Legacy Industrial Control Systems - Secure / Replace / Ignore?

2017 NSF Cybersecurity Summit for Large Facilities and Cyberinfrastructure

Phil Salkie, Jenariah Industrial Automation - phil@jenariah.com - 301-859-0500

Reading Period

Does the title of this class remind you of the days of MS-DOS? Did you think those days were long behind you (or that they'd passed before you even paid attention?) Well, it's entirely possible that mission critical infrastructure in your facility has been running on control systems which were programmed with MS-DOS or Windows XP, and maintaining or replacing them will require old operating systems, old software and old computers which will run them (or, in some cases, modern hardware running emulations of old computers.)

Fun Fact: There were programmable controllers before there were desktop computers. ("PC" meant "Programmable Controller" before it meant "Personal Computer", then was changed to "PLC" for "Programmable Logic Controller" after the IBM-PC came out.)

Not-So-Fun Fact: Some of those controllers are still out there and running.
They may have been programmed on dedicated hand-helds, with their programs stored on cassette tape, or hand-written on paper with pre-printed boxes for entering the data. These paper documents may be the only existing copy of the operating logic.

Fun Fact: There were PLCs before there were communications network standards.

Not-So-Fun Fact: Some of those controllers are still out there, running on dozens of different networking hardware layers and hundreds of software communications protocols. Names you may run into are: Modbus, CANBus, ProfiBus, FieldBus, InterBUS, CC-Link, DeviceNET, HART, CIP, Ethernet/IP, DF-1, DH-485, MELSECNet, BACnet, LON, ZigBee, SRTP, and many, many more.

Some of these protocols allow an external host to read and write anything stored in the controller's data memory, some allow an external host to read and write the controller's password or the logic program, some allow an external host to flash new firmware into the controller's operating system memory. Some protocols have undocumented functions and features which could be exploited.

Fun Fact: Desktop PC systems are designed with a four year product lifecycle.
One often sees PCs which are ten years old, and occasionally encounters a system that's over 20 years old. (Win98 SE, anyone?)

Not-So-Fun Fact: Controls systems are designed to be installed in capital equipment hardware which is amortized over twenty or thirty years. It is common to see controllers which were installed thirty or forty years ago, and one can occasionally see operating machinery with controls built before integrated circuits were common.

Fun Fact: The advent of the Desktop PC forced standardization of communications systems, including connectors, hardware layers, and higher level protocols.

Not-So-Fun Fact: Controls systems have used a bewildering variety of connectors, pinouts, standards, and protocols - often making changes to existing designs seemingly solely to break compatibility with existing systems. You may require specialized cables, programming interface boxes, or plug-in cards for your computer - you may even find that modern computing hardware is incapable of communicating with a given system, and you'll need to find a ten-year-old laptop with a built-in RS-232 port or a PCMCIA card slot.

Fun Fact: Modern PC motherboards support PCIe 2.0 X1, PCIe 2.0 X16, PCIe, and (possibly) PCI busses. They also support USB 2.0, USB 3.0, and (possibly) USB 3.1, USB-C, Lightning, and Firewire.

Not-So-Fun Fact: Control systems have used S-100, VME, STDbus (8,16, and 32), ISA-8, ISA-16, PC/104, PC/104-16, PCI, PCIe, and a host of other proprietary busses either based on existing standards of the time or developed entirely in isolation.
Programmable Controllers generally do not use standardized busses, but have proprietary data paths and connector sets. Dozens of bus architectures are currently in use in controls systems worldwide today.
If Desktop PCs are part of a control project, it is more than possible that they are using a bus architecture which is not available on new motherboards, and are using plug-in cards which cannot be purchased new (and may even be difficult to get repaired.)

Fun Fact: Modern computer systems use flash memory for long-term storage of data which is read more than it is written. (BIOS, etc.)

Not-So-Fun Fact: Programmable controllers may store firmware in mask-programmed ROM, UV-EPROM, or EEPROM/Flash Memory. They may store user programs and data in battery-backed SRAM, capacitor-backed SRAM, EEPROM/Flash Memory, UV-EPROM, or some combination of these. If vulnerabilities are known, patching may not be possible.

Many control systems have some sort of replaceable lithium battery to hold up SRAM data storage, or run a real time clock. While the shelf life of these batteries is 10+ years, when they are actually being used to support the memory (i.e. when sitting on a shelf as a spare part) they can deplete in less than a year.

The retention times of flash memory and EEPROM varies widely based on temperature and the amount of time since it was last powered - and, since the time frames we are dealing with are long in comparison to the existence of the components, largely theoretical at this juncture. As some of these parts have been sitting for upwards of a decade, we have started seeing failures due to flash memory "bit rot".

Fun Fact: Your facility almost certainly has multiple industrial control systems which are running building infrastructure like chillers, water pumps, and emergency power generation.

Not-So-Fun Fact: There's probably no central inventory of those systems, no backup and recovery plan, no preventative maintenance, and no budget to make any of that happen.

The scope of the problem:

ICS-Cert - As of mid-August, there have been 8 Alerts issued in 2017, and 118 Advisories issued. That's roughly as many as one weekly bulletin's worth of Vulnerability notices from US-CERT, and it encompasses only a tiny fraction of the vendors producing industrial control systems hardware and software (note also that these listings include more consumer-focused products, like the Tesla Model S and Hyundai automobiles.)

There could be two reasons for those numbers being so low:

Hypothesis A) Industrial controls equipment is among the most secure set of hardware and software in existence, since the vendors, engineers, installers, and users are all acutely aware of the risks involved should they do anything that would even remotely compromise system security.

Hypothesis B) Industrial controls equipment is among the least secure set of hardware and software in existence, because the vendors, engineers, installers, and users mostly operate under the assumption that anything which makes things go together more easily is worthwhile, that all controls systems have "Security by Obscurity" by nature, that all controls systems are isolated from the internet by multiple levels of airgapping or well-configured firewalls, and that no attackers have any interest in ICS equipment, because they are not glamorous targets. Because there is such a philosophy of pretending there are no problems, work on finding these issues is primarily done by the sorts of people (state actors, black hats) who would prefer that no reports be generated, so that the vulnerabilities remain available to them. Even if reports are forwarded to vendors, they have little or no incentive to actually change their systems since changes could break a large base of existing installed user software which the users may not have source code for.

We're here to talk about Hypothesis B, and how that kind of thinking has resulted in millions of vulnerable controls systems being installed worldwide, with more and more of them being accesible to attackers every day due to the addition of network connectivity to essentially insecure systems. We'll talk about figuring out what systems you have installed, deciding what level of action each system requires, and what options there may be for mitigations.

We'll also talk about what steps may be necessary to ensure you have a complete bare-metal recovery plan for all the control systems in your scope, and the importance of getting your IT department on board to understand, care **about**, and care **for** these control systems.

Finally, we'll discuss how detailed specification language can help ensure that any new replacement systems are not simply older insecure designs with updated hardware and more vulnerabilities than the systems they're replacing.

DHS ICS-CERT Training, available for free in Idaho Falls, ID
(I attended, and found it interesting - the red/blue exercise is some serious business.)

https://ics-cert.us-cert.gov/Training-Available-Through-ICS-CERT#workshop

Hands-On Format - Technical Level

ICS Cybersecurity (301) - 5 days

This event will provide hands-on training in discovering who and what is on the network, identifying vulnerabilities, learning how those vulnerabilities may be exploited, and learning defensive and mitigation strategies for control system networks. The week includes a Red Team / Blue Team exercise that takes place within an actual control systems environment. The training provides the opportunity to network and collaborate with other colleagues involved in operating and protecting control system networks.

Note that this course is not a deep dive into training on specific tools, control system protocols, control system vulnerability details or exploits against control system devices. This event consists of industrial control systems cybersecurity training and a Red Team / Blue Team exercise:

- Day 1 - Welcome, overview of the DHS Control Systems Security Program, a brief review of cybersecurity for Industrial Control Systems, a demonstration showing how a control system can be attacked from the internet, and hands-on classroom training on Network Discovery techniques and practices.

- Day 2 - Hands-On classroom training on Network Discovery, using Metasploit, and separating into Red and Blue Teams.

- Day 3 - Hands-On classroom training on Network Exploitation, Network Defense techniques and practices, and Red and Blue Team strategy meetings.

- Day 4 - 8-hour exercise where participants are either attacking (Red Team) or defending (Blue Team). The Blue Team is tasked with providing the cyber defense for a corporate environment, and with maintaining operations to a batch mixing plant, and an electrical distribution SCADA system.

- Day 5 - Red Team/Blue Team exercise lessons learned and round-table discussion.

- Prerequisites: Each attendee should have an understanding of ICS networks and IT network details. **Every student attending this course should bring a laptop computer (with a DVD drive).** The user must be able to boot the laptop to an operating system from the DVD. If using a DVD is not an option the user may run the operating system in a VM such as VMware Player, VMware Fusion or Oracle VirtualBox.

- This course is presented at a facility in Idaho Falls, Idaho, USA configured specifically for the aspects of the course. A Certificate of Completion will be provided at the conclusion of the course. Refer to the ICS-CERT calendar for a schedule of this training option.

Legacy Industrial Control Systems - Secure / Replace / Ignore?

2017 NSF Cybersecurity Summit for Large Facilities and Cyberinfrastructure

Phil Salkie, Jenariah Industrial Automation - phil@jenariah.com - 301-859-0500

Notes for Presentation

Our response will have four phases: **Inventory**, **Triage**, **Make Backups**, and **Take Action** (Note that during Inventory, you may find something that's so egregious that you move immediately to the "Backups" and/or "Take Action" stages.)

We're going to talk about Programmable Logic Controllers (PLCs), Operator Interface Terminals / Human Machine Interfaces (OITs or HMIs), and all the other bits and pieces which comprise the general category of Industrial Controls Systems, sometimes referred to as "SCADA", which stands for "Supervisory Controls And Data Acquisition." (This term originally meant the things which sit in various levels **above** the controllers and watch over them - note that if you talk to an actual controls contractor about a SCADA system, they may well have an entirely different set of expectations than you do.) Specifically, we're going over the reality of finding out that you are in charge of information security for dozens of types of hardware, firmware, and software which almost no-one knows you own, whose vendors may no longer exist or may not support the equipment you have, which contain programs that are crucial to your facility's continuing mission, and which may not have so much as had a reboot in over a decade.

This class is called “Legacy Industrial Control Systems - Secure / Replace / Ignore” - let’s define each word, and how much each word encompasses.

Industrial - "Industrial" as opposed to "Consumer" - we're not going to talk about securing your TV, or your Nest, or your IoT Dishwasher. We’re thinking about the controls which run your building HVAC, your elevators, your data center’s chillers and emergency generators. Also those that run bioreactors, position telescopes, dessicate samples, and control core drills.

Control - devices which have physical access to the real world, and can cause real-world harm. If I crack into your desktop PC, I can create havoc in your file systems, and potentially cost you quite a bit of money one way and another.
If I crack into your LN2 storage tank controller, I might be able to cause significant physical damage and/or bodily harm to anyone near the tank, not to mention damaging experiments in progress or causing patients to lose access to treatment equipment.

Systems - An isolated controller has a limited attack surface that involves physical access, or possibly has a single network connection. It's the ones which are integrated into larger control systems, with multiple controllers, multiple operator interfaces, SCADA systems, and other external devices - these systems have multiple possible attack points and are the much greater challenge to secure.

Legacy - in this context, I'm defining "Legacy" as anything that's currently installed. There's basically been very little emphasis on InfoSec in the controls industry, so it's depressingly likely that something installed last week is orders of magnitude more vulnerable than something installed five years ago. That’s because newer controllers are all being supplied with ethernet ports, an expensive rarity in a controller from 2010. Although ethernet ports are being supplied, the underlying software layers generally do not have the layers of protocol security which are expected in modern operating systems (HTTPS, SCP, SFTP, shared keys, robust passwords.)

Secure - that's an awfully broad term! For $5, you can buy a TSA approved luggage lock, which anyone can find a key for with a single Google search. Or you can spend upwards of five grand, buy a bullet-resistant ultra-high security door lock with a registered, pick and bump resistant RFID key. Or spend any amount in-between. Part of our task is to determine which systems get what level of security, and in what form or forms. Controls system security means anything from:

Preventing unauthorized access ("**Physical**")

Improving power conditioning and reliability and/or adding surge and lightning protection ("**Electrical**")

Fixing issues which cause the equipment to be misused or improperly controlled by the users ("**Operational**")

Collecting observational and diagnostics information on the controls system and providing for reporting of deviation ("**Monitoring**")

Securing the digital networks which are used in the operation and supervision of these controllers ("**Data** or **Cyber**")

Security on multiple levels is part of a program of **Defense In Depth**, the various things we do at each stage can help prevent an incident or they may slow down an intruder or delay an inadvertent action enough so they are detected and blocked before damage is done.

There will be systems which can be easily secured, systems which can be updated, or firewalled, or isolated - and there may be systems which you can't touch at all, and the only thing that's possible is to set up monitoring and alarms on their network traffic. There are some systems which, to be honest, aren't worth the time and effort it would take to secure them. We'll talk about all these types of systems, how to determine which is which, and what to do once you know what you're dealing with.

Replace - If you’ve made the decision to replace a control system, how can you be sure that you’re not just swapping one nightmare for another? Modern controllers can have a much broader attack surface and have more vulnerabilities than older systems - good specifications and a good knowledge of the devices being controlled can make all the difference.

Ignore - How can we suggest ignoring a known problem? Well, everything in life is a trade-off, and it may well be that once you’ve determined what control systems are out there, you’ll find that some are important, some aren’t as important, and some just don’t matter right now. That’s not to say they’re not on your inventory list - you’re not going to forget they exist - but at the present time, you’ve decided that if they keep running, that’s great, and if they stop for some reason, that’ll be OK too. Meanwhile, you’re focusing your limited time and funding on more mission-critical components, and if the ornamental fountains stop pumping, everyone’s going to just have to deal.

It's likely that those of you who are sitting here aren't going to actually be doing the physical work which I'll be discussing - you'll probably be involved in contracting this work out to some firm, or perhaps overseeing a team of folks who will be assessing your systems and making changes to them. Maybe you'll be requesting budgetary line items, or deciding which systems get funding for remediation.

Here's why it's so very, very important that you be familiar with the issues involved in identification, discovery, isolation, mitigation, monitoring, repair, upgrade, and replacement of ICS systems. These systems are often the oldest, least documented, least supported, and least understood of anything in a facility, yet they can be in control of mission critical hardware while being vulnerable to attack in ways which are difficult to mitigate. There are also many, many vendors, consultants, engineers, and officials who have some expertise in this field, and will happily sell you their services in one way or another - yet you may quickly find that their expertise doesn't match your problem space, or their remediation methods won't work with your facility, or the action plan they submit to you is missing a few critical steps - resulting in downtime or total system failure. (Or you may get lucky and they'll just do way more than was actually necessary, and charge you accordingly.)

In this problem space, knowledge is power - and we're going to start off by talking about how we acquire and apply that knowledge.

Phase 1 - Inventory

What's out there? Knowledge of existing systems

Build an initial Controls System Inventory List

For each system, initially you need to know:

Name (for your reference) - be sure panels are labeled!
Number of panels/boxes/stations in the system
Physical location of each panel
Level of importance to business operation (a system may be more than one of these)

Mission Critical - Telescope Positioner, Data Center A/C, BioReactor, Data Collection
Building Infrastructure - Elevators, Building A/C, Sewage
Backup Systems - Fuel for Emergency Gens, Power Transfer
Important but Bufferable - Production Chromatography, Building A/C, Generator Fuel System, lighting controls (These are systems which can be shut down for a while, can batch up their output, or which may be able to run in a manual control mode for a while during maintenance.)
Monetary cost if offline - solar power, microturbines, Regulatory monitoring
Non-Critical - Lawn sprinklers, fountains, ancillary lighting, secondary solar collectors, offline experiments

Known level of connectivity to Internal Network, Internet, Wireless, and POTS
(Don't overlook serial connections to other devices and PCs, wired telephone modem connections, nor wireless connections to sensors and internet, including 3G/4G data modems. Watch out for devices which bridge networks together inadvertantly, and remember that sub-components like OITs and UPSs can have connectivity too.)

Rough date system installed
PLC Brand (If present)
Operator Interface Terminal Brand (If Present)
SCADA Software Brand (If Present)
OS of SCADA PC (If Present)
Are custom controller boards installed?
All known passwords to every system device

Also, if readily available:

Name & contact info of System Integrator
Name & contact info of Panel Builder (UL File number can help with this)
Are operation and maintenance manuals on hand?
Are panel drawings on hand?
Are spare parts on hand?
Is programming software and cabling on hand?
Are PLC, OIT, SCADA program sources on hand? How about data tables and recipes which are necessary to operate the equipment, but may not be backed up?

Phase 2 - Triage

Which systems will get the most attention soonest?
What's the rough budget for mitigation?
What's the rough timeframe for mitigation?
What level of defense will each system get?

Threat profiles:

- Random intruders from the internet
- Random intruders from the house network
- Targeted intruders (industrial espionage) from internet or house network
- Internal espionage via network or physical access
- Extreme espionage (state actors) via diverse means
- Non-malicious employees with physical access accidentally pushing buttons, moving valves
- Protection against power/weather events
- Protection against loss of communications

Systems which are not connected and aren't going to be connected may be able to receive a lower priority, depending on their location and their function. Remember - if there's something that's unconnected but mission critical, you may be better off to _increase_ its connectivity in order to improve supervisory monitoring, especially if it's in a remote location which isn't normally manned, it's mounted outside, etc.

The result of the Inventory and Triage phases should be a list of all controls systems which are under your authority, how many panels each system has, where they are, what sort of controllers they have, how they're currently connected, how important they are to your mission (in various ways), the general age of the system and its software components, what level of documentation you have, what avenues of support may be available to you, what sorts of threat you think each system deserves to be protected against, and a feeling for the timeframe and the money you have to do the work.

Notice there's no "Penetration Testing" or "Verify Presence Of Possible Vulnerabilities" or even "Run NMAP Against The Controls System Network" in this procedure. ICS systems can be very brittle, and the vulnerabilites can be severe - picture the ability to read or overwrite, via an unsecured network or serial connection, any byte of disk space on a PC. Now picture that you have no backups of that PC, and it just happens to host your main data store. That's the level of vulnerability which is built in to most implementations of the Modbus protocol, spoken by almost every PLC and OIT in the world.

Now add to that the fact that some controllers will respond to protocol commands on any TCP port - meaning that you can send Modbus requests on port 25 (usually reserved for email), or programming requests on port 502 (Modbus TCP). All the more reason that even a more or less "safe" scan can cause havoc on controllers.

Once you've got a feeling for what systems you want to address first, you can start on a more detailed analysis of those systems.

Can it be powered down/stopped?
Some systems have remained powered since install, others are powered down regularly.
If PLC battery has failed since install, powerdown could lose PLC program and recipe data.
Power cycling could also affect the process and may need very specific advance scheduling.

Backup Battery in PLC?
Has it ever been changed? Can it be changed while system remains powered?
Does the PLC alert if battery is low?

Is there a backup/reserve system?
Has failover been tested? Is it known currently to work?
Is the reserve system complete control, or partial?
Is human intervention needed to transfer over to the backup system, or to continue full control when backup is running?

Do we have the programming tools and cables for all components?
Some older programming software tools are difficult to find.
Some require unsupported operating systems (XP, DOS) or laptops with built-in serial ports (not USB).
Some newer software won't work with controller programs which were built on older tools.
Most older PLC systems require special cabling.

Do we have software for all components?
PLC CPU, PLC Function Cards, Display Screens, 3rd Party Modules
(Not always easy to determine if cards get software loads, or to verify the software that's been loaded into the cards.)

If not, can we pull software from it?
(Many older PLCs have easily defeated passwording, we can often use that to our advantage to recover password-protected software.)
If we pull the software, it may only be the raw program, with no supporting documentation.

Can we verify that the software we have is what's running now?
Not all PLCs can do that. Some of the ones which can will only show that there is a difference, not give you a line-by-line breakdown of what has changed - sometimes you'll resort to measures such as generating text printouts, editing them to remove headers and footers, then using diff to compare the two.

Inventory of components and hardware/firmware versions
(This may require disassembly of stacked components to find labels, and/or connection with programming tools to find firmware info.)

Complete network map
This should include all controls components, switches, NAT routers, cabling, IP addresses, network types (remember, there are many possible ICS networks), other network addresses.
What networks are supported by in-house IT Department, what are controls-system-only?

Again, don't overlook serial data streams like Modbus, CANbus, HART, and so on, also remember wireless systems - point-to-point wifi, Zigbee, 3G/4G Data, etc.

Firewall rules
Obtain a complete listings of rules which relate to the controls system address spaces - look for open doors like PLC being placed in DMZ, or "Allow All" in ruleset.

Vulnerabilities
Once system discovery is complete, we can look in vulnerability databases to find out what potential threats there are against the hardware and networking systems we're using - but honestly, you may wind up just assuming the worst and working from there. If you assume that nothing is passworded, that any passwords which might be used are stored in plain-text or have hard-coded alternatives or are brute-forceable, that attached PC systems have programming tools sitting on them with the passwords readable, that operator interfaces are accessible via VNC or web pages, that any network connection can be used to reprogram, modify data values, or crash a controller, then you'll be more prepared to look at ways to really secure these very vulnerable and important systems.

Attack entry points
Networks, physical access, connectivity, accidental operation or shutdown, power issues

Possible results of attack
Target value to attacker
Target value to owner

Budget for remediation
First year budget may all be for discovery

Level of provider support
Drawings, software available? Upgrades?
If the Systems Integrator or Manufacturer are still available, they may be the best avenue for remediation - however, it's highly unlikely that they've given any thought whatsoever to ICS cybersecurity concerns, especially regarding older systems.

Level of manufacturer support
Hardware and Firmware upgrades?)
Many manufacturers have simply ignored all notifications of security issues with their hardware, some have provided firmware upgrades, some have released hardware with updated versioning numbers which contain updated system ROMs that fix vulnerabilities.

At this point, we know What Systems we have, and we know What Order we are going to address the systems in, and we know roughly what level of work we intend to do to mitigate the various vulnerabilities we see.

Phase 3 - Backups

Backing up ICS systems is not as simple as plugging a USB Hard Drive in and running some software. There are often many physical components required to really have a "bare metal" backup/restore solution - the good thing is, due to the nature of control systems, generally once a full backup is made, it doesn't need to be repeated often since the programs (and often the data) remains unchanged for long periods of time.

Batteries - not just CPU & OIT, some comms and motion cards as well.

Hardware required for programming

- Needs native serial port?
- Needs special cables?
- Needs PCMCIA Card?

Programming Integrated Development Environment

- Activation of software?
- Hardware drivers?
- Dongles (Parallel, USB)?

OS to support IDE

- Could be MS-DOS, Windows XP, Windows 7
- Can be quite specific as to patch level

Firmware version that's currently running on controllers

PLC / OIT program to load

DATA For PLC program if that's stored separately

Comms drivers for OIT - verify driver versions

What media is everything stored on? What media does the hardware need?

Consider obtaining/making a VirtualBox VDI file with everything needed.

Consider obtaining a laptop with everything configured, then label the heck out of it so it doesn't disappear when somebody is tidying up or getting rid of old, unused stuff. Remember that solid state hard drives can lose data within a year of sitting unpowered - it's a good idea to insist on a conventional hard drive in the laptop rather than a solid state drive.
(Special cabinets exist to store laptops permanently powered and sitting in suspend mode.)

Programs that get loaded into special function cards

- Communications Modules
- Co-Processor Modules
- Motion Controller Modules

Cabling for these special function modules
IDE software for these modules

Verify anything that can be verified - some things can't, which puts them higher on the "Replace" list.

Bring in component-level spares if you don’t have them

Power Supplies
Racks
CPU + any memory cards or add-ons
I/O Cards
Special Function Modules

It may well be worth building up a complete spare controller system which will let you test the restore process - even to the point of making a restore, then (at a propitious time) swapping out the CPU and other components for the backup units to make sure that they properly function.

Phase 4 - Take Action

For each system, decide what level of response it will receive:

Ignore
What the vast majority of vulnerable system owners are doing already - the difference being, we actually know what the system is, and we are making an informed decision to ignore it (for now.) Depending on the system, its function and benefit to the organization, and how it's connected to the outside world (if at all), it may have such a small risk profile or low vulnerability that it's not worth doing anything to it - but at least you'll _know_ that and have acted accordingly.

Monitor/Protect
(Not modify the core system, but add features to prevent or detect tampering.)

This includes physical access control to panels and operator devices, tamper switches on doors and on manual controls like valve handles, Intrusion Detection System rules to observe network traffic and alarm on changes from a preset standard (ICS traffic tends to be extremely predictable and very easy to set useful alarms on.)

Isolate
(Cut the cord - possibly lose functionality)
Some steps can be as simple as turning a key from "Remote Program" to "Run". Now the controller will need manual intervention before the PLC program can be modified - if it's a remote or unmanned station, this may not be a practical answer.
If someone says "But we need to be able to reprogram remotely!" feel free to ask "Why? What haven't we considered in the programming that requires it to be modified on a regular basis?" Remember that the overwhelming majority of ICS systems are programmed at installation, then rarely if ever modified.

Could wind up losing remote access for vendor service - is that function being used? Often it's in place, but dormant. You may well have been paying for service that never occurs.

Could wind up losing remote access for monitoring or other important data collection tasks. Note that not all data collection is continuous, especially in older systems - often a system will save days' or weeks' worth of log information, and a desktop PC somewhere will remote connect weekly or monthly to pull data from the PLC. Other systems will send an email on a timed basis, or even dial out on a modem to connect to a remote data collection device.

Enable and/or Upgrade passwords wherever possible - PLC, OIT, Smart Switches/NAT Routers/Gateways, SCADA PCs. This is far from a panacea, but it's all part of a program of Defense in Depth - everything that slows down an attacker increases the chance they will be detected before doing damage.

Remove configuration/programming tools from local PCs which can reach the PLC/OIT unless there's a _really_ good reason. Leaving those tools available on a laptop that's normally not connected and lives locked in a drawer is often more than enough.

Manually powered-up ethernet switch with timeout - only allow access after manual intervention, and only for a limited time. This is useful for situations where access is needed only for remote maintenance work.

Timer-powered ethernet switch - only allow access during pre-programmed times, like daily between midnight and 00:15. This is useful for batch data collection, cuts amount of time available to intruders to try their attacks.

While air gapping of any sort is not a perfect solution, it's a huge decrease in potential attack surface, and every little bit helps.

Firewall
(Maintain or even gain functionality)
Many industrial control protocols are wide open and have no means for authentication. Many implementations allow for full read/write access to the entire PLC's data memory, some also allow programming and even firmware access without forcing authentication.

Dedicated protocol firewall devices exist for some PLC protocols, primarily for Modbus since that's the "least common denominator" of PLC protocols.

Stand-alone protocol converters can be used as firewalls, limiting the amount of accessible data and making most or all of the data be read-only - sort of like a “Data Diode”, but without the need to invent special protocols for one-way physical data transfer.

Many operator displays have protocol conversion features, which allows the addition of data monitoring and control displays while improving system security - possibly also adding remote monitoring in a more secure way while maintaining existing controls connections.

Interposing a small conventional hardware firewall between a control system and the house network will allow you to limit the IP addresses which can query the controllers - control systems generally have a limited number of IPs which will attempt to access them.

Add a "communications interface" PLC to the controls system network, which is unable to write to the other PLCs and is only written to and read by them. It becomes a sort of DMZ for controls system data, any external devices read from and write to only this new PLC.

Intrusion prevention systems can be programmed to monitor traffic between house network and ICS network, that traffic tends to be limited and predictable - bit-bucket any traffic not from proper IPs or which is attempting to access the wrong ports.

Update hardware and/or firmware
(Form/fit/function replacements)

Many legacy PLC systems don't have updateable firmware. Some security issues can be addressed with an updated module release, which is almost always backwards compatible with existing systems.

While fixing known issues with device firmware is helpful, it is a rare system where this would be the only action indicated.

Upgrade system
(Replace PLC, replace OIT, but keep panels in place)

The PLC is just one part of a control panel. Often it's worth leaving the other components in place, and changing just the controller.
Look at issues of form and fit, whether wiring will reach the new PLC terminals. If fit is an issue, it may be possible to add a small panel containing just the PLC, and cable over to existing connection points.
Research to ensure compatibility with other systems which communicate with the device being upgraded.
Logistics of porting software from one platform to another, even when the new PLC is the same brand but a newer model.
Addition of security and monitoring features for vulnerable physical devices

Cabinet locks
Door tamper switches
Valve tamper switches
Power and air supply pressure monitoring

Replace entire system with a more modern system.
(Can be a very expensive option, but may be less costly overall than spending lots of time digging into poorly-documented existing systems.)

This option is often less painful because operation of the new system can be bench tested to some degree, and transition can be scheduled for an acceptable down-time (or some dual-operation method devised so that the change-over is done without losing complete functionality.) This can also be justified by increases in function, improvements in overall maintainability, even power efficiency.

Recognize that installing new equipment without paying proper attention to security during the specification and design phase can result in a system which is much more vulnerable than the one it has replaced.

If “Replace” is the chosen option, it’s likely that the research you’ve done into finding out as much as possible about the existing system will make the process of specifying, obtaining, installing, and commissioning the new system much easier and provide a much better result.

Sample Specification Language for Replacing Existing Control Systems

NOTE: PLC = Programmable Logic Controller. HMI = Human/Machine Interface
NOTE: Not all these ideas will apply to yours, some are contradictory and represent different levels of security paranoia.

General Good Ideas

Vendor shall produce a detailed description of system operation, specifying the function and behavior of each physical input and output point, describing all internal and external data pathways, and enumerating all data tables used for external control and monitoring of process logic, system status, and warning and alarm conditions. This description shall be used by the vendor to produce a system test plan which will be used to verify correct operation of the system at commissioning.
{Language like this is because if you don't ask for it, they won't provide it.}

Panels shall be built and labeled to the UL 508A standard.
{Just a good idea. UL makes sure that devices are designed not to catch fire. We like that.}

Panel Longevity/Maintenance Language

Controls hardware shall be PLC-based. HMI systems shall be dedicated panels..
No commercial general-purpose computing hardware shall be used.
PLC and HMI hardware shall have a minimum manufacturer support window of ten years.
{Trying to avoid having an entirely unsupportable computing platform six years from now.}

Panels shall be labeled internally with the integrator's name and contact information, and (if different) the manufacturer's name and contact information.
{This will matter after the panels have been in service for a decade.}

If any equipment in a panel uses a replaceable battery, that panel shall be labeled externally with the battery function, part number, installation date, and recommended replacement date.
{This makes the whole issue of lithium battery lifespan much easier to deal with.}

A laptop computer with a magnetic hard drive (not SSD) shall be provided which contains all development software required to program the PLCs and HMIs. All required software licenses shall be installed and registered to [Insert Facility Name Here]. All PLC and HMI programs, as well as all CAD drawings, manuals, and ancillary information shall be stored on this laptop. The laptop shall be physically labeled showing that it is the programming tool for [Insert Project Name Here].
{This may become very, very important after several years. You can consider insisting that a second laptop be provided, which you can store in a retention facility like Iron Mountain.}

Data Security Language

Programmable Controllers shall have a physical switch to select between Run, Remote Program/Run, and Program modes.
Controllers can not be programmed or remotely stopped when switch is in “Run” mode.
{I like this concept, but it pretty much locks you into Allen Bradley controllers.}

No PLC or HMI systems shall use wireless communications for any reason.
{Most locked-down, probably won’t affect your system’s design or cost unless you’re trying to get data from some remote device.}

PLC or HMI systems which use wireless communications must use only dedicated industrially-hardened point-to-point systems (i.e. Zigbee, WirelessHART), NOT systems which use 3rd party external gateways (i.e. 3G, 4G, LTE) or consumer-grade general-purpose networking (WiFi, Bluetooth)
{Use this if you need to get data from a remote tank or facility - if you can possibly avoid sending data over commercial airwaves, you’re better off from a security standpoint.}

No ethernet cable may be run between panels. Ethernet may be used inside of panels (i.e. for communications between PLC and HMI) but must be point-to-point - no switches, routers, or NAT routers may be installed in the panel. Ethernet to serial bridges (protocol converters) may be used to convert internal ethernet to an external serial protocol.
{Most restrictive. This is similar to language I’ve seen used by the Federal Reserve Banks.}

Communications between panels must not be TCP/IP based. Acceptable protocols include (but are not limited to) Modbus RTU, DeviceNet, ProfiBus, DL485, CC-Link, CANBus, and BACNet. {Again, Federal Reserve.}

If TCP/IP communications is used between panels, all IP switches shall be DIN rail mounted in the panels. IP Addresses used shall be from a pool defined by the IT department which does not intersect with the house IP pool. Any connection between house networks and controls networks shall be via a separate, dedicated Ethernet port on a controls device or via a NAT translating gateway located in one of the panels. Vendor shall provide a complete network map of the system, showing all ports in use, all data paths, and all protocols used.
{This maintains a separate network which has extremely limited points of contact with your house network. You may want to specify IP switch brand and part number, ensuring your IT deparment’s familiarity with the hardware and its capabilities. Note that not all controls network protocols behave nicely on all ethernet switches, especially smart switches. You may want to monitor spanning ports on these switches with your IDS.}

Controller and HMI programs shall be stored in a human-readable ASCII format, a compressed (zipped) ASCII format, or are capable of being exported to an ASCII format.
{This makes use of standard version control systems like “git” possible, but will likely confuse most vendors, who aren’t used to using any form of version control at all.}

Side note: ICS/SCADA as your very own "Shadow IT" department...

Controls system networks versus house ethernet:

IT department thinks controls people are sloppy, don't document anything, don't know anything about networking or the requirements of the hardware they're bringing, don't know what operating systems or software versions they're running, never patch or update anything, insist their data can't co-exist with any other data on the network, use weird IP addressing schemes, don't use any sort of IDS systems, have problems with advanced networking technologies, bring in thousands of dollars of hardware and software which doesn't get tracked, has no asset tags, no upgrade plans or maintenance agreements, provide no technical support, and generally are total control freaks.

Controls vendors think house IT people can't deal with anything that doesn't run Windows (and don't ever mention Windows CE, because they'll insist on trying to patch it every other Tuesday), take forever to do simple things like assign an IP address, insist on putting controls-related switches in data center racks clean across the building, use networking hardware which buffers packets to the point that PLC comms stops being reliable, will randomly reboot network switches (causing controls calamaties) or just disconnect cabling to check connectors or dress wires, and generally insist on decreasing the stability and reliability of a controls system to the point that it only works on Thursday afternoons.

(And they're both correct, to a point.)

Problems of integrating the knowledge bases of controls system networks and house IT networks:

The large majority of ICS systems with communications capabilities are not integrated into house IT networks. They have isolated networks which use a variety of protocols (see the Not-So-Fun Facts above) hardware implementations (Ethernet, Twisted Pair, 3G/4G, direct microwave, POTS, leased line, parallel bus, digital and analog I/O)

Unusual/stealth communications - you will sometimes find that systems transfer information in unexpected ways, sending each other an email, placing data on a web page, or using the physical systems under their control as a signalling method.

Many IT departments consider ICS the purview of the facilities engineers. Many facilities engineers consider ICS hardware to be a “Black Box” which will basically continue doing its thing until cockroaches rise up and take over the planet. Backups are rare, may be on old media (5-1/4“ floppies are depressingly common) and may be incomplete, or out of date, or impossible to verify.

As more and more controllers and displays sprout ethernet ports, it’s becoming critical to get IT involved with ICS. There are panels out there with $5000 CPUs and without asset tags. There are unmapped networks, unprotected data paths, and unknown communications methods. Critical infrastructure is depending on systems with no backups, no recovery plan, no scheduled maintenance, and no spare parts. IT is capable of dealing with all the issues which securing ICS brings, but IT is rarely invited to weigh in, and rarely is given authority to catalog, back up, and secure these systems.

Some Stories:

Three from just last week

Customer manufactures, repairs, and tests components for commercial and military aircraft. I was called in to modify software on a panel which we built in 1997, and which I had last worked on in March of 2000. Customer had retained no backups, cabling, programming tools, or other information (all of which had been provided at delivery.) The only reason we were contacted was one person remembered enough information about us to look in the Yellow Pages (!) and recognized the name of my previous employer. This panel is in daily use, and has been since installation - it is one-of-a-kind, and if it were to stop working, there's no quick or easy method of replacing its functionality. In order to so much as verify that the programs I had archived were what the system was actually running, I had to run MS-DOS based programming tools because the more modern Windows-based tools showed (incorrect) verification errors. We're now working on a program of obtaining spare parts and ensuring they have all tools and cabling required, and budgeting for a modern replacement system...

Customer is a large data center which we all rely on. When the building was built ten years ago, the data center was turned up before all components could be tested and commissioned. The emergency power generation system's fuel tank controls were built by a small firm from across the country, and its alarms weren't getting properly transferred to the building's alarm system. As an interim measure (for a decade) they relied on the security guard patrols to report the presence of an audible alarm behind a locked door to find out if the fuel system was working properly. I was called in to see if I could diagnose and repair the problem - I found that the customer had no software, no passwords, and no programming tools. I was able to modify the software in some communications translators to properly forward the alarm and status data, and showed that the data was properly populating the alarm maps in the Building Management System - but we couldn't test the page-out functions due to other issues with the alarm system. Last week I got a call saying that an alarm was stuck on and wouldn't clear from the display - on investigating, I found that there was an alarm, but it was on a tank in a different room (with a broken alarm sounder.) Turns out that each tank has three different names - a network name (DT-1 through DT-6), a tank name (DT-7 through DT-2) and a generator name ("MR-1", "M-2", etc.) When the alarm point list was made ten years ago, the list of generator names was inverted somehow, so every tank's alarms paged out as a different tank's name...

Customer is a food manufacturer and packager - they bought a system from Italy about ten years ago to count, weigh, and package food in cups. The machine was programmed using free software from the controller manufacturer - but in the intervening years, that software has been dropped from support, and communications drivers are only available for Win XP. A modern programming package is available, but it is locked to the device it's installed on, and is over $2800 per seat - customer wants a laptop and a desktop copy, so it would be over $5000 for the setup. I was asked to make a Win XP laptop, patched up as best I could and with all ethernet disabled, to use as a programming workstation running the free software...

Links

Dedicated Modbus Firewalls:

https://www.tofinosecurity.com/products/Tofino-Modbus-TCP-Enforcer-LSM
http://www.moxa.com/product/Industrial_Secure_Routers.htm
http://www.waterfall-security.com/solutions/industrial-protocols
http://www.sequi.com/portbloque-e.htm

Protocol Translators which can act as isolating firewalls:

http://www.redlion.net/products/industrial-networking/communication-converters/protocol-converters
http://www.sierramonitor.com/connect/all-protocol-gateway-products

Operator Interfaces which can do protocol translation as well as add real-time visual monitoring:

http://www.redlion.net/products/industrial-automation/hmis-and-panel-meters/hmi-operator-panels/
http://www.beijerelectronics.com/

Other Interesting Things:

History of PLCs:
http://www.control.lth.se/media/Education/DoctorateProgram/2012/HistoryOfControl/Vanessa_Albert-PLCDCS.pdf

Rockwell (Allen-Bradley) presentation on ICS CyberSecurity:
https://www.rockwellautomation.com/resources/downloads/rockwellautomation/pdf/events/packexpo/PE-07_Securing-Manufacturing-Control-Networks.pdf

Forums for PLCs and Controls:
http://control.com
http://www.mrplc.com/

DHS ICS CyberSec Training - Idaho Falls, Idaho:
https://ics-cert.us-cert.gov/Training-Available-Through-ICS-CERT#workshop

Sources for older PLC and OIT hardware:

http://radwell.com
http://ebay.com
http://labx.com
http://amazon.com
http://alibaba.com