tampering with cyber assets.
- Inspect physical connections to cyber assets.
- Check for missing, unfamiliar, or new hardware.
- Scan for suspicious or unauthorized RF emanations.

System and Configuration Check

- Log in to each cyber device and check the log history.
- Check each device for uptime.
- Look for new user accounts or other system changes.
- Check for new removable media.
- Verify the configuration of each device.

File and Binary Examination

- Check the timestamps of configuration files.
- Check the firmware of any upgradeable devices.
- Scan applicable cyber devices for viruses and malware.
- Obtain trusted configurations and restore suspicious devices.

Those investigating a physical security breach should at least consider the possibility that a cyber-related incident may also have occurred. As a seasoned employee familiar with the site, pause for a moment and consider the location from a malicious actor's perspective. How would you do it if you wanted to compromise the local equipment or gain surreptitious upstream network access? Then inspect the site with that perspective in mind. One must consider simply manipulating local ICS equipment. Recognize the risks posed by unauthorized changes to a control system's configuration or software and the potential to significantly impact equipment, operations, and overall reliability.

Your organization may have a specific definition for a cyber incident or breach. A *cybersecurity breach* is considered as unauthorized access and/or changes to a control system, unauthorized hardware or software placed onto a control system or network, an unauthorized physical connection to a control system network, etc.

The walk-down process noted above is based on a paper written by Michael Assante and Scott Swartz that was published by the Department of Homeland Security and subsequent updates based on real-world events. Updated elements of this paper can be found in the SANS research white paper, *Industrial Control Systems (ICS) Cybersecurity Response to Physical Breaches of Unmanned Critical Infrastructure Sites*, which can be found at <http://ics.sans.org/ics-library>.

ACTUAL EVENT EXAMPLES

Poland Tram Incident

- 12-year-old boy used programmable TV remote to control switch gear
- Inexplicable switching activity treated as unexplained behavior
- After collision, rigorous investigation revealed the boy's activity

PG&E Metcalf 500kv Substation (Spring 2013)

- Individuals entered manholes and cut fiber cables, disabling communications
- 100+ high-powered rifle rounds at 10 transformers and 3 transformer banks
- Cooling oil leaked, causing transformers to shut down
- No major power outages, injuries, or long-term damage

Entergy Keo 500kv Substation Attack (Fall 2013)

- Four different attacks on Entergy systems
- Events included setting the control house on fire
- Eventually caused a blackout affecting 10,000 people
- Message on control panels: "YOU SHOULD HAVE EXPECTED U.S."

Authorities investigated for cyber threats on last two

There have been ICS cases of inexplicable system behavior that pointed to local activity as a potential cause for the behavior.

A classic ICS example is the Poland tram incident. A 12-year-old boy had used a programmable TV remote to control switch gear. The inexplicable switching activity was treated as unexplained behavior and simply switched back when noticed by the tram control center operators. A more rigorous investigation might have uncovered that same boy in proximity with the manipulated switches every time. A more thorough investigation occurring after a collision revealed the boy's activity.

Other stories like this can be found at:

https://en.wikipedia.org/wiki/Metcalf_sniper_attack

<http://www.forbes.com/sites/williampentland/2013/10/07/weekend-attacks-on-arkansas-electric-grid-leave-10000-without-power-you-should-have-expected-u-s/>

<http://edition.cnn.com/2013/10/08/us/arkansas-grid-attacks/>

Other resources include:

- Requests for assistance from the ICS-CERT can be made by contacting them directly via telephone at 877-776-7585 or by sending an email to ics-cert@dhs.gov
- Information about the ICS-CERT can be found on their website at <https://ics-cert.us-cert.gov>

TAKEAWAYS AND RECOMMENDATIONS

Section takeaways

- There needs to be a relationship between physical-security and cybersecurity teams
- Physical breaches are often trivial, but more risk to the attacker
- Identify the types of sites you have and physical defenses that exist

Recommendations to owner/operators

- Build business processes with your physical security peers before you have a physical breach

Recommendations to vendors

- Consider tamper protections on your products, especially at Level 0 and 1
- Provide physical and cyber methods to identify tampering

This page intentionally left blank.

Course Roadmap

Day 1: ICS Overview

Day 2: Field Devices and Controllers

Day 3: Supervisory Systems

Day 4: Workstations and Servers

Day 5: ICS Security Governance

1. Introduction
2. GICSP Overview
3. ICS Concepts
 - Processes and Roles
 - Industries
 - **Exercise 1.1: Learning from Peers**
4. Purdue Levels 0 and 1
 - Controllers and Field Devices
 - Programming Controllers
 - **Exercise 1.2: Programming a PLC**
5. Purdue Levels 2 and 3
 - HMIs, Historians, Alarm Servers
 - Special Applications and Master Servers
 - Control Rooms and Plants
 - SCADA
 - **Exercise 1.3: Programming an HMI**
6. IT and ICS Differences
 - ICS Life Cycle Challenges
7. Physical and Cybersecurity
8. Secure ICS Network Architectures
 - ICS410 Reference Model
 - Design Example
 - **Exercise 1.4: Architecting a Secure DCS**

This page intentionally left blank.

Secure ICS Network Architectures

When you can't touch the endpoints, that only leaves the network . . .

This page intentionally left blank.

SECURE ICS NETWORK ARCHITECTURES

A secure network architecture is your first layer of defense

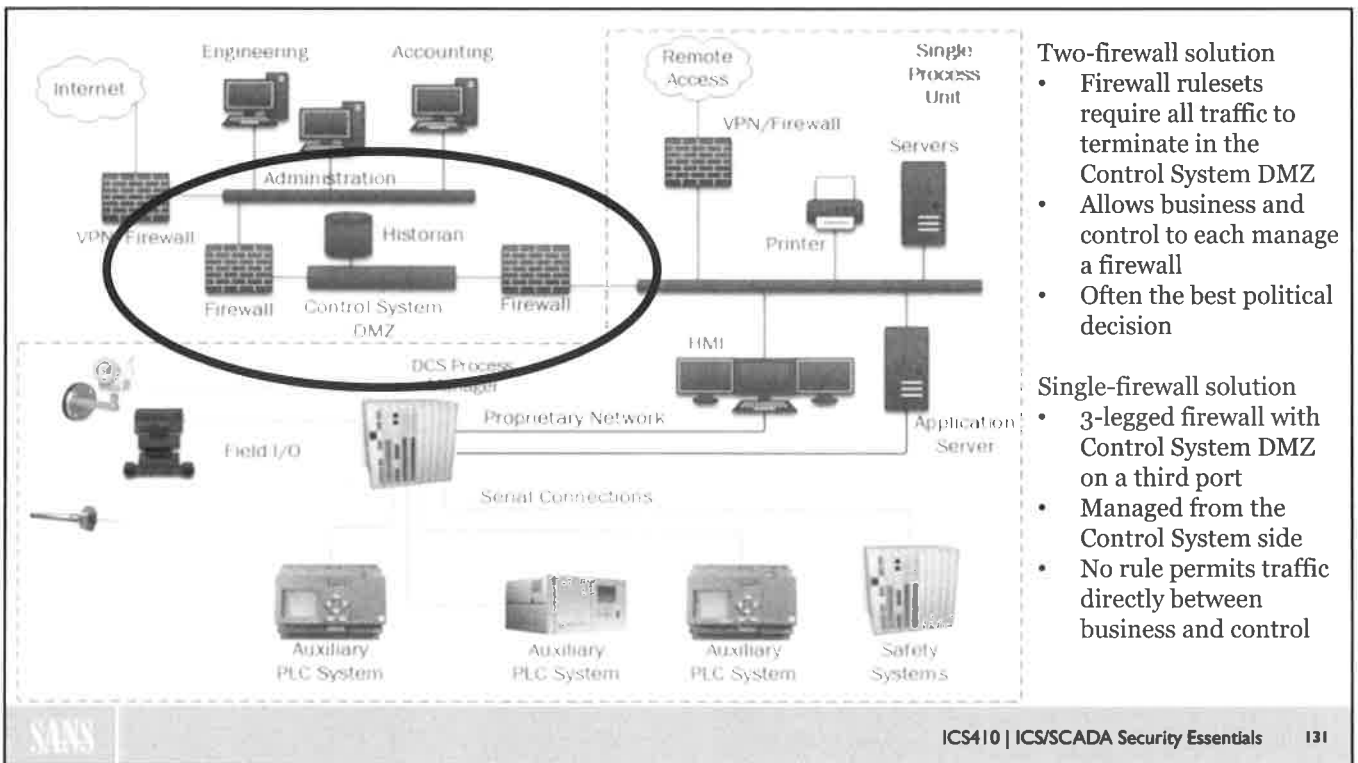
A strong perimeter between business and control networks is essential, but not enough

- Thus, the old adage: Hard and crunchy on the outside, soft and chewy in the middle
- We need to build additional sub-perimeters inside our control networks

Consider the firewall plus DMZ perimeter in the following diagram

When it comes to industrial control systems in today's age, having a defensible network is essential. This should start with properly segmenting your network and building a strong perimeter. In reality, a secure network architecture is your first layer of defense. Many organizations do this by dividing their systems into two networks, one for business and one for control. This makes sense because our control systems pose a higher risk to the organization if compromised. By dividing the network into two, organizations can build a security perimeter around the control network, limiting what attackers can get to if they compromise a normal business workstation. It also provides a point at which we can place special defenses to protect and monitor the traffic going into and leaving the control network.

A strong perimeter between business and control networks is essential, but it is not enough. Most of us have heard the old adage, "Hard and crunchy on the outside, soft and chewy in the middle," relating network security to a piece of candy. One perimeter is a good start; however, we need additional sub-perimeters inside our control networks. This allows us to create additional layers where we can create traffic controls, increase our visibility for monitoring, and provide network segments that can be contained in times of compromise.



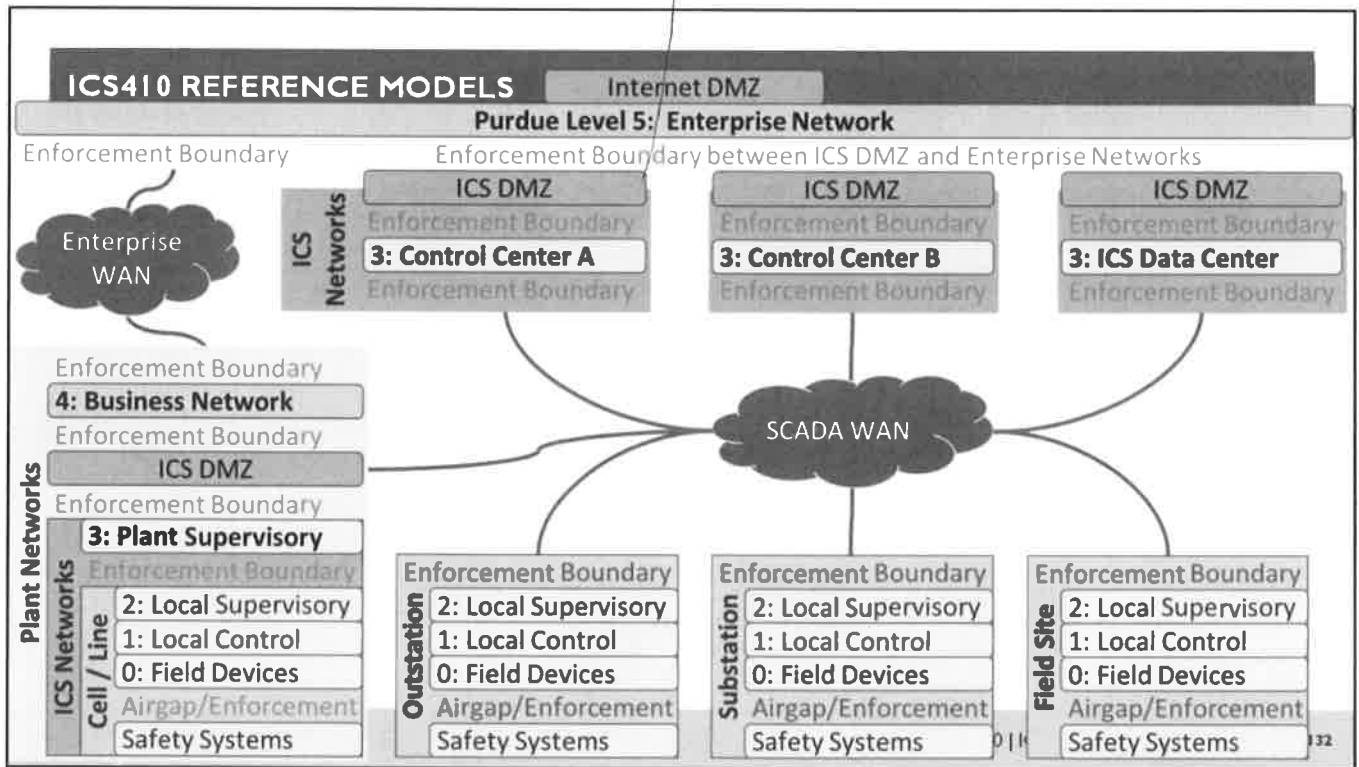
- Two-firewall solution**
- Firewall rulesets require all traffic to terminate in the Control System DMZ
 - Allows business and control to each manage a firewall
 - Often the best political decision

- Single-firewall solution**
- 3-legged firewall with Control System DMZ on a third port
 - Managed from the Control System side
 - No rule permits traffic directly between business and control

This is an example DCS reference architecture that demonstrates device placement within network segments and communications typically in use at the various layers of the system architecture.

This diagram shows two firewalls surrounding a DMZ. This assumes that each firewall’s ruleset does permit any traffic to pass from the business network to the control network without a stop at a system in the Control System DMZ. This same solution could be performed with a single firewall with the Control System DMZ off to the side of the firewall on a third port. This is debatably just as secure as long as the single-firewall solution is managed entirely from the Control System side, the more-secure and less-likely-to-be-compromised side. However, this management requirement usually is not acceptable to the enterprise security teams; thus, the biggest reason two firewalls are often used, allowing each side, business and control, to manage their own firewall.

Image Source: NIST SP800-82
<https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-82r2.pdf>



A digital copy of these models can be found on your student USB in the Posters folder

- ICS410 SCADA Reference Model.png
- ICS410 Plant Reference Model.png

Other posters that include previous versions of the ICS410 reference model are also on the USB

PURDUE LEVELS AND ENFORCEMENT BOUNDARIES

Networks should be organized by Purdue levels

- ICS410 reference model is based on guidance in ISA/IEC 62443
- Each level has different components, services, and functions
- A single Purdue level can contain multiple subnets
- Network defenses can be placed between subnets in the same Purdue level

Enforcement boundaries are where we place cybersecurity network defenses

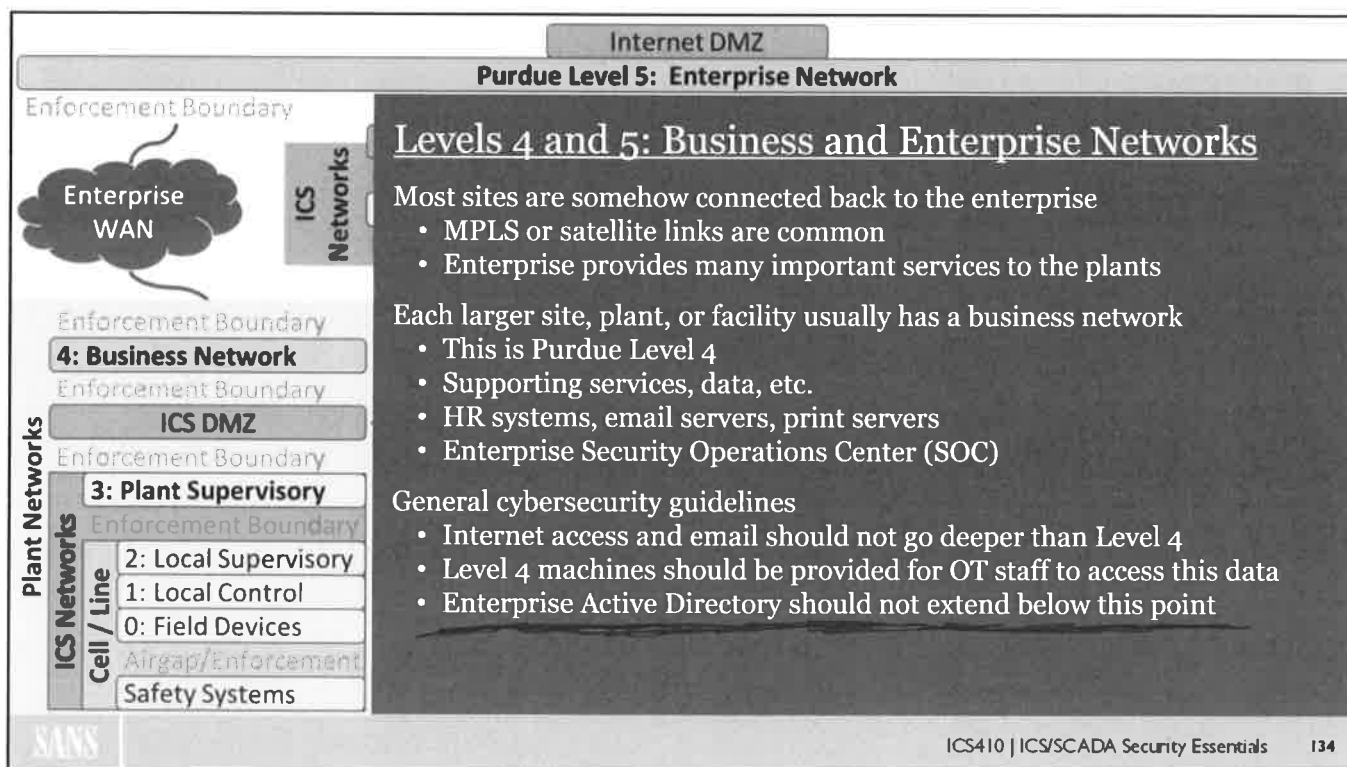
- **Limit communications through the boundary**
 - Network firewalls and next-gen firewalls
 - Data diodes and unidirectional gateways
- **Monitor and record all communications through the boundary**
 - Network Intrusion Detection Systems (NIDS)
 - Network Behavior Anomaly Detection (NBAD)

These will be discussed in detail later in the course

Purdue Levels

Levels reflect the hierarchical organization of industrial control systems and the networks they interconnect to. The "level" model begins closest to the "plant floor" or "field" with Level 0, Process Control Instrumentation Bus Network, which builds up to Level 1, Control Devices; Level 2, Supervisory Control Local Area Network; Level 3, Operations Support DMZ; and further up to Level 4 and 5 plant and business networks.

Enforcement Zones: Includes the functions necessary to segment and protect the various zones within an ICS environment. Items typically found in this zone include firewalls, routers (with ACLs), application firewalls, Data Guard technology, and unidirectional data diode technology. Technologies implemented may differ at the various enforcement zones within an ICS environment depending on the business needs and the level of risk determined by a specific enforcement zone.



Purdue Level 5: Enterprise Business Network

Contains corporate-level applications used to support enterprise businesses and user goals. Items typically found in this zone include internet access points, email servers, customer-facing web servers, internal web servers, CRM systems, HR systems, corporate directory architectures, enterprise document management systems, and remote access VPN endpoints.

Purdue Level 4: Business Network at the Plant

Contains IT shared services for a local site, plant, business unit, or subsidiaries. Items typically found in this zone include local file and print servers, local phone systems, site directory replicas, site-specific remote access solutions, security event aggregators, and site-specific internet access points. This is the network where all workstations that have access to enterprise email and internet access should be placed. All OT employees should have access to workstations, or be assigned second workstations in this network, for them to perform their normal enterprise needs.

Purdue Level 4: Plant's Local Business Network

Enforcement between ICS DMZ and Business Networks (Business pulls from or pushes to ICS DMZ)

ICS DMZ - Level 3 to 4 ICS DMZ - Level 4 to 3 ICS DMZ - Cloud Access ICS DMZ - Remote Access

Enforcement between Control Networks and ICS DMZ (Control pulls from or pushes to ICS DMZ)

Purdue Level 3: Master Servers, Workstations, Testing/Staging, Cybersecurity, Jump Hosts

ICS DMZ (This is not your internet DMZ!!!)

ICS DMZ and its enforcement boundaries are your primary ICS perimeter

- It is the sanitization room/procedures for all data between control and business zones
- All communications should stop at a system in this DMZ
- No communication should bypass the DMZ
- Simple proxies should be avoided since they blindly pass on exploits
- Don't place patch management servers here, instead place them in Purdue Level 3
- ICS Remote Access DMZ is a VPN endpoint, RDP/VNC/Citrix to jump host in Purdue Level 3

The most secure communications should look like this

- Level 4/5 pushes to this DMZ, and Level 3 pulls from it
- Level 3 pushes to this DMZ, and Level 4/5 pulls from it
- Larger sites can use multiple DMZ to further separate traffic and mitigate risk
- Smaller sites can use a single ICS DMZ

SANS

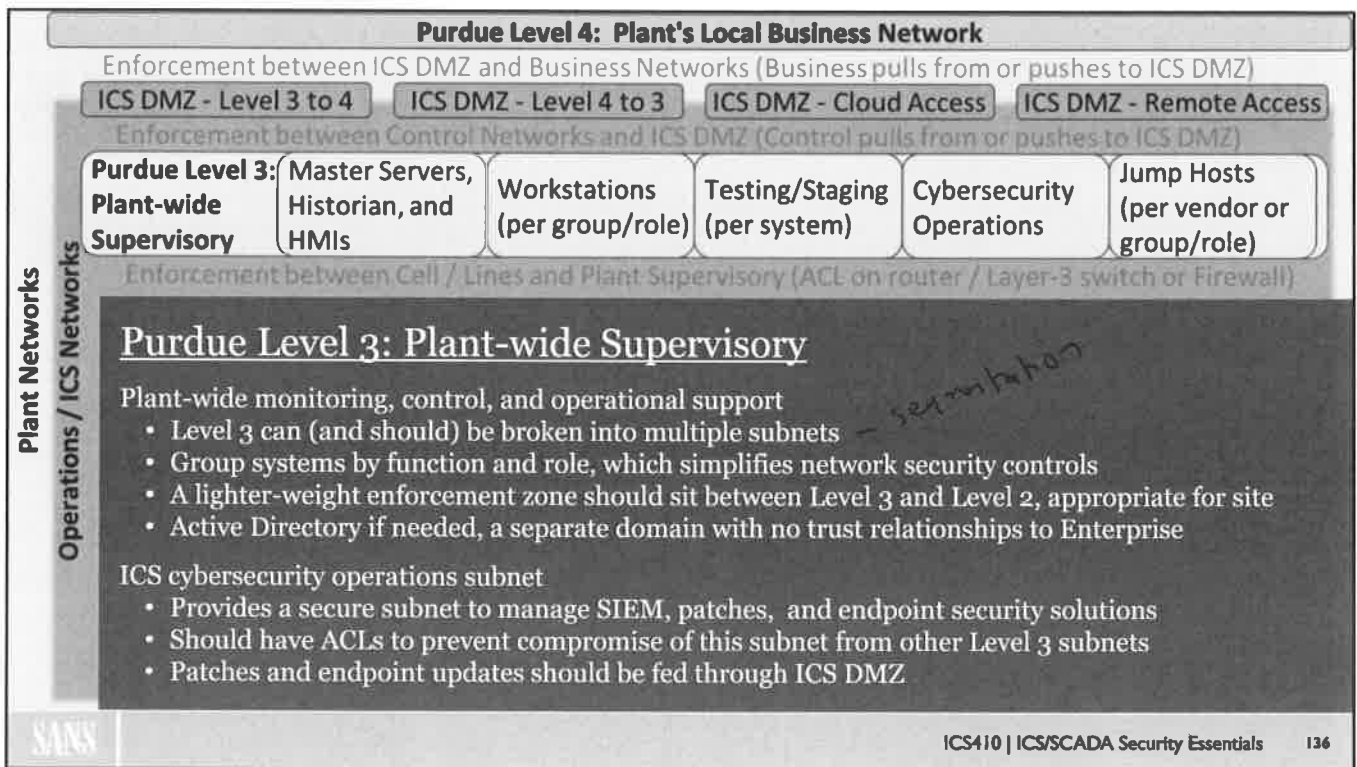
ICS410 | ICS/SCADA Security Essentials 135

The ICS DMZ, or as many ICS DMZs as you create, serve one single purpose: It creates a secure stop for all traffic being shared between the Enterprise and the ICS networks, allowing us to do cybersecurity checks, monitor for attacks, and create records of all transactions to aid us in incident response and network forensics. By forcing all traffic to stop at a system in the ICS DMZ, we greatly hamper the attacker ability to create bidirectional backdoor tunnels into the ICS networks.

Think of the ICS DMZ in the same light as the physical sanitization room and associated routines that all employees must pass through before entering or leaving the manufacturing area in the plant. The ICS DMZ provides the same type of sanitization room and procedures for all data entering or leaving the plant.

Throughout the course, we will make several more detailed recommendations of what to place in this ICS DMZ, but for now, here are some high-level guidelines. Most remote connectivity for OT staff and vendors should require a two-step process, VPN to the ICS DMZ, then create a second connection using a hardened RDP/VNC/Citrix to a jump host in Purdue Level 3, using internal ICS credentials, and prohibiting them from using any software or data on their remote workstations. All software and data needed should be pre-installed on the jump hosts. Secure methods to pass required data through the same malware-scanning servers used to pass other data from enterprise to ICS networks can be set up for use cases where certain individuals need to transfer data to the jump host for use.

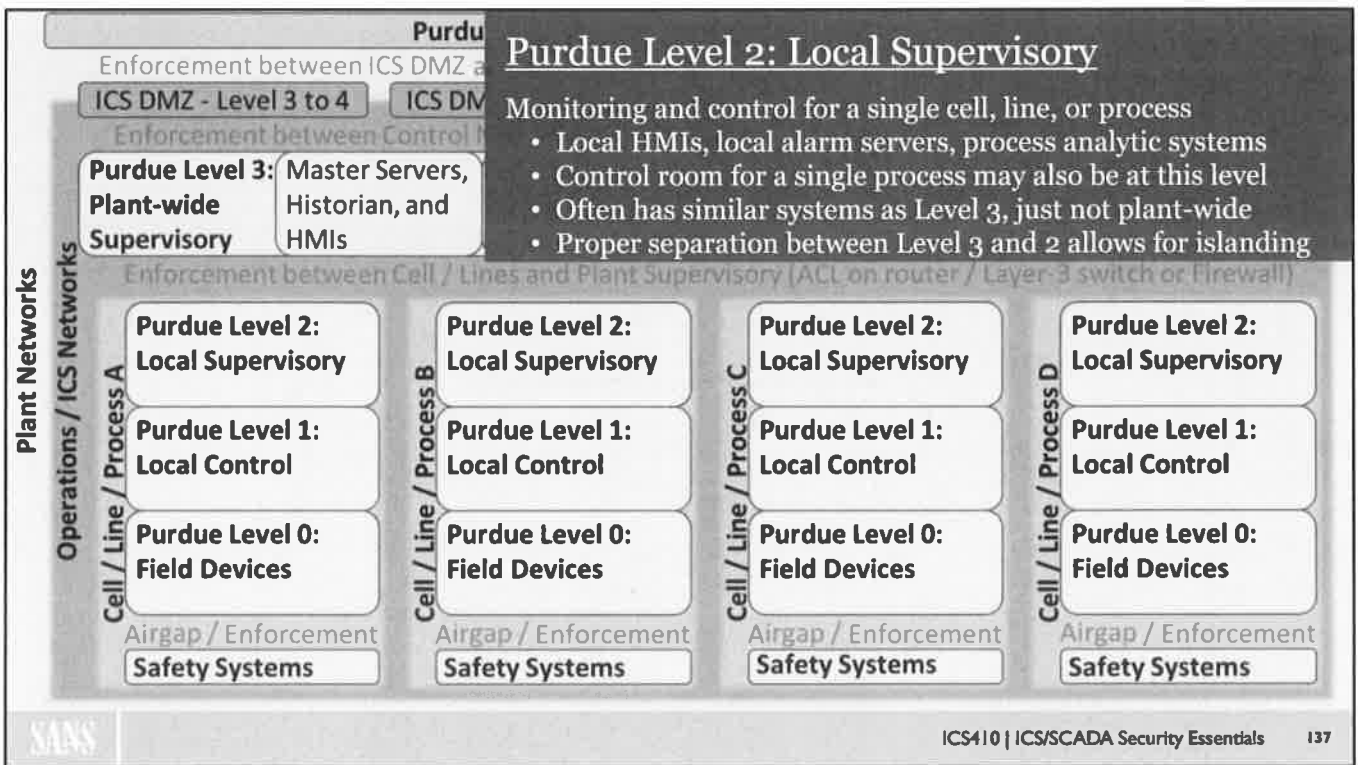
Another dated and ill-advised recommend that is often still repeated is to place your cybersecurity operations systems like patch management into your ICS DMZ. Don't do it. The next page will address this issue and the why.



Systems in this level include functions involved in managing the operations environment across the whole plant or manufacturing site. Items typically found in this zone include master servers managing operations for the whole site, plant-wide SCADA, plant alarm servers, aggregate plant historians for the site that can be replicated to read-only historians in the DMZ, operations scheduling resources, reliability tracking tools, operations simulation and modeling tools, contingency analysis tools, and data visualization utilities. There may also be dedicated operations-specific IT services such as DHCP, LDAP, DNS, and file servers.

Management of all operations/ICS network systems, including network equipment, Active Directory, and virtualization servers should be deployed separate from enterprise solutions, and entirely managed from within the operational/ICS networks. So regardless of the teams supporting these solutions, they should be required to use your standard remote access solution to establish connectivity into the ICS network before being able to manage their solutions.

A separate cybersecurity operations subnet should contain all cybersecurity operational systems such as SIEM solutions, patch management servers and other host-based protection software servers. This provides a safe location for this sensitive data, and prevents it from leaving the enhanced security of your ICS networks. Many older guidance documents have recommended placing these types of services into the ICS DMZ, however this creates an avenue of attack, as many of these cybersecurity solutions require connectivity to highly sensitive ICSs, often with the ability to change or modify them. Trying to place them in your ICS DMZ forces horribly insecure firewall rulesets to be created, allowing these servers almost unfettered access to ICSs from the ICS DMZ. Remember, the purpose of any DMZ is to securely share data with external networks, not to allow external networks to management your sensitive ICS. A further parallel, do IT organizations place patch management servers in their internet DMZ? No, they don't. That would only make sense if you were trying to allow the public internet the ability to manage patches on your internal servers.



Includes the functions involved with operating the real-time control system. Items typically found in this zone include: Control room operation workstations, local Human Machine Interfaces (HMI), engineering workstations, security event collectors, operations alarm systems, communications frontends, data historians, and network / application administrator workstations and engineering workstations specific to a particular manufacturing cell, process, or line.

Don't get confused, as many of the systems in Level 2 are similar to the systems that exist in Purdue Level 3. The biggest difference is the level of scope for the system. In a perfect world, each manufacturing cell, line, or process should be able to operate for a limited time without communications to Level 3, allowing us to island each process in the case of Level 3 compromise.

Purdue Level 1: Local Control

Controllers for the local cell, line, process

- DCS controllers, PLCs, RTU, etc...

Purdue Level 0: Field Devices

Sensors and actuators for the cell, line, process

- Basic sensors and actuators
- Smart sensors and smart actuators
- Other field instrumentation
- Fieldbus networks communication gateways

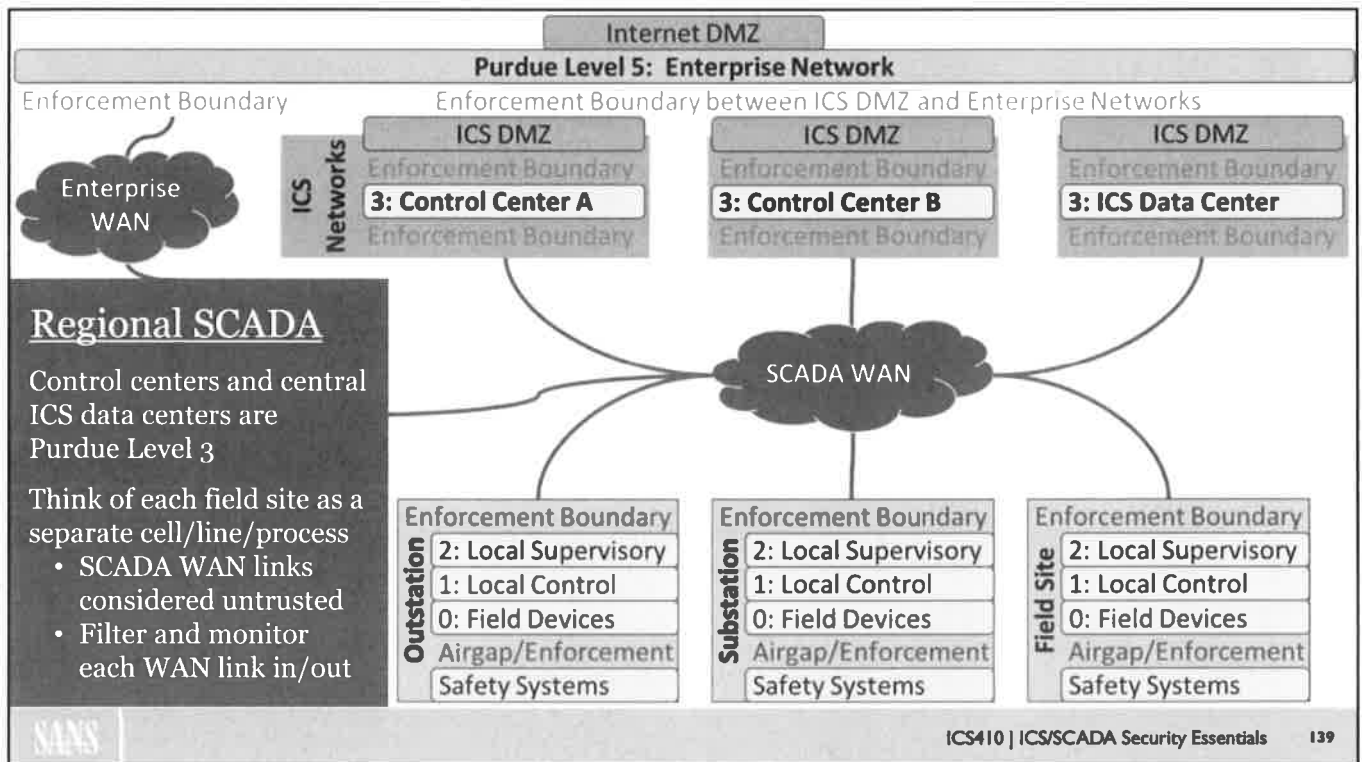


Purdue Level 1: Control Devices

Includes the functions involved at site-specific operating environments. Items typically found in this zone include: Dedicated operator workstation, Programmable Logic Controllers, control processors, programmable relays, remote terminal units, and process-specific microcontrollers.

Purdue Level 0: Field Devices

Includes the functions involved in transitioning from cyber to physical and from physical to cyber. Items typically found in this zone include: Sensors, actuators, motors, process-specific automation machinery and field instrumentation devices.



When ICSs span multiple remote sites and are managed by a regional SCADA solution, use of the Purdue model can become more confusing. This has led to a number of inconsistent uses of the Purdue model in SCADA between ICS industries, and even within single industries. Hopefully the reference model above will help you and everyone taking this class come to a more consistent use of the Purdue model for regional SCADA.

Remember, in a plant, Purdue level 3 contains systems that provide plant-wide supervise and control. Regional SCADA is the same, but instead of controlling systems plant-wide, they control systems region-wide. Using this simple guidance, all ICS control centers and central ICS data centers should be classified as Purdue Level 3.

As for remote sites, outstations, and substations, think of them as their own separate cell/line/process. They will each have their own Purdue level 0, 1, and 2. They will leverage the SCADA WAN to communicate back to the ICS control centers, which means you should be leveraging VPN technologies to safeguard that supervisory and control traffic from being intercepted by attackers as it passes through the WAN.

Remember, WAN connectivity, both physical cabling or wireless technology, is trivially accessible to the determined threat actor. Furthermore, all data going through the WAN is usually accessible from the WAN service provider's networks, so threat actors or malicious insiders at the WAN service provider have the ability to analyze, change, and even spoof your supervisory and control signals if you are not protecting it with cryptographic protections fully in your control, such as VPNs in your enforcement zones adjacent to the WAN. Other cybersecurity solutions that provide traffic filtering and monitoring can also aid you in the protection of your systems and detection of attackers.

TAKEAWAYS AND RECOMMENDATIONS

Section takeaways

- Digital copies of the ICS410 Reference Models are on your student USB in the Posters folder
- Keep the internet and elements of it like email out of control networks
- When you define an enforcement zone, use BIG magnifying glasses
 - Limit to only required traffic
 - Collect and keep a record of all traffic through it
 - Don't let virtualization or Active Directory through it
- Remember, each level can and should be broken into subnets with flow control between them

Recommendations to owner/operators

- If you aren't using the Purdue levels, start today
 - Start with enforcement zone and DMZ between business and control
 - Then start separating Level 2 and Level 3

Recommendations to vendors

- Continue building products that fit in this layered model
- Create guides for customers on how to do this with your products

This page intentionally left blank.

Course Roadmap

Day 1: ICS Overview

Day 2: Field Devices and Controllers

Day 3: Supervisory Systems

Day 4: Workstations and Servers

Day 5: ICS Security Governance

1. Introduction
2. GICSP Overview
3. ICS Concepts
 - Processes and Roles
 - Industries
 - **Exercise 1.1: Learning from Peers**
4. Purdue Levels 0 and 1
 - Controllers and Field Devices
 - Programming Controllers
 - **Exercise 1.2: Programming a PLC**
5. Purdue Levels 2 and 3
 - HMIs, Historians, Alarm Servers
 - Special Applications and Master Servers
 - Control Rooms and Plants
 - SCADA
 - **Exercise 1.3: Programming an HMI**
6. IT and ICS Differences
 - ICS Life Cycle Challenges
7. Physical and Cybersecurity
8. Secure ICS Network Architectures
 - ICS410 Reference Model
 - Design Example
 - **Exercise 1.4: Architecting a Secure DCS**

This page intentionally left blank.

EXERCISE 1.4: ARCHITECTING A SECURE DCS

DURATION TIME: 15 MINUTES

We are going to provide you with a DCS architecture that contains a number of poorly designed network decisions from a cybersecurity perspective based on the same business requirements we just considered.

Compare it to the last few lecture slides we just went through and answer some questions about it.

OBJECTIVES

Though designing and architecting an environment are always fun, you frequently need to understand, analyze, integrate with, or troubleshoot an existing environment. This exercise will try to do just that while also reinforcing the design principles we just covered during the lecture.

PREPARATION

Introduce yourself to your neighbor and ask them to join you for this lab.

Open the file “Day 1 – Poorly Designed DCS.pdf,” located on the ICS410 Student USB.

This page intentionally left blank.

EXERCISE 1.4: ARCHITECTING A SECURE DCS

ANALYSIS OF SAMPLE DIAGRAM

Take 5–10 minutes to answer the following questions with your neighbors:

- What operational risks does this configuration pose?
- What cybersecurity risks does this configuration pose?
- Can you think of reasons why the architect made some of these poor security design decisions?

This page intentionally left blank.

EXERCISE 1.4: ARCHITECTING A SECURE DCS

CLASSROOM DISCUSSION

How many items of concern did you find?

- What operational risks does this configuration pose?
- What cybersecurity risks does this configuration pose?
- Can you think of reasons why the architect made some of these poor security design decisions?

Student doing SelfStudy or OnDemand can find the classroom discussion points below

Discussion items include:

- **Directory Server trust relationship that bridges corporate and supervisory control network segments:** May have been configured to allow user credential federation; allows numerous vulnerabilities and threats to carry over into the process control network.
- **Engineering workstation and read-only historian contained in the business network:** May have been placed in this zone due to the availability of network communication; assets in a lower-trust zone with necessary bidirectional and administrative rights into the process control network.
- **Remote support directly into a mobile PC in the supervisory control layer:** May have been provided to allow temporary access to a remote worker to troubleshoot or tune an environment; lacks strong remote access authentication and VPN, and the terminating device can move from network to network.
- **Alarm servers separated out into a supervisory control support network:** May have been placed in this zone due to the availability of network communication and to allow a view to other users; if there is a network communication issue, the alarm servers will lose their ability to update and provide a view.
- **AV deployment servers and DNS contained in the supervisory control layer with access out to higher-layer networks:** May have been placed in this zone to allow communication to agents and possibly configured this way for system FAT with the intention to change when delivered for SAT; could be used to exfiltrate data or bring exploits into the segment.
- **Local HMI in the business network:** May have been placed in this zone due to the availability of network communication; an asset with explicit rights to impact the process control environment should be contained in the process control network.
- **IPS inline between the supervisory control layer and the process control layer:** May have placed an IPS device with the intent of operating as an IDS or intentionally placed with the intent of providing real-time protection of the critical assets; directly impacts communication and the overall process.
- **Single controller in the Process A:** There may be a failed or faulty controller that was not replaced, or perhaps the redundant controller was never wired in; this will impact the overall process reliability and availability requirements.

EXERCISE 1.4: ARCHITECTING A SECURE DCS

TAKEAWAYS AND RECOMMENDATIONS

Section takeaways

- We can rarely force everything into our perfect security architectural model
 - Sometimes we can't change existing networks
 - Sometimes vendor's products or support contracts don't support it
 - Do what you can, and evaluate what the real risks are when you can't
- Remember, someone else's design might be missing something we see
 - Or sometimes we are the one that is missing what they already saw . . .
 - Be slow to criticize but willing to ask the hard questions

This page intentionally left blank.

COURSE RESOURCES AND CONTACT INFORMATION



AUTHOR CONTACT

Justin Searle – justin@controlthings.io



SANS INSTITUTE

11200 Rockville Pike, Suite 200
N. Bethesda, MD 20852
301.654.SANS(7267)



ICS RESOURCES

ics.sans.org
Twitter: @sansics
SANS ICS Community
<https://ics-community.sans.org/signup>



SANS EMAIL

GENERAL INQUIRIES: info@sans.org
REGISTRATION: registration@sans.org
TUITION: tuition@sans.org
PRESS/PR: press@sans.org

This page intentionally left blank.