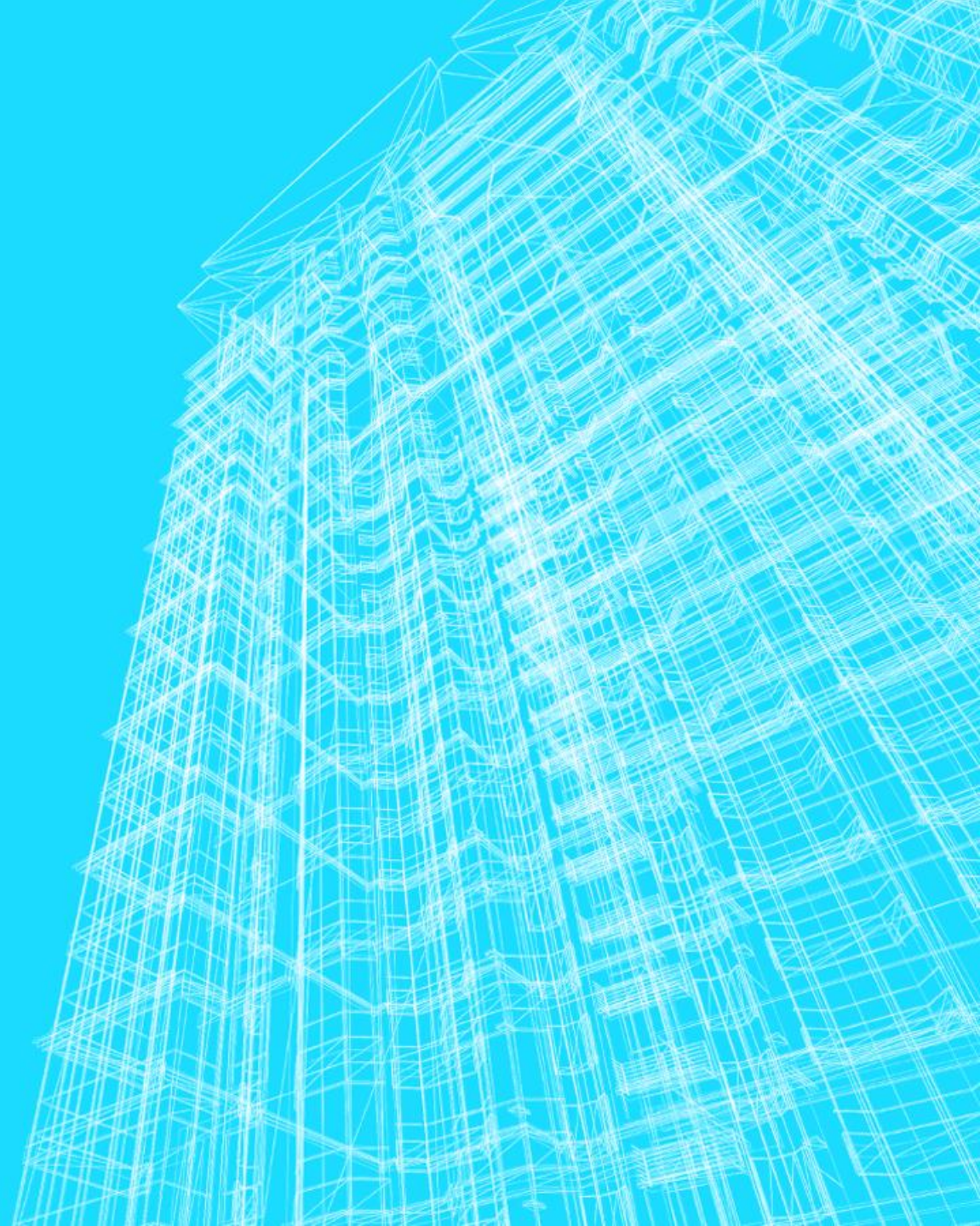


PLC TOP 20

PLACE AND DATE

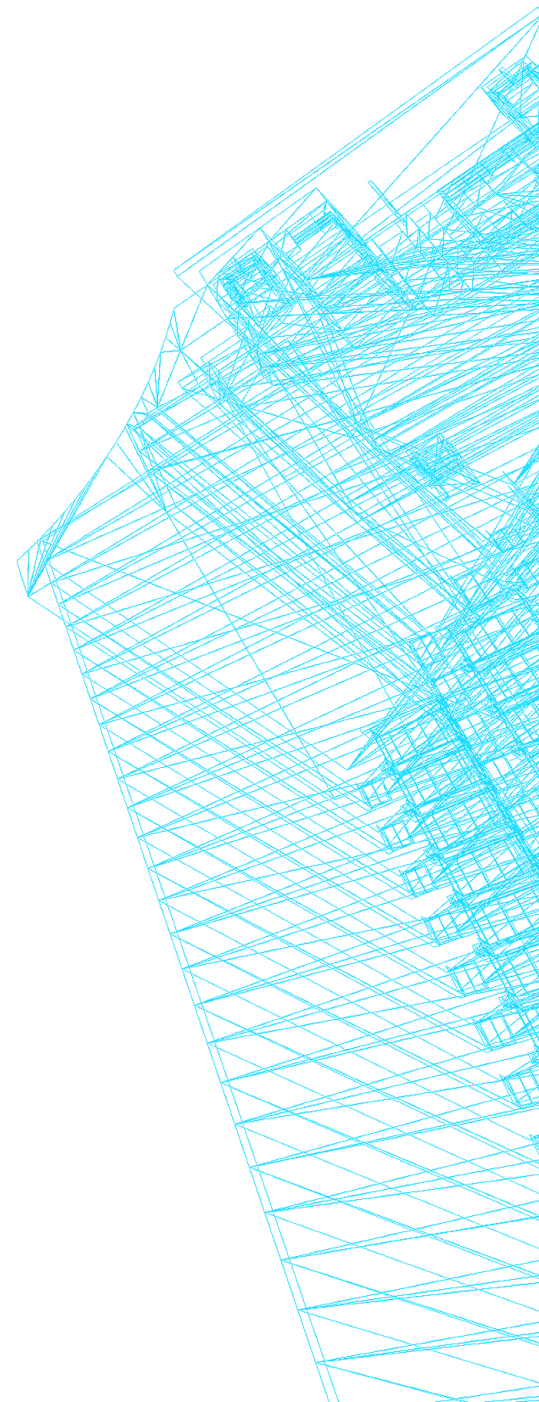




Agenda

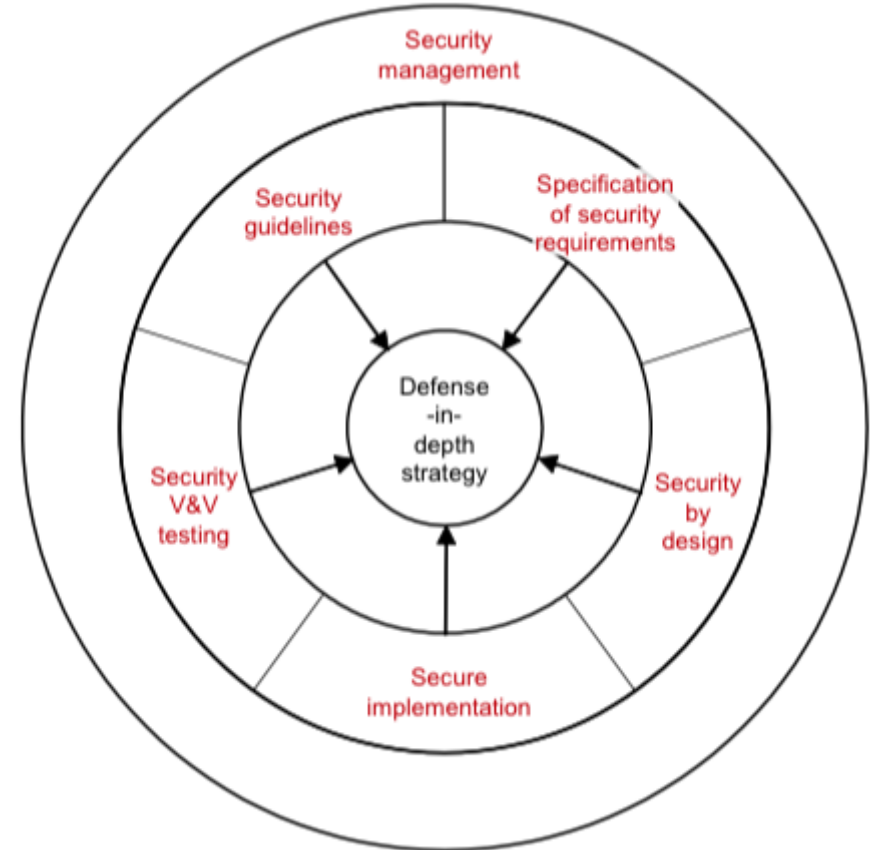
1. Why the need for the PLC Top 20
2. The Top 20
3. Want to learn more?
4. How to contribute?

WHY THE NEED FOR THE PLC TOP 20?



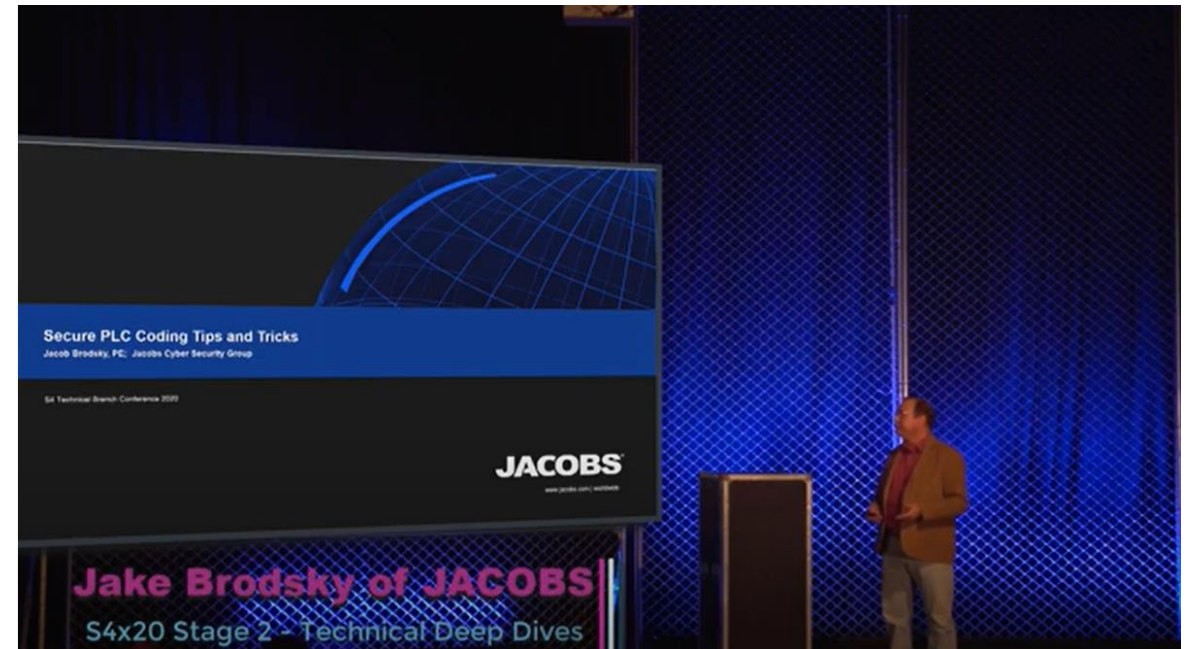
Why the need for the PLC top 20

- A lot of OT cyber security focuses on the IT components that support OT
- The concept of “secure coding” or “secure by design” is important
- Building secure PLC logic helps achieve a ‘Defense-in-Depth’ Strategy
- Not only for security, but benefits **Reliability** and **Maintainability** of automation logic

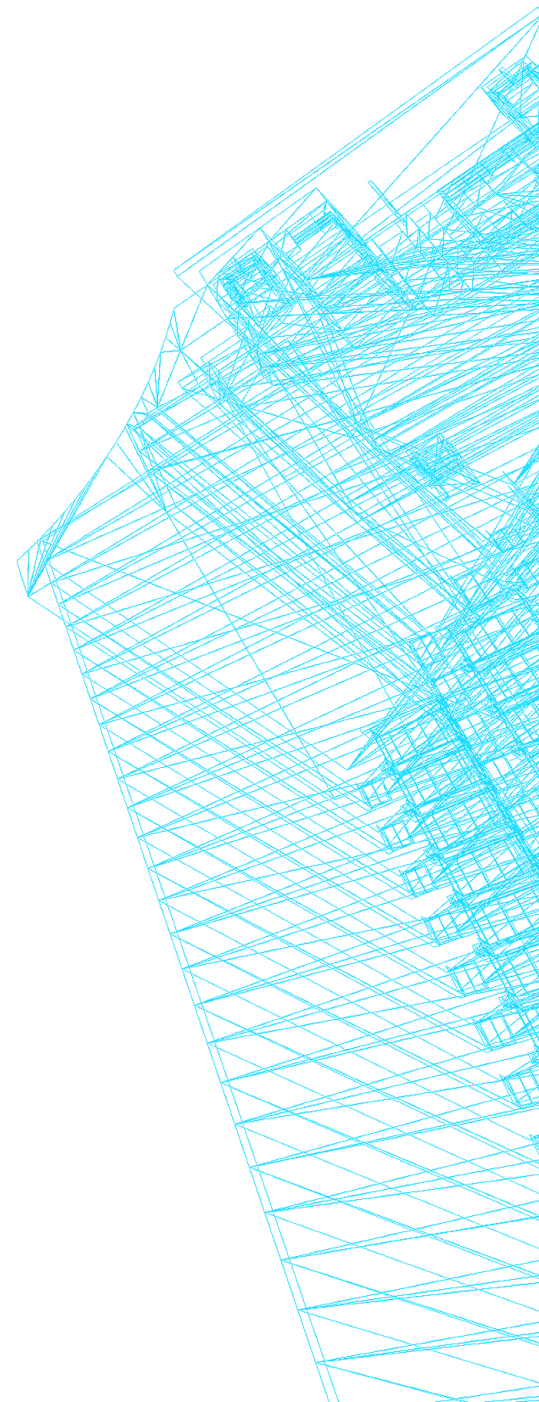


PLC Top 20 History & Project Overview

- Started with Jake Brodsky's S4x20 "Secure Coding practices for PLC's"
- Formed into a global OT Cyber security project supported by ISAGCA
- First version released June 2021



PLC TOP 20



Just before we start ...

2. Track operating modes

Keep the PLC in RUN mode. If PLCs are not in RUN mode, there should be an alarm to the operators.

Security Objective	Target Group
Integrity of PLC logic	Integration / Maintenance Service Provider Asset Owner

Guidance

If PLCs are not in RUN mode (e.g., PROGRAM mode), their code could be changed to track the RUN mode. Some PLCs have a checksum to alert for code changes, but if they do not, there's at least an indirect indicator of a potential issue while tracking operating modes:

- If PLCs are not in RUN mode, there should be an alarm to the operators. If they are aware that someone is supposed to be working on that control system, they can acknowledge the alarm and move on.
- The HMI should be configured to re-alert the operator toward the end of the shift about the presence of the alarm. The goal should be to keep track of any staff or contractors in the plant doing work that might impact the process.

Exception case: If the plant is in a testing or development phase, consider disabling this alarm but the plant should be isolated from higher levels of the network.

Example

If the PLC does not have a hardware switch for changing operating modes, it is recommended to at least make use of software mechanisms that can restrict changing PLC code, e.g., password protection in engineering software for reading and writing PLC code.

Why?

Beneficial for...?	Why?
Security	The operating mode (run / edit / write; for Allen Bradley PLCs: RUN / PROGram / REMote) determines if PLC can be tampered with. If the key-switch is in REMote state, it is technically possible to make changes to the PLC program over the communication interfaces even if the PLC is running.
Reliability	/
Maintenance	/

References

Standard / framework	Mapping
MITRE ATT&CK for ICS	Tactic: TA009 - Inhibit Response Function Technique: T0858 - Utilize/Change Operating Mode
ISA/IEC 62443-4-1	SI-1 : Security implementation review



PLC Top 20 Version 1

1. Modularize PLC Code
2. Track operating modes
3. Leave operational logic in the PLC wherever feasible
4. Use PLC flags as integrity checks
5. Use cryptographic and / or checksum integrity checks for PLC code
6. Validate timers and counters
7. Validate and alert for paired inputs / outputs
8. Validate HMI input variables at the PLC level, not only at HMI
9. Validate indirections
10. Assign designated register blocks by function (read/write/validate)
11. Instrument for plausibility checks
12. Validate inputs based on physical plausibility
13. Disable unneeded / unused communication ports and protocols
14. Restrict third-party data interfaces
15. Define a safe process state in case of a PLC restart
16. Summarize PLC cycle times and trend them on the HMI
17. Log PLC uptime and trend it on the HMI
18. Log PLC hard stops and trend them on the HMI
19. Monitor PLC memory usage and trend it on the HMI
20. Trap false negatives and false positives for critical alerts



1. Modularize PLC Code

*Split PLC code into modules, using different function blocks (sub-routines).
Test modules independently.*

Security Objective

- Integrity of PLC logic

Target Group

- Product Supplier

- Assists with input validation, access management, integrity validation etc
- For Security - Facilitates the detection of newly added code that could be malicious
- For Reliability - Helps control the program flow and avoid loops and can improve reliability
- For Maintenance - Makes the code easier to maintain, debug and update – assists with code reuse between projects



2. Track operating modes

Keep the PLC in RUN mode. If PLCs are not in RUN mode, there should be an alarm to the operators.

Security Objective

- Integrity of PLC logic

Target Group

- Integration / Maintenance
Service Provider Asset
Owner

- Makes it difficult to load new code and helps with change control
- You should generate operator alarm on PLCs that are not in RUN Mode

3. Leave operational logic in the PLC wherever feasible

Leave as much operational logic e.g., totalizing or integrating, as possible directly in the PLC. The HMI does not get enough updates to do this well.

Security Objective

- Integrity of PLC logic

Target Group

- Product Supplier
- Integration / Maintenance Service Provider
- Asset Owner

- For Security:
 - Allows consistency for verifying code changes
 - Makes it more difficult for an attacker to change values across distributed PLC vs centralised HMI
- For Reliability:
 - Better to be done on the PLC for accuracy, efficiency and integrity
- For Maintenance:
 - Coding is easier to understand and transfer between PLCs vs migration of HMI



4. Use PLC flags as integrity checks

Put counters on PLC error flags to capture any math problems.

Security Objective

- Integrity of PLC logic

Target Group

- Integration / Maintenance Service Provider

- For example, if you receive a divide by zero flag – you should investigate
- For Security - Can assist with tracking changes, especially for PLCs without cryptographic checks
- For Reliability - Can assist PLC reliability by picking up programming or IO errors before they cause a larger error

5. Use cryptographic and / or checksum integrity checks for PLC code

Use cryptographic hashes, or checksums if cryptographic hashes are unavailable, to check PLC code integrity and raise an alarm when they change.

Security Objective

- Integrity of PLC logic

Target Group

- Product Supplier
- Integration / Maintenance Service Provider
- Asset Owner

- Can be native functionality or achieve via external configuration management tools (such as Version Dog, Asset Guardian, PAS)
- For Security - Can assist with easily flagging tampered PLC Code
- For Reliability - Can be used for checking if using version controlled PLC code for an integration or manufacturer



6. Validate timers and counters

If timers and counters values are written to the PLC program, they should be validated by the PLC for reasonableness and verify backward counts below zero.

Security Objective

- Integrity of PLC variables

Target Group

- Integration / Maintenance Service Provider
- Asset Owner

- Timers and Counters can be technical preset to any value, use ranges to make sure they are feasible
- For Security - use a validation layer and don't write timers directly to I/O
- For Reliability - by flagging accidental operator input
- For Maintenance - can assist by documenting current timing settings

7. Validate and alert for paired inputs / outputs

If you have paired signals, ensure that both signals are not asserted together. Alarm the operator when input / output states occur that are physically not feasible. Consider making paired signals independent or adding delay timers when toggling outputs could be damaging to actuators.

Security Objective

- Integrity of PLC variables
- Resilience

Target Group

- Product Supplier
- Integration / Maintenance Service Provider

- For Security:
 - can help for PLCs that don't account for paired input signals being asserted at the same time
 - This should flag an operational error, programming error or something malicious might be going on
 - Can help avoid physical damage attack scenarios
- For Reliability:
 - These scenarios may point to a faulty sensor
 - The error may be due to quick toggling start and stop states

8. Validate HMI input variables at the PLC level, not only at HMI

HMI access to PLC variables can (and should) be restricted to a valid operational value range at the HMI, but further cross-checks in the PLC should be added to prevent, or alert on, values outside of the acceptable ranges which are programmed into the HMI.

Security Objective

- Integrity of PLC variables

Target Group

- Product Supplier
- Integration / Maintenance Service Provider

■ For Security:

- HMIs can provide input validation capability, but this can be worked around by an attacker
- Given that PLC protocols are general marketed as “open” protocols, malware can be developed quite easily. If the attacker has access to network traffic, variable mapping can be easily done

9. Validate indirections

Validate indirections by poisoning array ends to catch fence-post errors.

Security Objective

- Integrity of PLC variables

Target Group

- An Indirection is “the use of a value of a register in another register” and there are legitimate uses
- For Security - Most PLCs do not have an end of bounds error flag. This may cause:
 - An indirection to reading the wrong register and the program executing the wrong value
 - An unwanted value or code overwrite
- For Reliability – it can identify non-malicious programming errors

10. Assign designated register blocks by function (read/write/validate)

Assign designated register blocks for specific functions in order to validate data, avoid buffer overflows and block unauthorized external writes to protect controller data.

Security Objective

- Integrity of PLC variables

Target Group

- Product Supplier
- Integration / Maintenance Service Provider

- For Security:
 - Makes it easier to protect data blocks in the controller
 - Assists other security controls like protocol aware firewalls
 - Makes it easier to flag malicious actors
- For reliability:
 - Makes read and writes go faster
 - Network and comms errors on long messages can be found easier
- For Maintenance:
 - Can make it easier to find programming mistakes

11. Instrument for plausibility checks

Instrument the process in a way that allows for plausibility checks by cross-checking different measurements.

Security Objective

- Integrity of I/O values

Target Group

- Product Supplier
- Integration / Maintenance Service Provider

- Two different ways:
 - Compare integrated and time-independent measurements
 - Compare different measurement sources
- For Security – facilitates monitoring for manipulated values
- For Reliability – prevents acceptance or identifies corrupted/wrong measurements as inputs
- For Maintenance – rules out the possible causes of failure more quickly

12. Validate inputs based on physical plausibility

Ensure operators can only input what's practical or physically feasible in the process. Set a timer for an operation to the duration it should physically take. Consider alerting when there are deviations. Also alert when there is unexpected inactivity.

Security Objective

- Integrity of I/O values

Target Group

- Integration / Maintenance Service Provider

- Monitor for expected physical durations and/or monitor expected physical repeating activity
- For Security:
 - Deviations can indicate an actuator was already in the middle of a travel state and may be trying to fake I/O
 - Inactivity alerts facilitate monitoring for stuck or forced constraint values which could indicate tampering
- For Reliability:
 - Deviation give you an alert for broken equipment
 - Inactivity alerts may help flag measurement or control loop failures

13. Disable unneeded / unused communication ports and protocols

PLC controllers and network interface modules generally support multiple communication protocols that are enabled by default. Disable ports and protocols that are not required for the application.

Security Objective

- Hardening

Target Group

- Integration / Maintenance Service Provider

- For Security – Reduce the attack surface and make device vulnerability management easier in future
- For Reliability – Can assist with reducing malformed traffic and reduce potential PLC crashes
- For Maintenance – Reduces complexity and reduces whole of life administration effort

14. Restrict third-party data interfaces

Restrict the type of connections and available data for 3rd party interfaces. The connections and/or data interfaces should be well defined and restricted to only allow read/write capabilities for the required data transfer.

Security Objective

- Hardening

Target Group

- Integration / Maintenance Service Provider

- There are scenarios where the use of third-party data interfaces are important (e.g. long cable runs, large data exchanges)
- For Security:
 - Limit exposure to third party networks and equipment
 - Use data access control mechanisms to limit access
 - Authenticate devices to mitigate spoofing
- For Reliability – limits the ability of a third party to interfere with your process from third party locations or equipment

15. Define a safe process state in case of a PLC restart

Define safe states for the process in case of PLC restarts (e.g., energize contacts, de-energize, keep previous state).

Security Objective

- Resilience

Target Group

- Product Supplier
- Integration / Maintenance Service Provider

- For security – eliminate potential unexpected behaviour
- For reliability – avoid unexpected delays

16. Summarize PLC cycle times and trend them on the HMI

Summarize PLC cycle time every 2-3 seconds and report to HMI for visualization on a graph.

Security Objective

- Monitoring

Target Group

- Integration / Maintenance Service Provider

- For Security – changes in cycle times may be a good indicator that the logic in the system has been changed
- For Reliability – same as security, but for non-malicious use cases



17. Log PLC uptime and trend it on the HMI

Log PLC uptime to know when it's been restarted. Trend and log uptime on the HMI for diagnostics.

Security Objective

- Monitoring

Target Group

- Integration / Maintenance Service Provider

- For Security – most basic attacks cause a restart, tracking this is useful
- For Reliability – PLC restarts are a good diagnostic for monitoring for failures

Tip – if your PLC uses SNMP MIB-2 monitor SNMP OID 1.3.6.1.2.1.1.3



18. Log PLC hard stops and trend them on the HMI

Store PLC hard stop events from faults or shutdowns for retrieval by HMI alarm systems to consult before PLC restarts. Time sync for more accurate data.

Security Objective

- Monitoring

Target Group

- Integration / Maintenance Service Provider

- For Security – logs enabling troubleshooting and to ensure trustworthiness
- For Reliability – Logs are also a good source for debugging if not a malicious actor

19. Monitor PLC memory usage and trend it on the HMI

Measure and provide a baseline for memory usage for every controller deployed in the production environment and trend it on the HMI.

Security Objective

- Monitoring

Target Group

- Integration / Maintenance Service Provider
- Asset Owner

- For security – increased memory could be an indicator that the PLC logic has been modified
- For Reliability – managing total memory consumption to avoid system faults
- For Maintenance – useful for tuning and optimising scan times. Also useful for troubleshooting fault states.



20. Trap false negatives and false positives for critical alerts

Identify critical alerts and program a trap for those alerts. Set the trap to monitor the trigger conditions and the alert state for any deviation.

Security Objective

- Monitoring

Target Group

- Integration / Maintenance Service Provider

- For Security – can be used to identify attack obfuscation

Want to learn more?


<https://www.plc-security.com/>

Follow @securePLC on twitter


1 / 2

Secure PLC Coding Practices: Top 20 List

Version 1.0 (15 June 2021)



- 1. Modularize PLC Code**
Split PLC code into modules, using different function blocks (sub-routines). Test modules independently.
- 2. Track operating modes**
Keep the PLC in RUN mode. If PLCs are not in RUN mode, there should be an alarm to the operators.
- 3. Leave operational logic in the PLC wherever feasible**
Leave as much operational logic e.g., totaling or integrating, as possible directly in the PLC. The HMI does not get enough updates to do this well.
- 4. Use PLC flags as integrity checks**
Put counters on PLC error flags to capture any math problems.
- 5. Use cryptographic and / or checksum integrity checks for PLC code**
Use cryptographic hashes, or checksums if cryptographic hashes are unavailable, to check PLC code integrity and raise an alarm when they change.
- 6. Validate timers and counters**
If timers and counters values are written to the PLC program, they should be validated by the PLC for reasonableness and verify backward counts below zero.
- 7. Validate and alert for paired inputs / outputs**
If you have paired signals, ensure that both signals are not asserted together. Alarm the operator when input / output states occur that are physically not feasible. Consider making paired signals independent or adding delay timers when toggling outputs could be damaging to actuators.
- 8. Validate HMI input variables at the PLC level, not only at HMI**
HMI access to PLC variables can (and should) be restricted to a valid operational value range at the HMI, but further cross-checks in the PLC should be added to prevent, or alert on, values outside of the acceptable ranges which are programmed into the HMI.
- 9. Validate indirections**
Validate indirections by poisoning array ends to catch fence-post errors.
- 10. Assign designated register blocks by function (read/write/validate)**
Assign designated register blocks for specific functions in order to validate data, avoid buffer overflows and block unauthorized external writes to protect controller data.
- 11. Instrument for plausibility checks**
Instrument the process in a way that allows for plausibility checks by cross-checking different measurements.
- 12. Validate inputs based on physical plausibility**
Ensure operators can only input what's practical or physically feasible in the process. Set a timer for an operation to the duration it should physically take. Consider alerting when there are deviations. Also alert when there is unexpected inactivity.




CYBERSECURITY VENDOR POLICIES

Vendor PLC Cybersecurity Integration Requirements

Abstract
The purpose of this document is to define the minimal cybersecurity requirements that third-party vendors shall adhere to prior to connecting their equipment to the ASSET OWNER industrial control system network.


ASSET OWNER




PLC Security
TOP 20 LIST

PLC Top20DownloadRead ThisTranslationsApplication NotesResourcesContactLicense


Videos




July 13, 2022 - Sarah Fluchs
OTCEP Forum 2022 – Securing PLC Code Practices




October 21, 2021 – Vivek Ponnada
BSides Calgary – Top 20 Secure PLC Coding Practices




October 14, 2021 – Vivek Ponnada
NightWatch – Top 20 Secure PLC Coding Practices




November 04, 2021 – Don C. Weber
SANS ICS Concepts – Top 20 Secure PLC Coding Practices




September 06, 2021 – Arnaud Soulié
Secure PLC coding – Practices 6 & 8




August 08, 2021 – Sarah Fluchs & Vivek Ponnada
Def Con ICS Village – Secure Coding Practices for PLCs



July 14, 2021 – Vivek Ponnada
GRIMMCon 0x5 – Utilizing Native Functionality in ICS to improve Security



June 19, 2021 – Vivek Ponnada
AppSec Pacific Northwest Conference – Secure Coding of Industrial Control Systems



June 16, 2021 – Dale Peterson
Unsolicited Response Show – Top 20 Secure PLC Coding Practices

Want to learn more? (cont.)

TOP 20 Secure PLC Coding Practices (SPCP) Tool

Secure PLC Coding Practices ▾

Security Objective ▾

Target Group ▾

1. Product Supplier

2. Integration / Maintenance Service Provider

3. Asset Owner

1 - 3 / 3 < >

Beneficial for ▾

Why?

1. Security

Logs enable troubleshooting in case of an incident. Before a PLC becomes operational, especially after having experienced problems, it is important to ensure it is trustworthy.

2. Security

1. Limit the exposure to 3rd party networks and equipment.
2. Authenticate external devices to prevent spoofing.

1 - 60 / 60 < >

Rank ▴

Secure PLC Coding Practices

Description

Guidance

Example

View Top 20 Secure PLC Coding Practices

1

Modularize PLC Code

Split PLC code into modules, using different function blocks (sub-routines). Test modules independently.

Do not program the complete PLC logic in one place e.g., in the main Organization Block or main routine. Instead, split it into different function blocks (sub-routines) and monitor their execution time and their size in Kb.

Create separate segments for logic that functions independently. This helps in input validation, access control management, integrity verification etc.

Modularized code also facilitates testing and keeping track of the integrity of code modules. If the code inside the module has been meticulously tested, any modifications to these modules can be verified against the hash of the original code, e.g., by saving a hash of each of these modules (when that's an option in the PLC). This way, modules can be validated during the FAT/SAT or if the integrity of the code is in question after an incident.

Gas Turbine logic is segregated into "startup", "Inlet Guide Vanes Control", "Bleed Valve Control" etc. so that you can apply standard logic systematically. This also helps in troubleshooting quickly if there were to be a security incident.

Custom function blocks that are tested rigorously can be re-used without alteration (and alerted if change attempts are made) and locked against abuse/misuse with a password/digital signature.

[Top 20 Secure PLC Coding Practices](#)

1 - 20 / 20 < >

Security Objective

All

Integrity of PLC Io...

Integrity of PLC...

Monitoring

Hardening

Integrity of I/O valu...

Resilience

How to contribute?

Core team You?

Top 20 Secure PLC Environment Practices

Top 20 Secure PLC Coding Practices v2.0

Demonstrators / virtual secure PLC Coding playground

Translations to other languages

PLC Coding practices in purchasing specifications

Commenters You?

Comments & new proposed practices

Your idea?

You?

Collaborations

MITRE CWE

Engineers' associations

PLC vendors

PLC user groups

Supporters You?

Articles, podcasts...

Trainings



Thank you,
Questions?

