

# Securing the bridge between the cyber and physical worlds: Cybersecurity for PLCs

David Formby, PhD Cofounder and CEO/CTO Fortiphyd Logic

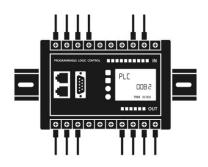
### Background

#### **Programmable Logic Controllers (PLCs)**

#### Cyber/IT









Bridge between cyber and physical

#### Insecure by design Most attention is paid to network

#### Outline

- Bio
- Threats to PLCs
- Secure PLC Coding Practices
- PLC Program Anomaly Detection



#### Bio – David Formby

- PhD Electrical & Computer Engineering, Georgia Tech
  - PLC ransomware worm
  - PLC anomaly detection
  - Open source ICS security simulation
- Dozens of ICS-CERT vulnerabilities
- Cofounder and CEO/CTO of Fortiphyd Logic
  - Network monitoring
  - PLC endpoint security
  - ICS security training





### Threats to PLCs



#### Threat Timeline for PLCs (Purdue Level 1)

	Ukraine attack 2015	PLC Ransomware (POC) 2018	Oldsmar 2021
Stuxnet 2010	<b>CRASHOVERRIDE</b> 2016	<b>TRITON</b> 2018	Rockwell Hardcoded Key Vuln 2021



### Stuxnet (2010)



#### • Target

- Iranian nuclear program
- Key points
  - Jumped "air-gap"
  - Reprogrammed PLC, and hid the changes
- Effects
  - Destroyed up to 1000 centrifuges



### Ukraine Attack (2015)



#### • Target

- Ukrainian power distribution companies
- Key points
  - Stole operator remote credentials
  - Used existing HMI
  - Complex amplified attack
    - DoS telephone service
    - Bricked RTUs, disabled UPS
- Effects

FORTIPHY

• 225,000 customers lost power

### CRASHOVERRIDE (2016)



#### • Target

- Ukrainian transmission substation
- Key points
  - Highly modular, complex malware
  - Deep understanding of ICS protocols
- Effects
  - Section of capital city, Kiev, lost power for one hour
  - Large-scale testing of malware not serious attempt

### PLC Ransomware (Proof-of-concept)



- Georgia Tech research project
- Lack of ICS attacks because financial motivation not secure
  - Ransomware could be effective monetization of insecurity
- Worm compromised one PLC and spread to others, locking out operators

### Trisis/Triton/HatMan (2018)



- Target
  - Saudi Arabian oil facility
- Key points
  - Reverse engineered programming software
  - PLC "rootkit" remains even if switch is in Run Mode
  - Only detected because of attacker programming mistake
- Effect
  - Emergency shutdown and long debug time

#### Oldsmar



#### • Target

- Small water utility in Florida
- Key points
  - Remote access to HMI
  - Credentials compromised
  - Attacker just played around on HMI
- Effect
  - Changed setpoint to dangerous levels, but operator immediately corrected

### Rockwell Hardcoded Key Vulnerability



- Vulnerable PLCs
  - Entire Logix line of PLCs
- Key Points
  - Hard coded key vulnerability
  - Anyone on network with key can reprogram PLC
  - Official response no patch in sight
- Effect
  - PLCs are still insecure by design

## Top 20 Secure PLC Coding Practices



#### Motivation

- Computer programming has well established guidelines for "secure coding" to mitigate vulnerabilities and produce reliable software
  - Software development life cycle (SDLC)
  - OWASP Secure Coding Practices (web applications)
  - Secure Coding Guidelines for Java SE
  - Microsoft Security Development Lifecycle
- PLC programming lacks similar standards

#### Secure PLC Coding Practices

- Presentation at S4 Conference by Jake Brodsky
  - Basic tips on what PLC programmers can do to add security
- First attempt at formalizing the tips
  - Organizers
    - Admeritia (Vivek Ponnada and Sarah Fluchs)
    - S4 Conference (Dale Peterson)
    - Other volunteers
  - June 2021
  - Top 20 Secure PLC Coding Practices



#### License

Copyright (c) 2021 admeritia GmbH, Langenfeld/Rheinland, Germany Permission is hereby granted, free of charge, to any person obtaining a copy of "Top 20 Secure PLC Coding Practices" and associated documentation files, to deal in the "Top 20 Secure PLC Coding Practices" without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the "Top 20 Secure PLC Coding Practices", and to permit persons to whom the "Top 20 Secure PLC Coding Practices" is furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all copies or substantial portions of the "Top 20 Secure PLC Coding Practices".

THE "Top 20 Secure PLC Coding Practices" IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE "Top 20 Secure PLC Coding Practices" OR THE USE OR OTHER DEALINGS IN THE "Top 20 Secure PLC Coding Practices".

#### Security Objectives

- Integrity (Practices 1-12)
  - PLC Logic Detecting/preventing unauthorized changes to program
  - PLC variables Detecting/preventing unauthorized changes to variables
  - IO values Detecting/preventing unauthorized changes to IO values
- Hardening (Practices 13,14)
  - Make initial compromise more difficult
- Resilience (Practices 15,7)
  - Assuming compromise, mitigate threat to continuing mission/operation
- Monitoring (Practices 16-20)
  - Detect incident as soon as possible to mitigate damage

#### Benefits

- Security
  - Mitigating threat from intentional attackers
- Reliability
  - Mitigating threat from accidents
- Maintenance
  - Making code easier to debug and maintain



### Additional References

- MITRE ATT&CK for ICS
  - Formal descriptions of attacks on ICS and mitigations
    - Tactics, Techniques, Procedures, Mitigations
- ISA 62443
  - 3-3 System security requirements and security levels
  - 4-1 Secure product development lifecycle requirements
  - 4-2 Technical security requirements for IACS components
- MITRE CWE
  - Common weakness enumeration



#### Limitations

- Secure PLC Coding Practices are only one layer of defense
  - Good network security and monitoring still necessary
- Most PLCs are still insecure by design
- Secure practices only make it harder for the attacker, not impossible
  - Many can be worked around by attacker with complete control of PLC
  - Monitoring is only useful if someone watches it, or gets alerts
- Only a first pass, still looking for user feedback and refinement

#### 1. Modularize PLC Code

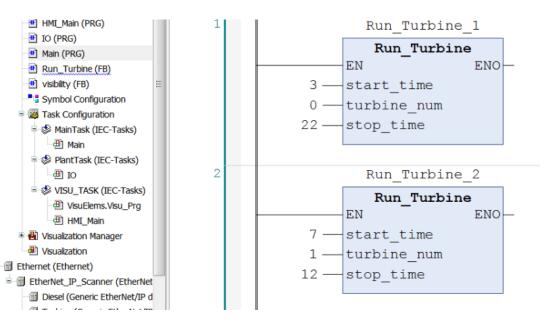
- Break PLC logic down into function blocks or subroutines that are:
  - Re-usable
  - Testable
  - Independent
- For each FB or subroutine:
  - Test thoroughly, ranges of inputs
  - Record execution time, size in rungs or kB, checksum/signature if available
  - Lock with password/certificate if possible and it makes sense
  - Track changes

**Integrity of PLC Logic** 

#### 1. Modularize PLC Code - Example

FORTIPHY

- Gas turbine startup sequence
  - Function block for general turbine, instantiate for each turbine
    - Made of smaller function blocks for each component



#### 2. Track operating modes

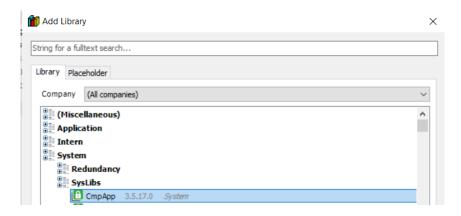
- Most PLCs have RUN and PROGRAM modes, and only can be programmed with normal software in PROGRAM mode
- Keep PLCs in RUN mode as much as possible
- Display current mode on HMI
- Alarm on mode changes, and continuously for PROGRAM mode
  - Exception extended maintenance windows, extra precautions elsewhere

**Integrity of PLC Logic** 

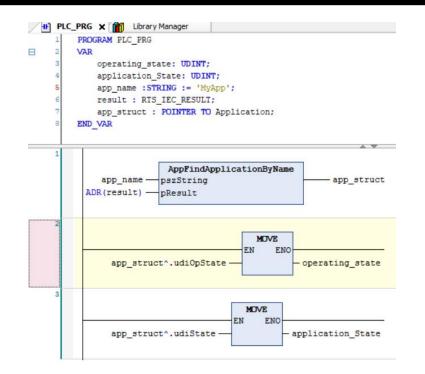
#### 2. Track operating modes - Example

On	ine Debug Tools Window Help	
<b>O</b> Ş	Login Alt+F8	「日白日日日日日
СŞ	Logout Ctrl+F8	
	Create boot application	Task 🕛 Run_Turbin
-	Download	💿 Details 🔤 Placeho
	Online Change	
	Source download to connected device	rt Software Solutions Gmb
	Multiple Download	: Software Solutions Gmb
	Reset warm	5 - Smart Software Solutio
	Reset cold	4.4.0.0 (3S - Smart Softw
	Reset origin	n, 4.4.0.0 (3S - Smart Sof
	Simulation	
	Security •	h, 3.5.14.0 (System)
	Operating Mode	n Debug
	Standard, 3.5.14.0 (System)	Locked
	🖲 🗀 Bistable Function Blocks	Operational

Online	De	ebug	Tools	Window	Help		
肥肥	Þ	Star	t			F5	
		Stop	)			Shift+F8	
J Devic		Sing	le Cycle			Ctrl+F5	



#### 2. Track operating modes - Example



Name	Туре	Inherited from	Address	Initial	Comment
OS_NONE	UDINT			16#0	
OS_PROGRAM_LOADED	UDINT			16#1	
OS_DOWNLOAD	UDINT			16#2	
OS_ONLINE_CHANGE	UDINT			16#4	
OS_STORE_BOOTPROJECT	UDINT			16#8	
OS_FORCE_ACTIVE	UDINT			16#10	
OS_EXCEPTION	UDINT			16#20	
OS_RUN_AFTER_DOWNLOAD	UDINT			16#40	
OS_STORE_BOOTPROJECT_ONLY	UDINT			16#80	

Name	Туре	Inherited from	Address	Initial	Comment
AS_NONE	UDINT			16#0	Application states
AS_RUN	UDINT			16#1	
AS_STOP	UDINT			16#2	
AS_DEBUG_HALT_ON_BP	UDINT			16#3	
AS_DEBUG_STEP	UDINT			16#4	
AS_SINGLE_CYCLE	UDINT			16#5	

#### 2. Track operating modes - Example

Name		Description	Context	
	Add Event Handler			>
	Event	StopDone		~
	Function to call			
	Scope	Application OPOUs		
	Implementation language	Structured Text (ST)		~
	Description	Called after application stops. Context=Con Debugging=Disabled	mmunication task.	

#### 3. Leave operational logic in the PLC

- HMIs have some programmability
- Leave safety and key operational logic in PLC
  - Summarizing/totaling logic
  - Logic to enable/disable buttons (timers, counters...)
  - Thresholds to trigger alarms

**Integrity of PLC Logic** 



#### 5. Use crypto/checksum integrity checks

- Cryptographic hashes, checksums, audit values, signatures
  - Methods for "fingerprinting" PLC program and configuration
  - Available on most PLCs, some more secure than others
- Log checksum/hash as part of SAT, documentation of final product
  - Alarm when it changes

**Integrity of PLC Logic** 

### 5. Use crypto/checksum integrity checks

#### Package Checksums

	X	y Manager 🛛 📷 Task Configuration 🖉 Device 🖉 🚱 Security Screen 🗙
Package Details License Agreement Installation Log		User Profile and Certificate Selection
Name	User	dform
CODESYS CANopen	Project	Digital Signature
Version 4.0.0.0	Devices	Issued for Issued by Valid from Valid until Thumbprint
Checksum		
62A187B043E9167132FC05EF2BE19B93E58CD958		Project File Decryption
Vendor CODESYS GmbH		Issued for Issued by Valid from Valid until Thumbprint
Copyright		Security Level
Copyright (c) 2017-2021 CODESYS Development GmbH		Activate the Use of Certificates for Enhanced Security
Description		Enforce encrypted communication
This package contains all necessary components for CODESYS		Enforce encryption of project files
CANopen.		☑ Enforce signing of project files
Installation date		☐ Enforce encryption of downloads, online changes and boot applications
Friday, October 1, 2021 1:17 PM		Enforce signing of downloads, online changes and boot applications
		Enforce signing of compiled libraries
		✓ Enforce timestamping of signed compiled libraries
		Timestamping server
More >>		а.
Close	https://help.codesys.com/api	i-content/2/codesys/3.5.12.0/en/ cds encrypting signing wi

#### **Signing Project Files with Certificates**

FORTIPHY LOGIC

### 7 / 11 / 12 - Plausibility

- Paired IO physically cannot happen at the same time
  - Start/stop, forward/reverse, open/close
- Cross check multiple sources of sensor data for related plausibility
  - Compare integrated/summed measurements with instantaneous values
    - Flow rates with volumes
  - Compare different sensors that should be related
    - Valve open & flow rate > 0 / valve closed & flow rate = 0
- Only allow inputs that are physically possible/safe
  - Set a timer for longer operations to verify it completes in physically plausible amount of time

Integrity of PLC Variables Resilience

FORTIPHY

#### 7. Validate and alert for paired IO



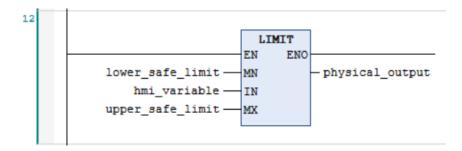


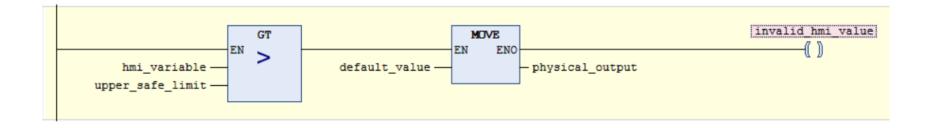
#### 8. Validate HMI inputs at PLC

- Values should be limited to safe ranges **not just** at HMI, but also in the PLC logic
- Limiting ranges in the HMI only prevents accidental attacks
  - Moderately skilled attackers can bypass HMI checks
- If invalid value is received in PLC, log it
  - Use a default safe value
  - Use last valid value
  - Limit to closest max

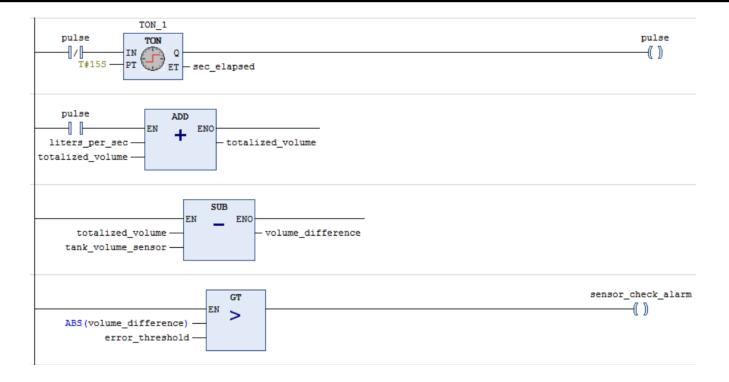
**Integrity of PLC Variables** 

#### 8. Validate HMI inputs at PLC

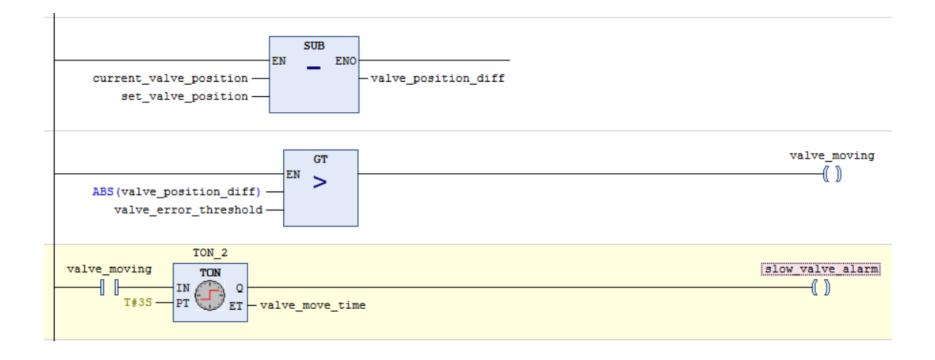




### 11. Instrument for plausibility checks



#### 12. Validate inputs for physical plausibility



# 13. Disable unneeded ports and protocols

- PLCs, especially newer ones, come with multiple protocols supported
- Disable ports and protocols that are not needed
  - Web server configuration
  - SNMP, Telnet, FTP
  - Modbus, OPC UA
- Develop data flow diagram showing required communication for PLCs and physical ports

Hardening

# 13. Disable unneeded ports and protocols -Examples

- Disable OPC UA if not used
- Don't add protocols unless you are using them
- Configure host firewall to only allow needed ports for CODESYS and whatever protocols

### 15. Define a safe process state for reboots

- PLCs can unexpectedly reboot
  - Accidental power loss
  - Exploit attempt
- In case of PLC reboot, start into a safe output state
  - Valves closed/open, motors on/off

Resilience

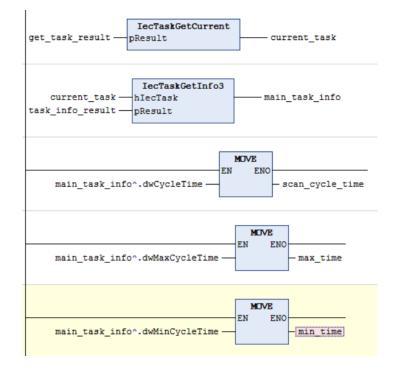
FORTIPHY

## 16. Summarize cycle times and trend them

- Program execution times usually measured as system variable
- Program execution times are relatively consistent unless
  - PLC program changes
  - Physical process changes significantly
  - Network changes significantly
- Monitor average, max, and min scan cycle times
  - Trend on HMI to visually detect unusual program behavior
- Automatically detect changes
  - Fortiphyd's LogicWatch Pro

# 16. Summarize cycle times and trend them

brary Plac	ceholder	
Company	(All companies)	
6	CmpEventMgr Implementation 3.5.17.0 System	
6	CmpEventMgr Interfaces * System	
6	CmpEventMgr 3.5.17.0 System	
6	CmpHilscherCIFX 3.5.17.0 System	
6	CmpIecTask 3.5.17.0 System	
6	CmpIecVarAccess Implementation 3.5.17.0 System	
6	CmpIecVarAccess Interfaces * System	
6	CmpIecVarAccess 3.5.17.0 System	
<	CmpIoMgrImplementation 3.5.17.0 System	>

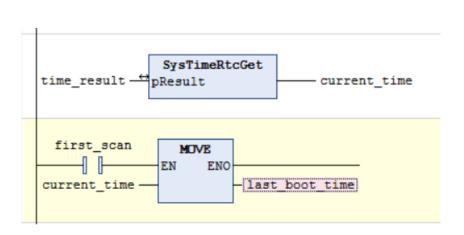


### 17. Log uptime and trend it

- PLC reboots can indicate maintenance problems or exploit attempts
- Monitor time since last reboot to detect reboots
  - In PLC logic
  - From network using SNMP

### 17. Log uptime and trend it

brary Plac	eholder	
Company	(All companies)	~
e	SysSocketAsync 3.5.17.0 System	^
E	SysTarget 3.5.17.0 System	
e	SysTask 3.5.17.0 System	
(fi	SysTime 3.5.17.0 System	
(ê	SysTimeCore 3.5.17.0 System	
lê	SysTimeLook 3.5.17.0 System	
lê	SysTimer 3.5.17.0 System	
lê	SysTimeRtc 3.5.17.0 System	
	SysTypes Interfaces * System	~
<		>



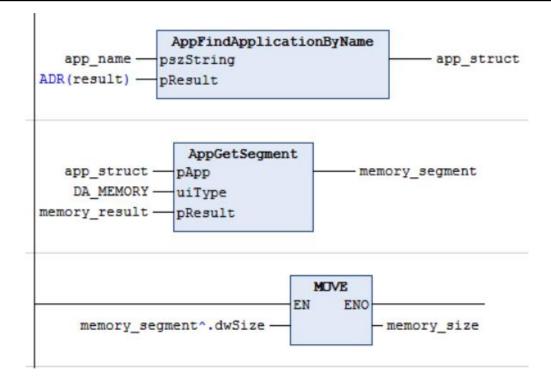
### 18. Log hard stops and trend them

- Hard stops from faults can indicate maintenance problems or attacks
- Log hard stops for investigation before continuing operations
  - Accurate timestamping important for debugging
  - Log which error codes

### 19. Monitor memory usage and trend it

- Two types of memory
  - Storage
  - RAM
- Change in program memory indicates change in program
  - More or less running code
- Track what is normal and alert on thresholds

### 19. Monitor memory usage and trend it



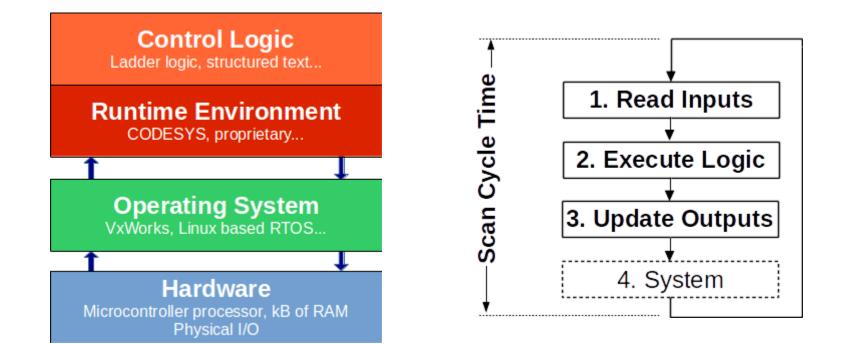
# PLC Anomaly Detection



### Motivation

- Vendors cannot secure their PLCs on their own
- Give operators a tool to add strong security themselves
- Leverage the limitations of PLCs as a strength
  - Single purpose programming -> predictable program execution
  - Slow processors -> changes in programming more noticeable
  - Limited memory -> incapable of storing large amounts of data
- Result
  - "Temporal Execution Behavior for Host Anomaly Detection in Programmable Logic Controllers" IEEE TIFS 2019
  - Patent, LogicWatch Pro

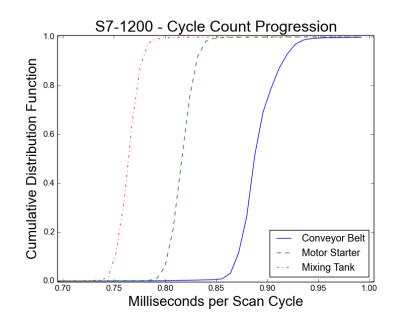
### **Change Detection**





# Change Detection

- Change in PLC program results in change in execution time
- Small change not noticeable in one scan cycle
  - Over thousands of cycles, accumulates into detectable change
- Tested across 3 main vendors
  - Rockwell, Siemens, Modicon
- Single instruction change was detectable on sample programs

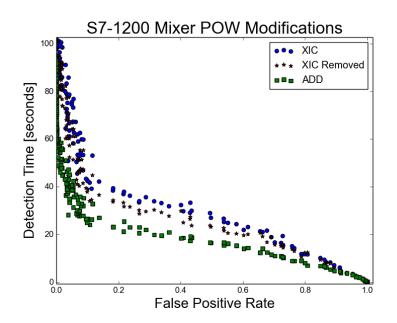


# Proof of Work

- Limitations
  - Attacker with knowledge of algorithm can bypass it
  - Harder to fake than static value, but still feasible to repeat previous values
- Proof of work (POW) function
  - Computationally expensive to solve, but easy to verify
  - Give PLC an "alibi", to prove it was busy doing POW and did not have time to execute anything else

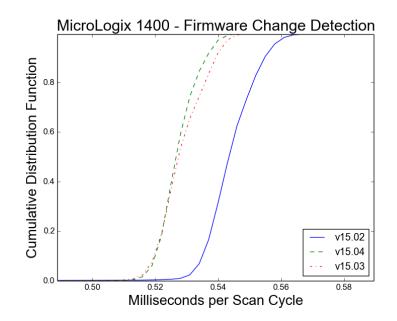
# Proof of Work

- Discrete logarithm
  - $x^y \mod p = z$
  - To solve:
    - General approach brute force multiply base over and over again
      - O(n), n size of modulus group
    - Best approaches O(sqrt(n))
  - To verify:
    - Exponentiation by squaring -O(log(n))



### Firmware Modification

- Changes to underlying firmware contribute to program execution time
- Different firmware versions were found to be detectable changes



# White Box Modeling

- Rare branches of code not executed very often
- Some PLCs list instruction execution time
  - With full "white box" knowledge of PLC program, can estimate execution time of rare branches
- Results
  - White box not as accurate, but viable approach

### Conclusion

- Threats to PLCs are increasing, but they are still insecure by design
- PLC programmers can add security
  - PLC programming
    - Integrity Don't trust anything from the network
    - Hardening Remove unnecessary functionality
    - Resilience Assume attack/misuse, try to fail safe
    - Monitoring Trend KPI of PLC to detect incidents
  - PLC program anomaly detection
    - Program execution time and proof of work functions

David Formby dformby@fortiphyd.com

Twitter: @fortiphyd LinkedIn



### References

- Fortiphyd Training Grounds Course on Secure PLC Coding
  - <u>https://fortiphyd.talentlms.com/catalog/info/id:166</u>
- Top 20 Secure PLC Coding Practices
  - <u>https://plc-security.com/</u>
- ISA 62443 Standards
  - <u>https://www.isa.org/standards-and-publications/isa-standards/find-isa-standards-in-numerical-order</u>
- MITRE ATT&CK for ICS
  - <u>https://collaborate.mitre.org/attackics/index.php/Main\_Page</u>
- MITRE CWE
  - <u>https://cwe.mitre.org/</u>