Sichere SPS-Codierungspraktiken: Top-20 Liste

Version 1.0 (15. Juni 2021/Deutsch 28.März 2024)

PLC Security TOP 20 LIST

1. SPS-Code modularisieren

Teilen Sie SPS-Code in Module auf, indem Sie verschiedene Funktionsbausteine (Unterroutinen) verwenden. Testen Sie Module selbstständig.

2. Betriebsmodi verfolgen

Halten Sie die SPS im RUN-Modus. Wenn sich die SPS nicht im RUN-Modus befindet, sollte ein Alarm für die Bediener ausgegeben werden.

3. Belassen Sie die Betriebslogik in der SPS, wo immer dies möglich ist

Belassen Sie so viel Betriebslogik wie z.B. Summieren oder Integrieren direkt in der SPS. Das HMI erhält nicht genügend Updates, um dies gut zu tun.

4. SPS-Flags als Integritätsprüfung verwenden

Setzen Sie Zähler auf SPS-Fehler-Flags, um mathematische Probleme zu erfassen.

5. Kryptografische und/oder Prüfsummen-Integritätsprüfungen für SPS-Code verwenden

Verwenden Sie kryptografische Hashes oder Prüfsummen, wenn kryptografische Hashes nicht verfügbar sind, um die Integrität des SPS-Codes zu überprüfen und einen Alarm auszulösen, wenn sie sich ändern.

6. Validieren von Timern und Zählern

Wenn Zeitschaltuhren und Zählerwerte in das SPS-Programm geschrieben werden, sollten diese von der SPS auf Plausibilität überprüft werden und Rückwärtszählungen unter Null verifizieren.

7. Validieren und Alarmieren für gekoppelte Ein-/Ausgänge

Wenn Sie gekoppelte Signale haben, stellen Sie sicher, dass beide Signale nicht zusammen bestätigt werden. Alarmieren Sie den Bediener, wenn Ein-/Ausgangszustände auftreten, die physikalisch nicht realisierbar sind. Erwägen Sie, gekoppelte Signale unabhängig zu machen oder Verzögerungstimer hinzuzufügen, wenn das Umschalten der Ausgänge für Aktoren schädlich sein könnte.

8. HMI-Eingangsvariablen auf SPS-Ebene validieren, nicht nur auf HMI-Ebene

Der HMI-Zugriff auf SPS-Variablen kann (und sollte) auf einen gültigen Betriebswertebereich am HMI beschränkt werden, aber weitere Gegenprüfungen in der SPS sollten hinzugefügt werden, um Werte außerhalb der akzeptablen Bereiche, die in der HMI programmiert sind, zu verhindern oder zu warnen.

9. Überprüfen von Dereferenzierungen

Überprüfen Sie Dereferenzierungen, indem Sie Array-Enden vergiften, um Fehler an Zaunpfosten abzufangen.

10. Zugewiesene Registerblöcke nach Funktion (Lesen/Schreiben/Validieren)

Weisen Sie bestimmten Funktionen bestimmte Registerblöcke zu, um Daten zu validieren, Pufferüberläufe zu vermeiden und unbefugte externe Schreibvorgänge zu blockieren, um Steuerungsdaten zu schützen.

11. Instrument zur Plausibilitätsprüfung

Instrumentieren Sie den Prozess so, dass Plausibilitätsprüfungen möglich sind, indem verschiedene Messungen abgeglichen werden.

12. Validieren Sie Eingaben auf der Grundlage physikalischer Plausibilität

Stellen Sie sicher, dass Bediener nur das eingeben können, was im Prozess praktisch oder physikalisch machbar ist. Stellen Sie einen Timer für einen Vorgang auf die Dauer ein, die er physisch dauern sollte. Erwägen Sie eine Warnung, wenn es Abweichungen gibt. Auch bei unerwarteter Inaktivität warnen.

13. Deaktivieren Sie nicht benötigte/ungenutzte Kommunikationsports und -protokolle

SPS-Steuerungen und Netzwerkschnittstellenmodule unterstützen in der Regel mehrere Kommunikationsprotokolle, die standardmäßig aktiviert sind. Deaktivieren Sie Ports und Protokolle, die für die Anwendung nicht erforderlich sind.

14. Datenschnittstellen von Drittanbietern einschränken

Schränken Sie die Art der Verbindungen und die verfügbaren Daten für 3rd-Party-Schnittstellen ein. Die Verbindungen und/oder Datenschnittstellen sollten klar definiert und so eingeschränkt sein, dass nur Lese-/Schreibfähigkeiten für die erforderliche Datenübertragung möglich sind.

15. Definieren Sie einen sicheren Prozesszustand im Falle eines SPS-Neustarts

Definieren Sie sichere Zustände für den Prozess bei SPS-Neustarts (z. B. Kontakte einschalten, stromlos schalten, vorherigen Zustand beibehalten).

16. Fassen Sie die SPS-Zykluszeiten zusammen und trenden Sie sie auf dem HMI

Fassen Sie die SPS-Zykluszeit alle 2-3 Sekunden zusammen und melden Sie sie an HMI zur Visualisierung in einem Diagramm.

17. Protokollieren Sie die SPS-Betriebszeit und trenden Sie sie auf dem HMI

Protokollieren Sie die Betriebszeit der SPS, um zu wissen, wann sie neu gestartet wurde. Trend- und Protokollverfügbarkeit auf dem HMI für die Diagnose.

18. Protokollieren Sie SPS-Hardstopps und trenden Sie sie auf dem HMI

Speichern Sie SPS-Hard-Stop-Ereignisse von Fehlern oder Abschaltungen für den Abruf durch HMI-Alarmsysteme, um sie vor dem Neustart der SPS zu konsultieren. Zeitsynchronisierung für genauere Daten.

19. Überwachen Sie die SPS-Speicherauslastung und zeigen Sie einen Trend auf dem HMI

Messen und stellen Sie eine Baseline für die Speicherauslastung für jeden Controller bereit, der in der Produktionsumgebung eingesetzt wird, und stellen Sie einen Trend auf dem HMI bereit.

20. Fangen Sie falsch negative und falsch positive Ergebnisse für kritische Warnungen ab

Identifizieren Sie kritische Warnungen und programmieren Sie eine Falle für diese Warnungen. Legen Sie den Trap fest, um die Auslösebedingungen und den Alarmstatus für Abweichungen zu überwachen.

1. SPS-Code modularisieren

Teilen Sie SPS-Code in Module auf, indem Sie verschiedene Funktionsbausteine (Unterroutinen) verwenden. Testen Sie die einzelnen Module .

Sicherheitsziel	Zilegruppe
Integrität der SPS-Logik	Produktlieferant /Integrator

Anleitung

Programmieren Sie nicht die komplette SPS-Logik an einer Stelle, z.B. im Hauptorganisationsbaustein oder in der Hauptroutine. Teilen Sie es stattdessen in verschiedene Funktionsblöcke (Unterprogramme) auf und überwachen Sie deren Ausführungszeit und ihre Größe in Kb.

Erstellen Sie separate Segmente für Logik, die unabhängig voneinander funktioniert. Dies hilft bei der Eingabevalidierung,
der Zugriffskontrolle, der Integritätsüberprüfung usw.

Modularisierter Code erleichtert auch das Testen und Verfolgen der Integrität von Codemodulen. Wenn der Code im Inneren des Moduls akribisch getestet wurde, können alle Änderungen an diesen Modulen mit dem Hash des Originalcodes verglichen werden, z. B. durch Speichern eines Hashs jedes dieser Module (wenn dies eine Option in der SPS ist). Auf diese Weise können Module während des FAT/SAT oder wenn die Integrität des Codes nach einem Vorfall in Frage gestellt wird, validiert werden.

Beispiel

Die Logik der Gasturbine ist in "Inbetriebnahme", "Steuerung der Einlassleitschaufeln", "Steuerung des Entlüftungsventils" usw. unterteilt, so dass Sie die Standardlogik systematisch anwenden können. Dies hilft auch bei der schnellen Fehlerbehebung, wenn es zu einem Sicherheitsvorfall kommen sollte.

Benutzerdefinierte Funktionsbausteine, die auf Herz und Nieren geprüft werden, können ohne Änderungen wiederverwendet werden (und bei Änderungsversuchen gewarnt werden) und mit einem Passwort/einer digitalen Signatur gegen Missbrauch/Missbrauch gesperrt werden.

Begründung

Vorteilhaft für	Begründung
Sicherheit	Erleichtert die Erkennung neu hinzugefügter Codeteile, die bösartig sein könnten. Hilft bei der Standardisierung der Logik, der Konsistenz und der Sperrung gegen unbefugte Änderungen.
Zuverlässigkeit	Hilft bei der Steuerung des Programmablaufs und vermeidet Schleifen, die dazu führen können, dass die Logik nicht richtig reagiert oder abstürzt.
Instandhaltung	Modularer Code ist nicht nur einfacher zu debuggen (Module können unabhängig voneinander getestet werden), sondern auch einfacher zu warten und zu aktualisieren. Die Module können auch für zusätzliche SPS verwendet werden, so dass gemeinsamer Code in separaten SPSen verwendet und identifiziert werden kann. Dies kann dem Wartungspersonal helfen, gängige Module bei der Fehlersuche schnell zu erkennen.

Referenzen

Standard / Framework	Mapping
MITRE ATT&CK for ICS	**Taktik:** TA 0104 Execution **Technik: T0889:** Program Organisation Units
ISA 62443-3-3	**SR 3.4**: Software- und Informationsintegrität
ISA 62443-4-2	**CR 3.4**: Software- und Informationsintegrität
ISA 62443-4-1	**SI-2**: Sichere Codierungsstandards
MITRE CWE	CWE-1120: Übermäßige Codekomplexität CWE-653: Unzureichende Kompartimentierung

2. Betriebsmodi überwachen

Halten Sie die SPS im RUN-Modus. Wenn sich die SPS nicht im RUN-Modus befindet, sollte ein Alarm für die Bediener ausgegeben werden.

Sicherheitsziel	Zilegruppe
Integrität der SPS-Logik	Integrator / Wartungsdienstleister /Asset Owner

Anleitung

Wenn sich SPSen nicht im RUN-Modus befinden (z. B. PROGRAM-Modus), kann ihr Code geändert werden. Daher ist der RUN-Modus zu überwachen. Einige SPS verfügen über eine Prüfsumme, um bei Codeänderungen zu warnen, aber wenn dies nicht der Fall ist, gibt es zumindest einen indirekten Hinweis auf ein potenzielles Problem bei der Verfolgung von Betriebsmodi:

- Wenn sich die SPS nicht im RUN-Modus befinden, sollten die Bediener alarmiert werden. Wenn sie wissen, dass jemand an diesem Kontrollsystem arbeiten soll, können sie den Alarm quittieren und weitermachen.
- Das HMI sollte so konfiguriert werden, dass der Bediener gegen Ende der Schicht erneut über den vorhandenen Alarm informiert werden.
- Ziel sollte es sein, den Überblick über alle Mitarbeiter oder Auftragnehmer in der Anlage zu behalten, die Arbeiten ausführen, die sich auf den Prozess auswirken könnten.

Ausnahmefall: Wenn sich die Anlage in einer Test- oder Entwicklungsphase befindet, sollten Sie diesen Alarm deaktivieren, aber die Anlage sollte von höheren Ebenen des Netzwerks isoliert werden.

Beispiel

Verfügt die SPS nicht über einen Hardware-Schalter zum Wechseln von Betriebsarten, empfiehlt es sich, zumindest Software-Mechanismen zu nutzen, die das Ändern von SPS-Code einschränken können, z.B. einen Passwortschutz in Engineering-Software zum Lesen und Schreiben von SPS-Code.

Begründung

Vorteilhaft für	Begründung
Sicherheit	Die Betriebsart (Ausführen / Bearbeiten / Schreiben; bei Allen Bradley SPSen: RUN / PROGram / REMote) bestimmt, ob die SPS manipuliert werden kann. Befindet sich der Schlüsselschalter im Zustand REMote, ist es technisch möglich, Änderungen am SPS-Programm über die Kommunikationsschnittstellen vorzunehmen, auch wenn die SPS läuft.
Zuverlässigkeit	/
Instandhaltung	/

Referenzen

Standard / Framework	Mapping
MITRE ATT&CK for ICS	**Taktik**: TA0107 Inhibit Response Function **Technik :** T0858 – Change Operating Mode
ISA 62443-4-1	**SI-1**: Überprüfung der Sicherheitsimplementierung

3. Belassen Sie die Betriebslogik in der SPS, wo immer dies möglich ist

Belassen Sie so viel Betriebslogik wie z.B. Summieren oder Integrieren direkt in der SPS. Das HMI erhält nicht genügend Updates, um dies gut zu tun.

Sicherheitsziel	Zilegruppe
Integrität der SPS-Logik	Produktlieferant / Integrator / Wartungsdienstleister /Asset Owner

Anleitung

HMIs bieten ein gewisses Maß an Codierungsfunktionen, die ursprünglich darauf abzielten, Bedienern bei der Verbesserung der Visualisierung und Alarmierung zu helfen, die einige Programmierer eingesetzt haben, um Code zu erstellen, der lieber in der SPS verbleiben sollte, um vollständig und überprüfbar zu bleiben.

Wenn Sie die Werte so nah wie möglich am Feld berechnen, werden diese Berechnungen genauer. Das HMI erhält nicht genügend Updates, um gut zu summieren / zu integrieren. Außerdem gibt es immer eine Latenz zwischen HMI und SPS. Wenn sich der Code in der SPS befindet und ein HMI neu gestartet wird, kann es immer Totalisatoren/Zählungen von einer SPS empfangen.

Zu vermeidender HMI-Code ist insbesondere alles, was mit Sicherheitsfunktionen wie Verriegelungen, Timer, Holds oder Freigaben zu tun hat.

Für die Analyse von Prozessdatenwerten im Zeitverlauf ist ein Prozessdatenhistorian die bessere Wahl als das HMI. Verwenden Sie Abfragen in einer Prozesshistorian-Datenbank, um summierte Werte (über eine Periode, über eine Charge, über einen Prozesszyklus) mit lokal aggregierten Summen in der SPS-Logik zu vergleichen. Die Warnung bei einer größeren Varianz kann durch Unterschiede in der Datengranularität erklärt werden.

Beispiel

- Code zum Festlegen von Bedingungen zum Aktivieren/Deaktivieren von Schaltflächen: Aktivierungs-/Deaktivierungsaktionen sollten auf der SPS-Ebene gesteuert werden, andernfalls können Aktionen auf dem HMI (oder über das Netzwerk) in der SPS ausgeführt werden, obwohl sie die (beabsichtigten) Bedingungen nicht erfüllen.
- Timer, um dem Bediener Aktionen zu ermöglichen (Verzögerungstimer für aufeinanderfolgende Motorstarts, Timer für Ventile geschlossen/offen oder Motor gestoppt) sollte nicht auf der HMI-Schicht, sondern in der SPS platziert werden, die diesen Motor/ dieses Ventil steuert.
- Schwellenwerte für Alarme müssen Teil der SPS-Codes sein, obwohl sie auf HMIs angezeigt werden.
- Wassertank mit wechselndem Volumen: Die SPS, die den Durchfluss in und aus dem Tank steuert, kann Einfaches Summieren des Volumens (und Kreuzvalidierung von Summen). Das HMI könnte dies auch, müsste aber zuerst die Werte von der SPS

erhalten. Diese Werte benötigen genaue Zeitstempel, um im Falle einer Latenz korrekte Summen zu erhalten, oder sie können Werte verfehlen, wenn das HMI neu gestartet wird.

Begründung

Vorteilhaft für	Begründung
Sicherheit	1. Ermöglicht Konsistenz bei der Überprüfung von Codeänderungen. Die HMI-Codierung hat ihre Änderungskontrolle außerhalb der SPS, in der Regel nicht mit der gleichen Strenge (insbesondere in der Bau- und Inbetriebnahmephase), so dass die Systembesitzer keinen vollständigen Überblick haben und sogar wichtige Überlegungen verloren gehen. HMIs enthalten keine "erzwungenen Signale" oder geänderten Wertelisten wie SPS oder SCADAs, so dass Änderungen auf HMI-Ebene schwieriger zu erkennen sind und praktisch unmöglich sind, Teil eines Berechtigungsänderungsmanagementplans zu sein. 2. Für einen Angreifer ist es schwieriger, Summen zu manipulieren, die auf viele SPS verteilt sind, als Summen, die alle im HMI berechnet werden. 3. Wenn sich ein Teil der Aktivierungs-/Deaktivierungsfunktionen nicht in der SPS befindet, können Angreifer möglicherweise die SPS und die E/A manipulieren, ohne den HMI-Teil bearbeiten zu müssen, da die richtigen Informationen bereits auf dem Bedienerbildschirm verschleiert sind.
Zuverlässigkeit	1. Die Berechnungen sind effizienter und genauer, wenn sie näher am Feld liegen. Außerdem sind Summen und Zählungen weiterhin verfügbar, wenn die HMI neu gestartet wird (SPS starten nicht so oft neu und speichern diese Werte normalerweise im nichtflüchtigen Speicher). 2. Unterschiedliche Quellen für Eingaben und Verriegelungen können zu nicht erwarteten Fehlern führen. Es kann verschiedene Technologien für HMIs in einer Anlage geben (SCADA-Schicht, aber auch Feldsteuerungspanels), und Änderungen in einer dieser Technologien werden nicht über die restlichen Schichten verbreitet, was zu Inkonsistenzen in der Visualisierung und möglichen Ausfällen im Betrieb führt.
Instandhaltung	Das Programm ist, im Gegensatz zu Übertragung von HMIs zu HMIs, leicht zu verstehen und von SPS zu SPS zu übertragen.

Referenzen

Standard / Framework	Mapping
MITRE ATT&CK for ICS	**Taktik**: TA0106 Impair Prozess Control T0836 Modyfy Parameter
ISA 62443-3-3	**SR 3.6** : Deterministische Ausgabe
ISA 62443-4-2	**CR 3.6** : Deterministische Ausgabe

4. SPS-Flags als Integritätsprüfung verwenden

Setzen Sie Zähler auf SPS-Fehler-Flags, um mathematische Probleme zu erfassen.

Sicherheitsziel	Zilegruppe
Integrität der SPS-Logik	Produktlieferant / Integrator / Wartungsdienstleister

Anleitung

Wenn der SPS-Code einwandfrei funktionierte, aber plötzlich durch Null dividiert, untersuchen Sie dies! Wenn etwas Peer-to-Peer von einer anderen SPS kommuniziert und die Funktion/Logik eine Division durch Null vornimmt, obwohl dies nicht erwartet wurde, untersuchen Sie dies!

Die meisten Programmierer ignorieren das Problem als mathematischen Fehler oder, schlimmer noch, gehen davon aus, dass ihr Code perfekt ist, und lassen die SPS in einen harten Fehlerzustand übergehen.
Während der Codeentwicklung müssen Ingenieure testen und validieren ihre Codemodule (Snippets oder Routinen), indem sie Daten, die außerhalb der erwarteten Grenzen liegen, eingeben. Dies kann als Unit Level Test bezeichnet werden.

Weisen Sie für Firmware, Logik und Protokoll-Stack verschiedene, gesperrte Speichersegmente zu. Testen Sie den Protokoll-Stack auf Missbrauchsfälle. Missbrauchsfälle können eigenartige Flag-Bedingungen in einem Paket-Header sein.

Beispiel

SPS-Fehler, die durch Out-of-Bounds-Daten verursacht werden, sind sehr häufig. Dies geschieht beispielsweise, wenn ein Eingabewert dazu führt, dass Array-Indizes außerhalb des Bereichs liegen, oder Timer mit negativen Voreinstellungen oder durch Null-Ausnahmen dividieren.

Typische Flaggen von Interesse sind:

- Dividieren durch Null
- Gegenüberlauf
- negativer Zähler oder Timer voreingestellt
- I/O-Scan-Überlauf

Begründung

Vorteilhaft für	Begründung
Sicherheit	Angriffe auf SPS können die Änderung ihrer Logik, die Aktivierung eines neuen Programms, das Testen von neuem Code, das Laden eines neuen Prozessrezepts, das Einfügen von Hilfslogik zum Senden von Nachrichten oder das Aktivieren einer Funktion umfassen. Da die meisten SPS keine kryptografischen Integritätsprüfungen bereitstellen, können Flags ein guter Indikator sein, wenn eine der oben genannten Logikänderungen eintritt.
Zuverlässigkeit	Ernst genommene Flags können verhindern, dass die SPS mit Programmierungs- oder I/O-Fehlern läuft. Wenn ein Fehler auftritt, ist die Ursache des Fehlers offensichtlicher.
Instandhaltung	./.

Referenzen

Standard / Framework	Mapping
MITRE ATT&CK für ICS	**Taktik**: TA0106 Impair Prozess Control T0836 Modyfy Parameter
ISA 62443-3-3	**SR 3.5**: Eingabe-Validierung **SR 3.6**: Deterministische Ausgabe
ISA 62443-4-2	**CR 3.5**: Eingabe-Validierung **CR 3.6**: Deterministische Ausgabe
ISA 62443-4-1	**SI-2**: Sichere Codierungsstandards **SVV-1**: Prüfung der Sicherheitsanforderungen
MITRE CWE	**CWE-128:** Wrap-Around-Fehler **CWE-190:** Integer-Überlauf **CWE-369:** Dividieren durch Null **CWE-754:** Unsachgemäße Prüfung auf ungewöhnliche oder außergewöhnliche Bedingungen

5. Verwenden Sie kryptographische und/oder Prüfsummen-Integritätsprüfungen für SPS-Code

Verwenden Sie kryptografische Hashes oder, wenn kryptografische Hashes nicht verfügbar sind Prüfsummen, um die Integrität des SPS-Codes zu überprüfen und einen Alarm auszulösen, wenn sie sich ändern

Sicherheitsziel	Zilegruppe
Integrität der SPS-Logik	Produktlieferant / Integrator / Wartungsdienstleister /Asset Owner

Anleitung

A) Prüfsummen

Wenn (kryptographische) Hashes nicht möglich sind, können Prüfsummen eine Option sein. Einige SPSen generieren eine eindeutige Prüfsumme, wenn Code in die SPS-Hardware heruntergeladen wird. Die Prüfsumme sollte nach SAT vom Hersteller / Integrator dokumentiert werden und Bestandteil der Garantie-/Servicebedingungen sein.

Wenn die Prüfsummenfunktion nicht nativ in der Steuerung verfügbar ist, kann diese auch in der EWS/HMI generiert und z. B. einmal täglich abgetastet werden, um sie mit dem Hash des Originalcodes in der SPS zu vergleichen, um zu überprüfen, ob sie übereinstimmen. Dies liefert zwar keine Echtzeitwarnungen, ist aber gut genug, um zu verfolgen, ob jemand versucht, Änderungen am SPS-Code vorzunehmen.

Der Prüfsummenwert kann auch in ein SPS-Register verschoben und für einen Alarm konfiguriert werden, wenn er sich ändert, der Wert kann an Historiker usw. gesendet werden.

B) Hashes

SPS-CPUs verfügen in der Regel nicht über die Verarbeitungskapazität, um Hashes während des Betriebs zu generieren oder zu überprüfen. Der Versuch, einen Hash zu erstellen, kann dazu führen, dass die SPS abstürzt. Aber die Engineering-Software der SPS könnte in der Lage sein, Hashes aus dem SPS-Code zu berechnen und sie entweder in der SPS oder an einer anderen Stelle in der Steuerung zu speichern.

Beispiel

SPS-Anbieter, von denen bekannt ist, dass sie über Prüfsummenfunktionen verfügen:

- Siemens (siehe Beispiel)
- Rockwell

Auch externe Software kann zur Generierung von Prüfsummen verwendet werden z.B.:

- Octplant (vormals „Version dog“)
- Asset Guardian
- PAS

Beispiel für eine Siemens-Implementierung

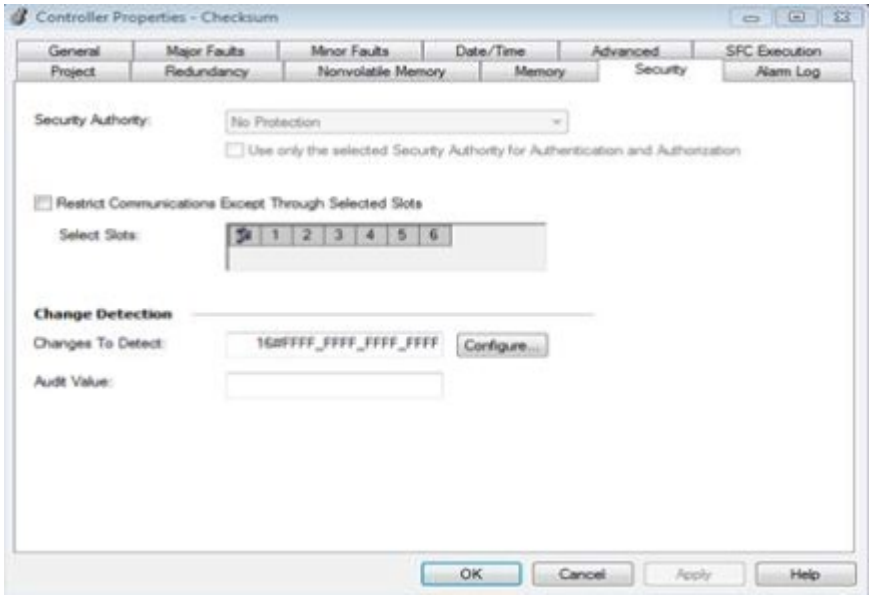
Beispiel für die Erstellung von Prüfsummen in Siemens S7-1500 SPS:
GetChecksum-Function Block liest die aktuelle Prüfsumme und mit einem leichtgewichtigen Skript kann die "SAT-Prüfsumme" als Referenz gespeichert werden. Eine Abweichung von der Referenz-Prüfsumme kann mit der Datalog-Funktion hinterlegt werden.

	Date	UTC Time	Referenz	Aktuell
1	11/21/2019	9:55:11	84 2A 76 DF 5B 31 F4 16	FF 2C EA 71 44 D7 81 04
2	11/21/2019	9:57:33	FF 2C EA 71 44 D7 81 04	FF 2C EA 71 44 D7 81 04
3	11/21/2019	9:58:17	FF 2C EA 71 44 D7 81 04	5B 7C 57 7E E2 3E EF C3
4	11/21/2019	9:58:36	FF 2C EA 71 44 D7 81 04	5B 7C 57 7E E2 3E EF C3
5	11/21/2019	9:58:44	5B 7C 57 7E E2 3E EF C3	5B 7C 57 7E E2 3E EF C3

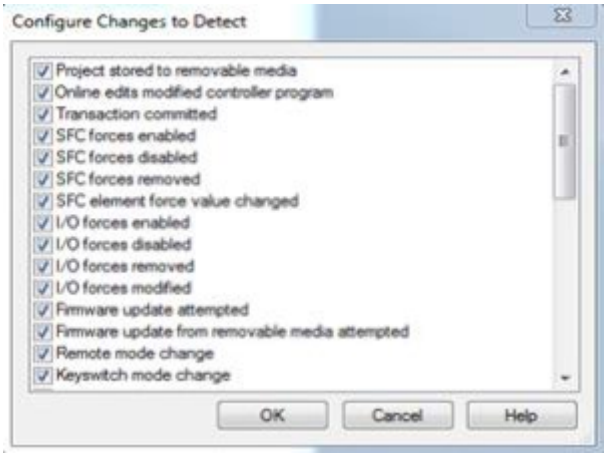
Rockwell-Implementierungsbeispiel:

Dies ist ein Teilbeispiel dafür, wie ein Unternehmen eine Ebene der SPS-Programmänderungserkennung in seiner ICS-Umgebung entwickeln kann. Dieses Beispiel gilt speziell für eine ControlLogix-SPS von Rockwell Automation und ist nicht vollständig. Es wird jedoch veranschaulicht, wie der Zustand des SPS-Prozessors in ein Register innerhalb der SPS abgerufen wird. Sobald sich ein Register in der SPS befindet, kann das Unternehmen damit einen Konfigurationsänderungsalarm für die Anzeige auf einem HMI erstellen, die Rohzustandsinformationen zur Trendanalyse und Überwachung an ein HMI übertragen oder zur Langzeiterfassung an einen Historian senden. Diese Praxis bietet die Möglichkeit, mithilfe vorhandener Tools und Funktionen ein Situationsbewusstsein dafür zu erlangen, wann sich kritische Cyber-Assets ändern. Es liegt an der Organisation, die Verwendung dieses Beispiels in einer Methode durchzuführen, die in ihrer Umgebung am besten funktioniert.

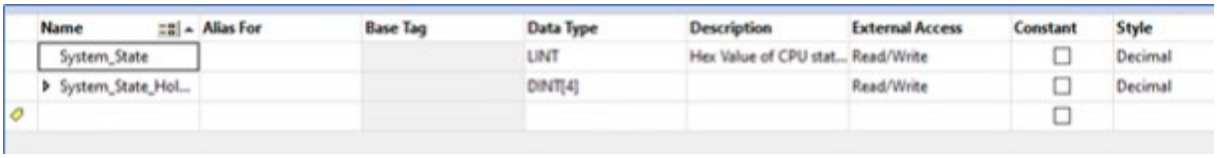
1. Wählen Sie im Dialogfeld "Controller-Eigenschaften" die Schaltfläche "Konfigurieren" auf "Change to Detect" aus:



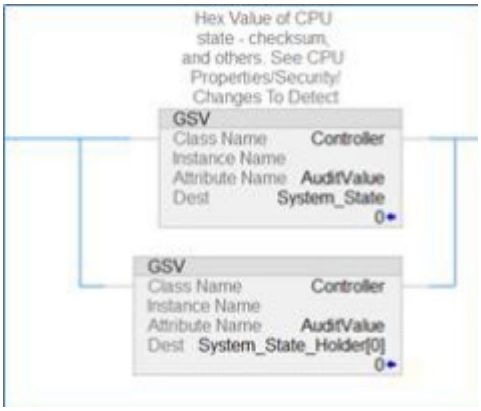
2. Wählen Sie innerhalb des Auswahlfensters alle zu überwachenden Positionen aus:



3. Erstellen Sie ein Tag, um die Informationen zum Prozessorstatus zu erhalten. Dieses Tag kann vom Typ "LINT" oder ein 2-Wort-Array vom Typ "DINT" sein



4. Verwenden Sie die GSV-Anweisung (Get System Values), um die Prozessorstatusinformationen aus dem Speicher abzurufen und in ein Tag zu verschieben, das in der Logik verwendet oder auf dem HMI gelesen werden kann:



Begründung

Vorteilhaft für	Begründung
Sicherheit	Angriffe auf SPS können die Änderung ihrer Logik, die Aktivierung eines neuen Programms, das Testen von neuem Code, das Laden eines neuen Prozessrezepts, das Einfügen von Hilfslogik zum Senden von Nachrichten oder das Aktivieren einer Funktion umfassen. Da die meisten SPS keine kryptografischen Integritätsprüfungen bereitstellen, können Flags ein guter Indikator sein, wenn eine der oben genannten Logikänderungen eintritt.
Zuverlässigkeit	Ernst genommene Flags können verhindern, dass die SPS mit Programmierungs- oder I/O-Fehlern läuft. Wenn ein Fehler auftritt, ist die Ursache des Fehlers offensichtlicher.
Instandhaltung	./.

Referenzen

Standard / Framework	Mapping
MITRE ATT&CK für ICS	**Taktik**: TA0104 Execusion, TA0106 Impair Prozess Control **Technik:** T0873 Protect File Infection,
ISA 62443-3-3	**SR 3.5**: Eingabe-Validierung **SR 3.6**: Deterministische Ausgabe
ISA 62443-4-2	**CR 3.4**: Software- und Informationsintegrität **EDR 3.1:** Bereitstellung von Vertrauensankern von Produktlieferanten
ISA 62443-4-1	**SI-1:** Überprüfung der Sicherheitsimplementierung **SVV-1:** Prüfung der Sicherheitsanforderungen
MITRE CWE	**CWE-345**: Unzureichende Überprüfung der Datenauthentizität • **CWE-353:** Fehlende Unterstützung für Integritätsprüfung • **CWE-354:** Unsachgemäße Validierung des Integritätsprüfungswerts

6. Timer und Zähler validieren

Wenn Zeitschaltuhren und Zählerwerte in das SPS-Programm geschrieben werden, sollten diese von der SPS auf Plausibilität überprüft werden und Rückwärtszählungen unter Null erkennen.

Sicherheitsziel	Zilegruppe
Integrität der SPS-Variabelen	Integrator / Wartungsdienstleister Asset Owner

Anleitung

Timer und Zähler können technisch auf einen beliebigen Wert voreingestellt werden. Daher sollte der gültige Bereich für die Voreinstellung eines Timers oder Zählers eingeschränkt werden, um die betrieblichen Anforderungen zu erfüllen.

Wenn Remote-Geräte, wie z. B. ein HMI, Timer oder Zählerwerte in ein Programm schreiben:

- Lassen Sie das HMI nicht direkt auf den Timer oder Zähler schreiben, sondern durchlaufen Sie eine Validierungslogik
- Validierung von Voreinstellungen und Timeout-Werten in der SPS

Die Validierung von Timer- und Zählereingängen ist einfach direkt in der SPS durchzuführen (ohne dass ein Netzwerkgerät erforderlich ist, das Deep Packet Inspection unterstützt), da die SPS "weiß", was der Prozesszustand oder der Kontext ist. Es kann validieren, "was" es bekommt und "wann" es die Befehle oder Sollwerte erhält.

Beispiel

Beim SPS-Start sind Timer und Zähler in der Regel auf bestimmte Werte voreingestellt.

Wenn es einen Timer gibt, der Alarme bei 1,3 Sekunden auslöst, dieser Timer aber böswillig auf 5 Minuten voreingestellt ist, löst er den Alarm möglicherweise nicht aus.

Wenn es einen Zähler gibt, der bewirkt, dass ein Prozess beendet wird, wenn er 10.000 erreicht, aber von Anfang an auf 11.000 festgelegt ist, wird der Prozess möglicherweise nicht beendet.

Begründung

Vorteilhaft für	Begründung
Sicherheit	Wenn I/O, Timer oder Presets direkt in I/O geschrieben werden und nicht von der SPS validiert werden, wird die SPS-Validierungsschicht umgangen und dem HMI (oder anderen Netzwerkgeräten) wird eine ungerechtfertigte Vertrauensebene zugewiesen.
Zuverlässigkeit	Die SPS kann auch validieren, wenn ein Bediener versehentlich falsche Timer- oder Zählerwerte voreingestellt hat.
Instandhaltung	Gültige Bereiche für Timer und Zähler zu dokumentieren und automatisch zu validieren, kann bei der Aktualisierung der Logik hilfreich sein.

Referenzen

Standard / Framework	Mapping
MITRE ATT&CK für ICS	**Taktik**: TA0104 Execusion, TA0106 Impair Prozess Control **Technik:** T0873 Protect File Infection,
ISA 62443-3-3	**SR 3.5**: Eingabe-Validierung
ISA 62443-4-2	**CR 3.5**: Eingabe Validierung
ISA 62443-4-1	**SI-2:** Secure Coding Standar **SVV-1:** Prüfung der Sicherheitsanforderungen

7. Validieren und Alarmieren für gepaarte Ein-/Ausgänge

Wenn Sie gekoppelte Signale haben, stellen Sie sicher, dass beide Signale nicht zusammen bestätigt werden. Alarmieren Sie den Bediener, wenn Ein-/Ausgangszustände auftreten, die physikalisch nicht realisierbar sind. Erwägen Sie, gekoppelte Signale unabhängig zu machen oder Verzögerungstimer hinzuzufügen, wenn Sie die Ausgänge umschalten könnte die Aktuatoren beschädigen.

Sicherheitsziel	Zilegruppe
Integrität der SPS-Variabelen **Resillienz**	Produktlieferant / Integrator / Wartungsdienstleister

Anleitung

Gepaarte Ein- oder Ausgänge sind solche, die physikalisch nicht gleichzeitig stattfinden können. Sie schließen sich gegenseitig aus. Obwohl gepaarte Signale nicht gleichzeitig bestätigt werden können, es sei denn, es liegt ein Fehler oder eine böswillige Aktivität vor, verhindern SPS-Programmierer diese Assertion oft nicht.

Die Validierung ist am einfachsten direkt in der SPS durchzuführen, da die SPS den Prozesszustand oder den Kontext kennt. Gepaarte Signale sind leichter zu erkennen und zu verfolgen, wenn sie sequentielle Adressen haben (z. B. Eingang 1 und Eingang 2).

Ein weiteres Szenario, in dem gepaarte Ein- oder Ausgänge Probleme verursachen können, ist, wenn sie nicht gleichzeitig bestätigt, sondern schnell umgeschaltet werden, wodurch Aktoren beschädigt werden.

Beispiel

Beispiele für gepaarte Signale:

START und STOPP

- Unabhängiger Start und Stopp: Konfigurieren Sie Start und Stopp als diskrete Ausgänge anstelle von mit einem einzigen Ausgang, der ein- und ausgeschaltet werden kann. Dadurch sind keine gleichzeitigen Auslöser möglich. Für einen Angreifer ist es viel komplizierter, schnell ein- und auszuschalten, wenn zwei verschiedene Ausgänge gesetzt werden müssen.
- Timer für Neustart: Erwägen Sie auch, einen Timer für einen Neustart hinzuzufügen, nachdem ein Stopp an Vermeiden Sie schnelles Ausschalten von Start-/Stoppsignalen.
- VORWÄRTS und RÜCKWÄRTS
- ÖFFNEN und SCHLIESSEN

Beispiele für das Umschalten von Signalpaaren, die schädlich sein könnten:
Wenn die SPS / MCC einen diskreten Eingang akzeptiert, bietet dies einem Angreifer eine einfache Möglichkeit, physischen Schaden an Aktoren zu verursachen. Das bekannte Szenario für das Umschalten von Ausgängen, um Schaden zu verursachen, wäre ein MCC, aber diese Vorgehensweise gilt für alle Szenarien, in denen das Umschalten von Ausgängen Schaden verursachen könnte. Ein Proof of Concept, bei dem schnell umschaltbare Ausgänge echten Schaden anrichten konnten, war der Aurora-Generator-Test im Jahr 2007, der vom Idaho National Laboratory durchgeführt wurde, bei dem das Umschalten von Ausgängen außerhalb der Synchronität zu Schäden am Leistungsschalter führte.

Begründung

Vorteilhaft für	Begründung
Sicherheit	5. Wenn SPS-Programme nicht berücksichtigen, was passiert, wenn beide gepaarten Eingangssignale gleichzeitig ausgegeben werden, ist dies ein Guter Angriffsvektor. 6. Wenn beide gepaarten Eingangssignale bestätigt werden, ist dies eine Warnung, dass Es liegt ein Bedienungsfehler, ein Programmierfehler oder etwas Bösartiges vor. 7. Dadurch wird ein Angriffsszenario vermieden, bei dem physischer Schaden angerichtet werden kann Aktuatoren verursacht werden.
Zuverlässigkeit	8. Gepaarte Eingangssignale können darauf hindeuten, dass ein Sensor defekt oder falsch verdrahtet ist oder dass ein mechanisches Problem vorliegt, z. B. ein festsitzender Schalter. 9. Das schnelle Umschalten von Start und Stopp kann auch versehentlich erfolgen, so dass auch Schäden vermieden werden, die versehentlich verursacht werden könnten.
Instandhaltung	**./.**

Referenzen

Standard / Framework	Mapping
MITRE ATT&CK für ICS	**Taktik:**TA0106 Impair Prozess Control **Technik:** T0836 Modify Parameter, T0806 Brute Force I/O
ISA 62443-3-3	**SR 3.5**: Eingabe-Validierung **SR 3.6** Deterministischer Output
ISA 62443-4-2	**CR 3.5**: Eingabe Validierung **CR 3.6** Deterministischer Output
ISA 62443-4-1	**SI-2:** Secure Coding Standard **SVV-1:** Prüfung der Sicherheitsanforderungen
MITRE CWE	**CWE 754:** Unzureichende Prüfung auf ungewöhnliche oder besondere Bedingungen

8. HMI-Eingangsvariablen auf SPS-Ebene validieren, nicht nur auf HMI

HMI-Zugriff auf SPS-Variablen kann (und sollte) auf einen gültigen Betriebswertebereich am HMI beschränkt werden, darüber hinaus sollten weitere Gegenprüfungen in der SPS hinzugefügt werden, um Werte außerhalb der akzeptablen Bereiche, die in der HMI programmiert sind, zu verhindern oder zu warnen.

Sicherheitsziel	Zilegruppe
Integrität der SPS-Variabelen	Produktlieferant / Integrator / Wartungsdienstleister

Anleitung

Die Eingabevalidierung kann Out-of-Bounds-Überprüfungen auf gültige Betriebswerte sowie auf gültige Werte in Bezug auf Datentypen umfassen, die relativ zum Prozess sind.

Wenn eine SPS-Variable einen Wert empfängt, der außerhalb des zulässigen Bereichs liegt, stellen Sie SPS-Logik bereit, um entweder

- Geben Sie einen **Standardwert** in diese Variable ein, der sich nicht negativ auf den Prozess auswirkt, und kann als Flag für Warnmeldungen verwendet werden oder
- Geben Sie den **letzten richtigen Wert** für diesen Wert ein und protokollieren Sie das Ereignis zur weiteren Analyse.

Beispiel

Beispiel 1

Für eine Operation muss ein Benutzer einen Wert für den Ventildruck in ein HMI eingeben. Gültige Bereiche hierfür 0-100, und die Eingabe des Benutzers wird von der Benutzereingabefunktion auf dem HMI an die Variable V1 in der SPS übergeben. In diesem Fall

1. Der HMI-Eingang für die Variable V1 hat einen eingeschränkten Bereich von 0-100 (Dez.), der im HMI programmiert ist.
   arabische Ziffer. Die SPS verfügt über eine Cross-Check-Logik, die besagt:

```
IF V1 < 0 OR IF V1 > 100, SET V1 = 0.
```

Dadurch wird eine positive Antwort eines vermeintlich sicheren Werts auf eine ungültige Eingabe dieser Variablen bereitgestellt.

Beispiel 2

Eine Operation erfordert eine Benutzereingabe für Messschwellenwerte für eine Variable, die immer innerhalb eines INT2-Datenbereichs liegen sollte. Die Benutzereingabe wird vom HMI an die Variable V2 in der SPS übergeben, die ein 16-Bit-Datenregister ist.

1. Der HMI-Eingang zur Variablen V2 hat einen eingeschränkten Bereich von -32768 bis 32767 (Dez.), der in das HMI.
2. Die SPS verfügt über eine Datentyp-Cross-Check-Logik, die die Überlaufvariable (V3) überwacht, die existiert direkt nach V2 in der Speicherstruktur der SPS:

```
IF V2 = -32768 OR IF V2 = 32767 AND V3 != 0,

SET V2 = 0 AND SET V3 = 0 AND SET DataTypeOverflowAlarm = TRUE.
```

Beispiel 3

Skalieren Sie PV (Prozesswert), SP (Sollwert) und CV (Steuervariable) für PID (Proportional, Integral, Derivative Controller) auf konsistente oder Roheinheiten, um Skalierungsfehler zu vermeiden, die zu Steuerungsproblemen führen. Eine falsche Skalierung kann zu unbeabsichtigten Missbrauchsfällen führen.

Begründung

Vorteilhaft für	Begründung
Sicherheit	1. Während HMIs in der Regel eine Art Eingabevalidierung bieten, kann ein böswilliger Akteur modifizierte Pakete erstellen oder wiedergeben, um beliebige Werte an die Variablen in der SPS zu senden, die für äußere Einflüsse offen sind (offen für Werte, die z. B. von einem HMI übergeben werden). 2. SPS-Protokolle werden in der Regel als "offene" Protokolle vermarktet und für die breite Öffentlichkeit veröffentlicht, so dass die Entwicklung von Malware, die "offene" Protokollinformationen verwendet, trivial sein kann. Die SPS-Variablenzuordnung kann in der Regel durch die Analyse des Datenverkehrs während der Aufklärungsphasen eines Angriffs erfolgen und dem Eindringling so die notwendigen Informationen liefern, um bösartigen Datenverkehr zum Ziel zu erstellen und dadurch einen Prozess mit nicht autorisierten Werkzeugen zu manipulieren. Durch die Überprüfung von Werten, die an die SPS übergeben werden, bevor diese Daten in den Prozess implementiert werden, werden gültige Datenbereiche sichergestellt und ein ungültiger Wert in diesen Speicherbereichen abgeschwächt, indem sichere Bereiche erzwungen werden, wenn ein Wert während des SPS-Scans als außerhalb des Bereichs erkannt wird.
Zuverlässigkeit	./.
Instandhaltung	**./.**

Referenzen

Standard / Framework	Mapping
MITRE ATT&CK für ICS	**Taktik:**TA0106 Impair Prozess Control **Technik:** T0836 Modify Parameter,
ISA 62443-3-3	**SR 3.5**: Eingabe-Validierung **SR 3.6** Deterministischer Output
ISA 62443-4-2	**CR 3.5**: Eingabe Validierung **CR 3.6** Deterministischer Output
ISA 62443-4-1	**SI-2:** Secure Coding Standard **SVV-1:** Prüfung der Sicherheitsanforderungen
MITRE CWE	**CWE 1320:** Unsachgemäßer Schutz für Signalpegelwarnungen außerhalb des zulässigen Bereichs

9. Validieren von Umleitungen

Überprüfen Sie Derefenzierungen, indem Sie Array-Enden vergiften, um „Fencepost Error“/ „Zaunpfahlfehler“ abzufangen.

Sicherheitsziel	Zilegruppe
Integrität der SPS-Variabelen	Produktlieferant / Integrator / Wartungsdienstleister

Anleitung

Eine Dereferenzierung ist die Verwendung des Werts eines Registers in einem anderen Register. Es gibt viele Gründe für die Verwendung von Dereferenzierungen.

Beispiele für notwendige Indirektionen sind:

- Frequenzumrichter (VFDs), die unterschiedliche Aktionen für unterschiedliche Frequenzen auslösen

Nachschlagetabellen.

- Um zu entscheiden, welche Pumpe zuerst in Betrieb genommen werden soll, basierend auf den aktuellen Laufzeiten

SPS haben in der Regel kein "Ende eines Array"-Flags, daher ist es eine gute Idee, es in Software zu erstellen. Ziel ist es, ungewöhnliche/ungeplante SPS-Vorgänge zu vermeiden.

Beispiel

Programmierung der Anweisungsliste (IL)

Der Ansatz kann in wenige Funktionsblöcke umgewandelt und möglicherweise sogar für andere Anwendungen wiederverwendet werden.

1. Array-Maske erstellen

Überprüfen Sie, ob das Array binär groß ist. Wenn es sich nicht um eine binäre Größe handelt, erstellen Sie eine Maske mit der nächsten Größe auf einer binären Skala. z.B. wenn Sie 5 Register benötigen (nicht binär):

```
[21 31 41 51 61]
```

Definieren Sie ein Array von 8:

```
[x x 21 31 41 51 61 x]
```

Nehmen Sie als Nächstes den Indexwert, der für die Dereferenzierung ermittelt werden soll - in diesem Beispiel ist es 3.

Achtung: Der Index beginnt bei 0!

```
[21 31 41 51 61]
          ^
```

Index: 3

Füge einen Versatz hinzu, der das vergiftete Ende ausgleicht. Der Offset kann 1 oder höher sein, in diesem Fall ist es 2:

```
[x x 21 31 41 51 61 x]
              ^
```

Index inkl. Offset: 3 + 2 = 5

und dann AND der Index einschließlich Offset mit einer Maske, die der Array-Größe entspricht.
In diesem Beispiel ist die Array-Größe 8, also Index 7, so dass die Maske 0x07 wäre. Die Maske stellt sicher, dass der maximale Index, den Sie erhalten können, 7 ist, zum Beispiel:

6 UND 0x07 würde 6 zurückgeben.

7 UND 0x07 würde 7 zurückgeben.

8 UND 0x07 würde 0 zurückgeben.

9 UND 0x07 würde 1 zurückgeben.

Dadurch wird sichergestellt, dass Sie immer einen Wert im Array adressieren.

2. Vergiftete Enden einsetzen

Das Ende der Vergiftung ist optional. Sie wären in der Lage, manipulierte Indirektionen ohne die Vergiftung zu erkennen, aber die Vergiftung hilft, Zaunpfahlfehler abzufangen, da Sie einen Wert zurückerhalten, der keinen Sinn ergibt.

Der Punkt ist, dass bei Index 0 des Arrays ein ungültiger Wert vorhanden sein sollte – z. B. -1 oder 65535. Das ist "das vergiftete Ende". Ebenso machen Sie bei den letzten Elementen des Arrays dasselbe:

Für das obige Array könnte die vergiftete Version also wie folgt aussehen:

```
[-1 -1 21 31 41 51 61 -1]
```

3. Erfassen Sie den Wert der Dereferenzierungsadresse ohne Maske

Notieren Sie dann den Wert der indirekten Adresse ohne UND-Maske und Offset:

In diesem Beispiel würden Sie 51 für Index 3 aufzeichnen.

```
[21 31 41 51 61]
          ^
_________
______Index 3
```

4. UND-Maske ausführen und Werte vergleichen (=Dereferenzierungsvalidierung)

Vergleichen Sie Ihren aufgezeichneten Wert mit dem Wert, nachdem Sie den Versatz und die UND-Maske vorgenommen haben.

4a. Fall A: Korrekte Dereferenzierung

Erstens, Versatz:
Index plus Offset = 3 plus 2 = 5

Zweitens, Maske:
5 UND 0x07 = 5

Drittens, Dereferenzierungsprüfung:

```
[-1 -1 21 31 41 51 61 -1]
                  ^
 _________________
```

Index inkl. Offset: 5
Wert = 51 entspricht dem aufgezeichneten Wert, also ist alles in Ordnung.

4b. Fall B: Manipulierte Dereferenzierung

Wenn Sie jetzt eine manipulierte Dereferenzierung hatten, sagen wir 7, lassen Sie uns sehen,

Erstens, Versatz:
Index plus Offset = 7 plus 2 = 9

Zweitens, Maske:
9 UND 0x07 = 1

Drittens, Dereferenzierungsprüfung:
[-1 **-1** 21 31 41 51 61 -1
____^

Index inkl. Offset: **1**
Wert = **-1** entspricht nicht dem aufgezeichneten Wert und zeigt auch Ihr vergiftetes Ende an, sodass Sie wissen, dass Ihre Dereferenzierung manipuliert ist.

5. Störung / Programmierer-Alarm ausführen

Wenn sich dieser validierte Wert von Ihrem aufgezeichneten Wert unterscheidet, wissen Sie, dass etwas nicht stimmt. Lösen Sie einen Software-Qualitätsalarm aus.

Überprüfen Sie dann den Wert für die Dereferenzierung. Wenn es sich um einen vergifteten Wert handelt, sollten Sie einen weiteren Softwarequalitätsalarm auslösen. Dies ist ein Hinweis auf einen Zaunpfostenfehler.

Begründung

Vorteilhaft für	Begründung
Sicherheit	Die meisten SPS verfügen über keine Funktion zur Verarbeitung von Out-of-Bounds-Indizes für Arrays. Es gibt zwei potenziell gefährliche Szenarien, die sich aus Dereferenzierungsfehlern ergeben können: **Erstens**, wenn eine Dereferenzierung dazu führt, dass aus dem falschen Register gelesen wird, wird das Programm mit falschen Werten ausgeführt. **Zweitens**, wenn eine falsche Dereferenzierung dazu führt, dass in das falsche Register geschrieben wird, überschreibt das Programm Code oder Werte, die Sie beibehalten möchten. In beiden Fällen können Dereferenzierungsfehler schwer zu erkennen sein und schwerwiegende Auswirkungen haben. Sie können durch menschliches Versagen verursacht werden, aber auch böswillig eingefügt werden.
Zuverlässigkeit	Identifiziert nicht böswillige menschliche Fehler bei der Programmierung
Instandhaltung	/

Referenzen

Standard / Framework	Mapping
MITRE ATT&CK für ICS	**Taktik:**TA0106 Impair Prozess Control **Technik:** T0836 Modify Parameter,
ISA 62443-3-3	**SR 3.5**: Eingabe-Validierung **SR 3.6** Deterministischer Output
ISA 62443-4-2	**CR 3.5**: Eingabe Validierung **CR 3.6** Deterministischer Output
ISA 62443-4-1	**SI-2:** Secure Coding Standard **SVV-1:** Prüfung der Sicherheitsanforderungen
MITRE CWE	**CWE 129:** Fehlerhafte Validierung des Array-Index

10. Zugewiesene Registerblöcke nach Funktion (Lesen/Schreiben/Validieren)

Weisen Sie bestimmte Registerblöcke für bestimmte Funktionen zu, um Daten zu validieren, Pufferüberläufe und blockieren Sie nicht autorisierte externe Schreibvorgänge, um Steuerungsdaten zu schützen.

Sicherheitsziel	Zilegruppe
Integrität der SPS-Variablen	Produktlieferant / Integrator / Wartungsdienstleister

Anleitung

Temporärer Speicher, auch bekannt als Notizblockspeicher, ist ein leicht ausnutzbarer Speicherbereich, wenn diese Vorgehensweise nicht befolgt wird. z.B. könnte das einfache Schreiben in ein "Modbus"-Register, das außerhalb der Grenzen liegt, dazu führen, dass Speicherregister überschrieben werden, die für temporäre Berechnungen verwendet werden.

Im Allgemeinen können andere Geräte im SPS-Netzwerk für Lese- und Schreibvorgänge auf den Registerspeicher zugreifen. Einige Register können von einem HMI gelesen werden, andere von einem SCADA-System usw. Spezifische Register-Arrays für eine bestimmte Anwendung zu haben, erleichtert auch die Konfiguration des schreibgeschützten Zugriffs von einem anderen Gerät/HMI (in der Steuerung oder bei Verwendung einer externen Firewall).

Beispiele für Funktionen, für die ausgewiesene Registerbausteine sinnvoll sind, sind:

- Lesen
- Schreiben (von HMI / Controller / anderem externen Gerät)
- Validieren von Schreibvorgängen
- Berechnungen

Das Sicherstellen externer Schreibvorgänge in zulässige Register trägt auch dazu bei, Fehler beim Zurücksetzen des Hauptspeichers zu vermeiden, die entweder auf eine Ausführung außerhalb des Bereichs oder böswillige Versuche zurückzuführen sind. Diese ausgewiesenen Registerblöcke können verwendet werden als Puffer für E/A-, Timer- und Zählerschreibvorgänge, indem überprüft wird, ob der Puffer vollständig geschrieben ist (er enthält nicht teils alte, teils neue Daten) und alle Daten im Puffer validiert werden.

Hintergrund:

Hauptspeicher und Registerspeicher werden unterschiedlich genutzt. Der Hauptspeicher wird zum Speichern der aktuell ausgeführten Programmlogik verwendet, während der Registerspeicher von der aktuell ausgeführten Logik als temporärer Speicher verwendet wird. Obwohl der Registerspeicher temporär ist, muss er, da er von der ausführenden Logik verwendet wird, einige wichtige Variablen enthalten, die sich auf die Hauptlogik auswirken würden.

Beispiel

Beispiele dafür, was passieren könnte, wenn diese Praxis nicht umgesetzt wird:
(*Referenz: G. P. H. Sandaruwan, P. S. Ranaweera, Vladimir A. Oleshchuk, PLC Security and Critical Infrastructure Protection):*

- Siemens verwendet den Scratchpad-Speicher typischerweise im Flag-Bereich von Flag 200.0 bis Flag 255.7. Wenn ein Bit in diesem Bereich geändert wird, besteht die Wahrscheinlichkeit einer schwerwiegenden Fehlfunktion der SPS, basierend auf der Wichtigkeit dieses Bits oder Bytes.
- Angenommen, ein Angreifer kann sich Zugang zu einem der Rechner im SPS-Netzwerk verschaffen und diesen Rechner mit einem Wurm infizieren, der in der Lage ist, beliebige Werte in den Registerspeicher zu schreiben. Da sich die Werte des Registerspeichers willkürlich geändert haben, kann sich der Druckwert ändern.
- Durch das Ausführen der Logik wird basierend auf der Änderung ein neuer Wert festgelegt, der dazu führen kann, dass das System seine Sicherheitsmargen überschreiten und möglicherweise zum Absturz gebracht wird.

Beispiele für die Implementierung dieser Vorgehensweise:

- In einem Szenario, in dem es eine Sicherheitszone gibt (aber das DCS lesen kann), kann die Firewall alle "write"-Versuche mit der Regel, dass diese Register in der Sicherheitszone READ ONLY sind.
- In einem anderen Szenario könnte es einige schreibfähige Register geben, während andere schreibgeschützt sind.
- Da sich jedoch alle READ ONLY-Register in einem einzigen Array befinden, ist es einfacher, sie im Controller (oder in einer Firewall) zu konfigurieren.

Begründung

Vorteilhaft für	Begründung
Sicherheit	Erleichtert den Schutz der Steuerungsdaten durch Funktion (Lesen/Schreiben/Validieren). Erleichtert protokollsensitiven Firewalls ihre Arbeit: Die Regeln werden einfacher, da sehr klar ist, auf welche Registerblöcke das HMI zugreifen darf. Erleichtert die Verwaltung der (einfacheren) Regeln in der Firewall. Das Vornehmen von nicht autorisierten Änderungen am internen temporären Speicher ist eine leicht ausnutzbare Schwachstelle (Bypass Logic Attack). Wenn die Ein- und Ausgänge von SPS-Routinen ordnungsgemäß validiert sind, können alle Änderungen (durch einen böswilligen Akteur oder aus Versehen) leicht abgefangen werden, anstatt lange in der Logiksequenz zu verbleiben und später in der Ausführung Fehler zu verursachen / Probleme zu verursachen.
Zuverlässigkeit	Lese- und Schreibvorgänge werden beschleunigt, da die Anzahl der Transaktionen reduziert wird. Auch autorisierte Änderungen und Programmierfehler können zu Fehlfunktionen führen, wenn der temporäre Speicher nicht geschützt ist. Netzwerk- und Kommunikationsfehler bei langen Nachrichten können zu unbeabsichtigten Fehlern führen, wenn die Gültigkeit der Daten vor der Verarbeitung nicht überprüft wird.
Instandhaltung	Programmierfehler, die dazu führen, dass in den temporären Speicher geschrieben wird, können das Auffinden von Fehlern erschweren, sodass das Problem durch die Zuweisung bestimmter Register für Schreibvorgänge vermieden werden kann.

Referenzen

Standard / Framework	Mapping
MITRE ATT&CK für ICS	**Taktik** : TA0107 - Inhibit Response Function, TA0106 - Impair Process Control **Technik**: T0835 - Manipulate I/O Image , T0836 - Modify Parameter
ISA 62443-3-3	**SR 3.4**: Software- und Informationsintegrität **SR 3.5**: Eingabe-Validierung **SR 3.6**: Deterministische Ausgabe
ISA 62443-4-2	**CR 3.4**: Software- und Informationsintegrität **CR 3.5**: Validierung der Eingaben **CR 3.6**: Deterministische Ausgabe
ISA 62443-4-1	**SD-4**: Best Practices für sicheres Design **SI-1**: Überprüfung der Sicherheitsimplementierung **SI-2**: Sichere Codierungsstandards **SVV-1**: Prüfung der Sicherheitsanforderungen
MITRE CWE	**CWE-787**: Schreiben außerhalb der Grenzen **CWE-653**: Unzureichende Kompartimentierung

11. Instrument zur Plausibilitätsprüfung

Instrumentieren Sie den Prozess so, dass Plausibilitätsprüfungen möglich sind, indem verschiedene Messungen abgeglichen werden.

Sicherheitsziel	Zilegruppe
Integrität der I/O Werte	Produktlieferant / Integrator / Wartungsdienstleister

Anleitung

Es gibt verschiedene Möglichkeiten, die physikalische Plausibilität für die Validierung von Messungen zu nutzen:

a) Vergleichen Sie integrierte und zeitunabhängige Messungen

Plausibilitätsprüfungen können durchgeführt werden, indem zeitabhängige Werte über einen Zeitraum integriert oder differenziert und mit zeitunabhängigen Messungen verglichen werden.

b) Vergleichen Sie verschiedene Messquellen

Auch die Messung desselben Phänomens auf unterschiedliche Weise kann eine gute Plausibilitätsprüfung sein.

Unterschiedliche Messquellen müssen nicht unbedingt unterschiedliche physikalische Sensoren sein, sondern können auch die Nutzung alternativer Kommunikationskanäle bedeuten (siehe Beispiele).

Beispiel

a) Vergleichen Sie integrierte und zeitunabhängige Messungen

- Dosierbare Pumpen- und Tankfüllstandsanzeige: Die Volumenänderung sollte dem integrierten Durchfluss entsprechen.
- Brenner in einem Kessel: Die zugeführte kalorische Wärme sollte dem Temperaturanstieg entsprechen.

b) Vergleichen Sie verschiedene Messquellen

- Verwendung der Fluggeschwindigkeit, des künstlichen Horizonts, der vertikalen Geschwindigkeit und der Höhe im Flugzeug zur Messung des Phänomens des steigenden / sinkenden Flugzeugs.
- Vergleich von Prozessparameterwerten unabhängiger Datenlogger (gebunden in 4-20mA-Schleifen oder Relaiskontakte und über unabhängige Kommunikationskanäle übertragen) zu SCADA-Systemdaten (die auf "normalem" Weg über SPS und HMI kommen) und Alarmierung bei Abweichungen und deutlich abweichenden Werten.

Begründung

Vorteilhaft für	Begründung
Sicherheit	Erleichtert die Überwachung auf manipulierte Werte (vorausgesetzt, nicht alle Sensoren werden gleichzeitig manipuliert).
Zuverlässigkeit	Verhindert die Annahme oder identifiziert (für zukünftige Maßnahmen) beschädigte / falsche Messungen als Eingaben.
Instandhaltung	Schließt mögliche physikalische Ursachen für Ausfälle schneller aus.

Referenzen

Standard / Framework	Mapping
MITRE ATT&CK für ICS	**Taktik** : TA0106 - Impair Process Control
ISA 62443-3-3	**SR 3.5**: Eingabe-Validierung **SR 3.6**: Deterministische Ausgabe
ISA 62443-4-2	**CR 3.5**: Validierung der Eingaben **CR 3.6**: Deterministische Ausgabe
MITRE CWE	**CWE-754**: Unsachgemäße Prüfung auf ungewöhnliche oder außergewöhnliche Bedingungen

12. Validieren Sie Eingaben auf der Grundlage physikalischer Plausibilität

Stellen Sie sicher, dass Bediener nur das eingeben können, was im Prozess praktisch oder physikalisch machbar ist. Stellen Sie einen Timer für einen Vorgang auf die Dauer ein, die er physisch dauern sollte. Erwägen Sie eine Warnung, wenn es Abweichungen gibt. Auch bei unerwarteter Inaktivität warnen.

Sicherheitsziel	Zilegruppe
Integrität der I/O Werte	Integrator / Wartungsdienstleister

Anleitung

a) Überwachen Sie die erwartete physische Dauer

Wenn die Operation länger als erwartet dauert, um von einem Extrem ins andere zu gelangen, ist das ein Grund zur Beunruhigung. Wenn es aber zu schnell geht, ist das auch ein Grund zur Beunruhigung.

Eine einfache Lösung könnte eine Step-Timeout-Warnung sein. Dies wäre nützlich für sequenz-/schrittgesteuerte Aufgaben.
Zum Beispiel dauert der Schritt "Objekt von A nach B bewegen" 5 Sekunden vom Start des Schritts bis die Übergangsbedingung (Sensor: Objekt bei B angekommen) erfüllt ist.
Wenn die Bedingung deutlich zu früh oder zu spät erfüllt wird, wird der Step-Timeout ausgelöst.

b) Überwachen Sie die erwartete sich wiederholende körperliche Aktivität

Die physikalische Plausibilitätsprüfung kann auch bedeuten, dass eine physikalisch unplausible Inaktivität gemeldet wird: Wenn ein regelmäßiger, sich wiederholender Zyklus von Ereignissen (z. B. Chargen, Tagesabläufe) erwartet wird, würde ein Inaktivitäts-Timer alarmieren, wenn etwas, von dem erwartet wird, dass es sich ändert (diskreter oder analoger Wert), viel zu lange statisch bleibt.

Beispiel

a) Überwachen Sie die erwartete physische Dauer

- Die Tore eines Damms brauchen eine gewisse Zeit, um von vollständig geschlossen zu vollständig geöffnet zu werden
- In einem Abwasserversorger dauert es eine bestimmte Zeit, bis sich ein Pumpensumpf füllt.

b) Überwachen Sie die erwartete sich wiederholende körperliche Aktivität

- Der Herstellungsprozess oder die Chargenverarbeitung in der Rohrleitung sollte regelmäßig zwischen den Betriebsarten.
- Kommunale Kläranlagen haben in der Regel einen täglichen Aktivitätszyklus
- Durchflussraten im Zulauf.

c) Beschränken Sie die Bedienereingabe für Sollwerte auf das praktische/physikalisch Mögliche.

- z. B. der Fall Oldsmar, Florida, ermöglichte Bedienereingaben, die a) tausendmal mehr sind

als das, was normalerweise benötigt wurde b) das physikalisch nicht möglich ist. Versuchen Sie, wo immer möglich, die Betriebsgrenzen im SPS-Code zu konfigurieren, anstatt HMI-Skripte zu verwenden.

Begründung

Vorteilhaft für	Begründung
Sicherheit	1. Abweichungen können darauf hindeuten, dass sich ein Aktuator bereits mitten in einem Reisezustand befand oder dass jemand versucht, die I/O zu fälschen, z. B. durch einen Replay-Angriff. 2. Inaktivitätswarnungen erleichtern die Überwachung von eingefrorenen oder erzwungenen konstanten Werten, die das Ergebnis von System- oder Gerätemanipulationen sein könnten.
Zuverlässigkeit	1. Abweichungen geben Ihnen eine frühzeitige Warnung bei defekten Geräten aufgrund elektrischer oder mechanischer Ausfälle. 2. Inaktivitätswarnungen helfen dabei, Messungen oder Systemregelkreise zu kennzeichnen, die aufgrund eines physischen Gerätefehlers oder eines Problems mit dem logischen Steuerungsalgorithmus oder fehlgeschlagener/unsachgemäßer Bedienereingabe fehlschlagen (d. h. statisch sind).
Instandhaltung	./.

Referenzen

Standard / Framework	Mapping
MITRE ATT&CK für ICS	**Taktik** : TA0106 - Impair Process Control **Technik:** T0806 Brute Force I/O
ISA 62443-3-3	**SR 3.5**: Eingabe-Validierung **SR 3.6**: Deterministische Ausgabe
ISA 62443-4-2	**CR 3.5**: Validierung der Eingaben **CR 3.6**: Deterministische Ausgabe
MITRE CWE	**CWE-754**: Unsachgemäße Prüfung auf ungewöhnliche oder außergewöhnliche Bedingungen

13. Deaktivieren Sie nicht benötigte/ungenutzte Kommunikationsports und -protokolle

SPS-Steuerungen und Netzwerkschnittstellenmodule unterstützen in der Regel mehrere Kommunikationsprotokolle, die standardmäßig aktiviert sind. Deaktivieren Sie Ports und Protokolle, die für die Anwendung nicht erforderlich sind.

Sicherheitsziel	Zilegruppe
Härten	Integrator / Wartungsdienstleister

Anleitung

Gängige Protokolle, die in der Regel standardmäßig aktiviert sind, sind z. B. HTTP, HTTPS, SNMP, Telnet, FTP, MODBUS, PROFIBUS, EtherNet/IP, ICMP usw.

Es empfiehlt sich, ein Datenflussdiagramm zu entwickeln, das die erforderliche Kommunikation zwischen der SPS und anderen Komponenten im System darstellt.

Das Datenflussdiagramm sollte sowohl die physischen Ports der SPS als auch die logischen Netzwerke zeigen, mit denen sie verbunden sind. Für jeden physischen Port sollte eine Liste der erforderlichen Netzwerkprotokolle identifiziert und alle anderen deaktiviert werden.

Beispiel

Beispielsweise enthalten viele SPSen einen eingebetteten Webserver für Wartung und Fehlerbehebung. Wenn diese Funktion nicht verwendet wird, sollte sie nach Möglichkeit deaktiviert werden, da dies ein Angriffsvektor sein könnte.

Begründung

Vorteilhaft für	Begründung
Sicherheit	Jeder aktivierte Port und jedes aktivierte Protokoll vergrößert die potenzielle Angriffsfläche der SPS. Der einfachste Weg, um sicherzustellen, dass ein Angreifer sie nicht für unbefugte Kommunikation verwenden kann, besteht darin, sie vollständig zu deaktivieren.
Zuverlässigkeit	Wenn eine SPS nicht über einen bestimmten Port oder ein bestimmtes Protokoll kommunizieren kann, reduziert dies auch die potenzielle Menge an (fehlerhaftem) Datenverkehr, sei er bösartig oder nicht, was die Wahrscheinlichkeit verringert, dass die SPS aufgrund unbeabsichtigter / fehlerhafter Kommunikationspakete abstürzt.
Instandhaltung	Das Deaktivieren nicht verwendeter Ports und Protokolle erleichtert auch die Wartung, da es die Gesamtkomplexität der SPS reduziert. Was nicht da ist, muss nicht administriert oder aktualisiert werden.

Referenzen

Standard / Framework	Mapping
MITRE ATT&CK für ICS	**Taktik** : TA0102 Discovery **Technik:** T0804 Block Reporting Messages
ISA 62443-3-3	**SR 7.6**: Netzwerk- und Sicherheitskonfigurationseinstellungen **SR 7.7:** Geringste Funktionalität
ISA 62443-4-2	**EDR 2.13:** Verwendung physikalischer Diagnose- und Testschnittstellen
ISA 62443-4-1	**SD-4:** Best Practices für sicheres Design **SI-1:** Überprüfung der Sicherheitsimplementierung **SVV-1:** Testen von Sicherheitsanforderungen

14. Datenschnittstellen von Drittanbietern einschränken

Schränken Sie die Art der Verbindungen und die verfügbaren Daten für 3rd-Party-Schnittstellen ein. Die Verbindungen und/oder Datenschnittstellen sollten klar definiert und so eingeschränkt sein, dass nur Lese-/Schreibfähigkeiten für die erforderliche Datenübertragung möglich sind.

Sicherheitsziel	Zilegruppe
Härten	Integrator / Wartungsdienstleister

Anleitung

In einigen Fällen stellen aufgrund langer Kabelwege oder eines großen Datenaustauschs Schnittstellendatenverbindungen einen besseren Geschäftsfall dar als ein fest verdrahteter Datenaustausch zwischen zwei separaten Parteien.

Die folgenden Richtlinien sollten beim Entwerfen und Implementieren einer Datenaustauschschnittstelle eines Drittanbieters berücksichtigt und befolgt werden, sofern dies praktikabel ist:

- Verwenden Sie ein dediziertes Kommunikationsmodul, das entweder direkt mit der SPS oder den Datenaustauschgeräten des Drittanbieters verbunden ist, oder verwenden Sie dedizierte Netzwerkgeräte, die physisch vom Kernnetzwerk jeder Partei getrennt sind.
- Die MAC-Adresse der angeschlossenen Geräte ist in der Regel in Systemvariablen für jedes ICS verfügbar Ethernet-fähiges Gerät, das es ermöglicht, die Geräteidentität mit einem Multi-Faktor-Ansatz zu überprüfen (IP-Adresse + MAC-Maker-Code = vertrauenswürdiges Gerät). Diese Praxis ist sicherlich nicht narrensicher, da MAC- und IP-Adressen gefälscht werden können, aber sie dient dazu, die Messlatte in Bezug auf die Kommunikation zwischen vertrauenswürdigen ICS-Systemen und -Geräten höher zu legen.
- Wählen Sie bei der Auswahl eines Protokolls für 3rd-Party-Schnittstellen ein Protokoll, das die Fähigkeit des Dritten, Daten in das System des Eigentümers zu schreiben.
- Wählen Sie eine Verbindungsmethode und einen Verbindungsport, der verhindert, dass der Drittanbieter in der Lage, die SPS oder die Datenaustauschgeräte des Eigentümers zu konfigurieren.
- Der Dritte sollte nicht in der Lage sein, Daten zu lesen oder zu schreiben, die nicht ausdrücklich definiert und zur Verfügung gestellt werden.
- Verwenden Sie einen Watchdog-Timer zur Überwachung der Kommunikation, damit keine Befehle an eine SPS im Fehlermodus.
- Serielle Verbindung: Verwenden Sie ein dediziertes Kommunikationsmodul für jede 3rd-Party-Schnittstelle mit Ein eingeschränktes Array von Daten. Stellen Sie sicher, dass die Seite des Besitzers der Verbindung der Initiator und der Drittanbieter der Responder ist.
- Ethernet/IP: Einige SPSen ermöglichen die Funktion von Kommunikationsmodulen als Firewall und können Deep Packet Inspection (DPI) durchführen oder die Schnittstellen der Kommunikationsmodule auf eine Begrenzung beschränken den Datenaustausch zu einer vordefinierten Teilmenge. Wenn diese Funktionen verfügbar sind und ein Ethernet/IP-Protokoll verwendet wird, stellen Sie sicher, dass die Funktionen aktiviert und konfiguriert sind.
- Wenn betriebliche oder vertragliche Anforderungen den Eigentümer daran

hindern, die vorherigen Punkte zu erfüllen, sollten Sie die Verwendung einer separaten "Datenkonzentrator"-SPS (auch bekannt als Proxy/DMZ) in Betracht ziehen, um die Daten zu puffern und den Eigentümer vor unerwünschten Schreibvorgängen/Programmierungen durch die Drittpartei zu schützen. Stellen Sie sicher, dass die Backplane dieser SPS nicht vom 3rd-Party-Netzwerk aus durchquert werden kann.

Beispiel

- Pipeline- oder Leasing-Einheiten für den automatischen eichpflichtigen Verkehr (LACT), die Kohlenwasserstoffe oder Wasser, die zwischen einem Upstream-Produktions- oder Pipeline-Unternehmen und einem Midstream-Pipeline-Unternehmen mit Netzwerk- oder seriellen Schnittstellenverbindungen ausgetauscht werden, die Mess-, Zustands- und Freizügigkeitsinformationen zwischen Unternehmen austauschen.
- Regionaler Trinkwasserversorger (Importeur) teilt die gelieferte Wasserdurchflussmenge an der Weiche an das Wasserwerk einer örtlichen Gemeinde

Begründung

Vorteilhaft für	Begründung
Sicherheit	1. Begrenzen Sie die Gefährdung durch Netzwerke und Geräte von Drittanbietern. 2. Authentifizieren Sie externe Geräte, um Spoofing zu verhindern., besteht darin, sie vollständig zu deaktivieren.
Zuverlässigkeit	Schränkt die Möglichkeit für absichtliche oder unbeabsichtigte Änderungen oder den Zugriff von Standorten oder Geräten von Drittanbietern ein.
Instandhaltung	./.

Referenzen

Standard / Framework	Mapping
MITRE ATT&CK für ICS	**Taktik** : TA0105 Impair Process Control **Technik:** T0836 Modify Parameter
ISA 62443-3-3	**SR 7.6**: Netzwerk- und Sicherheitskonfigurationseinstellungen **SR 7.7:** Geringste Funktionalität
ISA 62443-4-2	**SR 7.6:** Netzwerk- und Sicherheitskonfigurationseinstellungen **CR 7.7:** Geringste Funktionalität
ISA 62443-4-1	**SD-4:** Best Practices für sicheres Design **SI-1:** Überprüfung der Sicherheitsimplementierung **SVV-1:** Testen von Sicherheitsanforderungen

15. Definieren Sie einen sicheren Prozesszustand im Falle eines SPS-Neustarts

Definieren Sie sichere Zustände für den Prozess bei SPS-Neustarts (z. B. Kontakte einschalten, stromlos schalten, vorherigen Zustand beibehalten).

Sicherheitsziel	Zilegruppe
Resillienz	Produktlieferant Integrator / Wartungsdienstleister

Anleitung

Wenn irgendetwas einer SPS befiehlt, mitten in einem Arbeitsprozess neu zu starten, sollten wir erwarten, dass das Programm reibungslos und mit minimaler Unterbrechung des Prozesses aufgenommen wird. Stellen Sie sicher, dass der von ihm gesteuerte Prozess neustartsicher ist.

Wenn es nicht praktikabel ist, die SPS so zu konfigurieren, dass sie einen sicheren Neustart ermöglicht, stellen Sie sicher, dass sie Sie auf diese Tatsache hinweist und keine neuen Befehle ausgibt. Stellen Sie in diesem Fall außerdem sicher, dass die Standardarbeitsanweisungen (SOP) sehr klare Anweisungen zum Einstellen der manuellen Steuerungen enthalten, damit die SPS den Prozess ordnungsgemäß startet.

Dokumentieren Sie außerdem alle Vorgänge zum Starten, Herunterfahren, zur stationären Steuerung und zum Neustart des Flugsteuerungssystems.

Beispiel

/

Begründung

Vorteilhaft für	Begründung
Sicherheit	**Eliminiert potenziell unerwartetes Verhalten:** Der grundlegendste Angriffsvektor für eine SPS besteht darin, sie zum Absturz und / oder Neustart zu zwingen. Für viele SPSen ist das gar nicht so schwer, denn viele SPSen kommen mit unerwarteten Eingaben oder zu viel Verkehr nicht gut zurecht. Es gibt zwar mehrere Diagnosen für Controller-Aktionen während der Ausführung, aber es ist in der Regel nicht klar, wie der Start mit einem laufenden Prozess gehandhabt wird. Dies kann dazu führen, dass Es ist ungewöhnlich, aber es ist ein grundlegender Angriffsvektor, wenn wir böswilliges Verhalten eines Angreifers berücksichtigen.
Zuverlässigkeit	**Vermeiden Sie unerwartete Verzögerungen:** Wenn nach dem Einschalten einer SPS die Zustandsmaschine in einen Zustand mit einigen Bedingungen initialisiert wird, die den Prozess nicht starten lassen, und der Bediener das System nicht normalisieren kann, müsste ein Techniker das SPS-Programm eingeben, um zu erzwingen, dass die Bedingungen in den gewünschten Zustand versetzt werden, um den Betrieb starten zu können. Dies kann zu Verzögerungen und Produktionsausfällen führen..
Instandhaltung	./.

Referenzen

Standard / Framework	Mapping
MITRE ATT&CK für ICS	**Taktik** : TA0107 Inhibit Response Function **Technik:** T0816 Device Restart/Shutdown
ISA 62443-3-3	**SR 3.6**: Deterministische Ausgabe
ISA 62443-4-2	**CR 3.6**: Deterministische Ausgabe
ISA 62443-4-1	**SVV-1**: Prüfung der Sicherheitsanforderungen

16. Fassen Sie die SPS-Zykluszeiten zusammen und trenden Sie sie auf dem HMI

Fassen Sie die SPS-Zykluszeit alle 2-3 Sekunden zusammen und melden Sie sie dem HMI zur Visualisierung in einem Diagramm.

Sicherheitsziel	Zilegruppe
Überwachung	Integrator / Wartungsdienstleister

Anleitung

Zykluszeiten sind in der Regel Systemvariablen in einer SPS und können zur Zusammenfassung in SPS-Code verwendet werden. Die Zusammenfassung sollte durchgeführt werden, um Durchschnitts-, Spitzen- und Mindestzykluszeiten zu berechnen. Das HMI sollte bei diesen Werten einen Trend darstellen und eine Warnung ausgeben, wenn es signifikante Änderungen gibt.

Die Zykluszeit ist die Zeit, die benötigt wird, um jede Iteration der Logik für die SPS zu berechnen. Bei den Iterationen handelt es sich um die Kombination aus Kontaktplandiagrammen (KOP), Funktionsblockdiagrammen (FBS), Anweisungslisten (IL) und strukturiertem Text (ST). Diese Logikkomponenten können mit den sequentiellen Funktionsplänen (SFC) zusammengefügt werden.

Die Zykluszeiten sollten auf einer SPS konstant sein, es sei denn, es gibt Änderungen an z.B.

- Netzwerkumgebung
- SPS-Logik
- Prozess

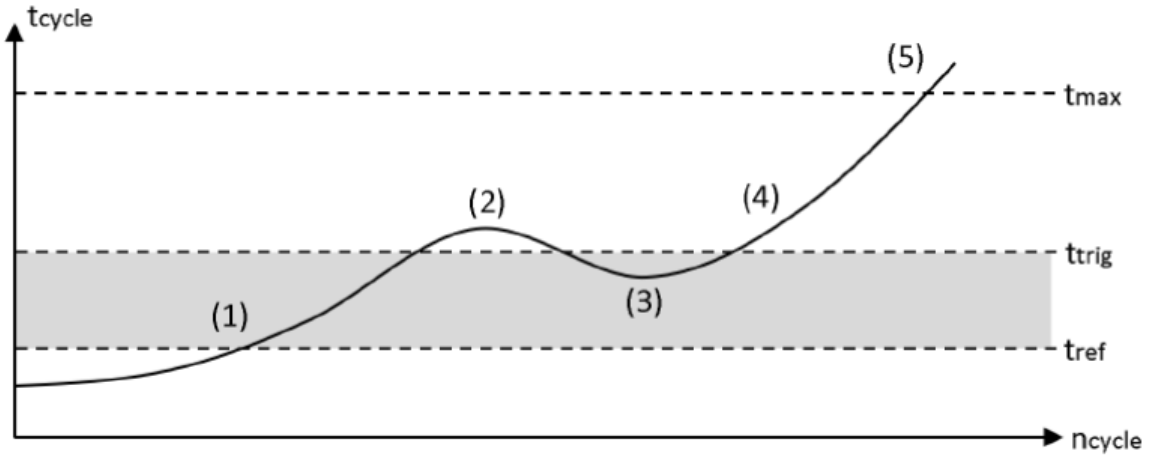
Daher können ungewöhnliche Zykluszeitänderungen ein Indikator dafür sein, dass sich die SPS-Logik geändert hat und somit wertvolle Informationen für Integritätsprüfungen liefern.

Die Visualisierung von Werten im Zeitverlauf mithilfe eines Diagramms bietet eine intuitive Möglichkeit, die Aufmerksamkeit auf Anomalien zu lenken, die mit absoluten Werten schwieriger zu erkennen wären.

Beispiel

Viele SPSen verfügen über eine "maximale Zykluszeit"-Überwachung auf Hardware-Ebene. Überschreitet die Zykluszeit den Maximalwert, setzt die Hardware die CPU auf STOP (5). Natürlich sind sich Angreifer dessen bewusst und werden einen möglichen Angriffscode so schlank wie möglich halten, um die Auswirkungen auf die Gesamtzykluszeit zu minimieren. In einem zusätzlichen Software-Zykluszeitüberwachungsprogramm wird ein Referenzzyklus-t_{ref} als Basiszykluszeit definiert. Da kleine Schwankungen natürlich sind, muss ein akzeptabler Schwellenwert definiert werden (1,3) Die Zyklusüberwachung wird ausgelöst, wenn der Schwellenwert überschritten wird (2,4).

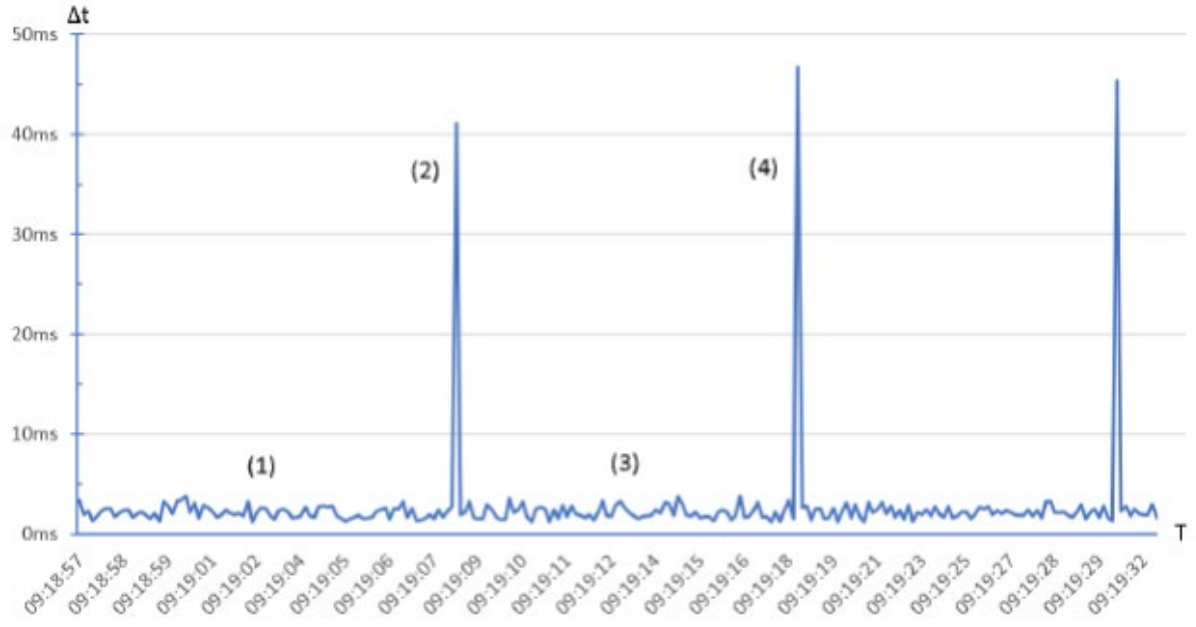
Wenn die Zykluszeiten auf das HMI zugeführt werden, sind hohe CPU-Lasten auf einen Blick sichtbar. Das folgende Beispieldiagramm zeigt ein SPS-Programm mit periodisch ausgeführtem Schadcode. (1,3) im Normalbetrieb akzeptable Zykluszeitschwankungen ("Rauschen") aufweisen, wird auf (2,4) Angriffscode ausgeführt, der die Zykluszeit erhöht.



Jede Abweichung von der Referenzzeit kann wie folgt in einem Logfile gespeichert werden:

SeqNo	Date	UTC Time	Abweichung
1	2019-11-22	09:05:50.021	40,821ms
2	2019-11-22	09:06:00.069	44,391ms
3	2019-11-22	09:06:10.120	44,994ms
4	2019-11-22	09:06:20.166	40,561ms
5	2019-11-22	09:06:30.211	40,725ms

Wenn die Zykluszeiten auf das HMI zugeführt werden, sind hohe CPU-Lasten auf einen Blick sichtbar. Das folgende Beispieldiagramm zeigt ein SPS-Programm mit periodisch ausgeführtem Schadcode. (1,3) im Normalbetrieb akzeptable Zykluszeitschwankungen ("Rauschen") aufweisen, wird auf (2,4) Angriffscode ausgeführt, der die Zykluszeit erhöht.



Begründung

Vorteilhaft für	Begründung
Sicherheit	Zu den Angriffen auf SPS gehören das Ändern der Logik, das Aktivieren eines neuen Programms, das Testen von neuem Code, das Laden eines neuen Prozessrezepts, das Einfügen von Hilfslogik , um Nachrichten zu senden oder eine Funktion zu aktivieren. Für die meisten SPS sind herkömmliche kryptografische Integritätsprüfungen nicht durchführbar. Es ist jedoch gut, Warnung, wenn eine der oben genannten logischen Änderungen eintritt. Da die Zykluszeiten Unter normalen Umständen eher konstant, sind Änderungen der Zykluszeiten ein guter Indikator dafür, dass sich die Logik in einer der oben genannten Logikkomponenten geändert hat.
Zuverlässigkeit	Siehe Sicherheit, aber aus nicht böswilligen Gründen.
Instandhaltung	/

Referenzen

Standard / Framework	Mapping
MITRE ATT&CK für ICS	**Taktik** : TA0104 Execution **Technik:** T0873 Project File Infection
ISA 62443-3-3	**SR 3.4**: Software- und Informationsintegrität
ISA 62443-4-2	**EDR 3.2**: Schutz vor Schadcode
MITRE CWE	**CWE-754:** Unsachgemäße Prüfung auf ungewöhnliche oder außergewöhnliche Bedingungen

17. Protokollieren Sie die SPS-Betriebszeit und trenden Sie sie auf dem HMI

Protokollieren Sie die Betriebszeit der SPS, um zu wissen, wann sie neu gestartet wurde. Trend- und Protokollverfügbarkeit auf dem HMI für die Diagnose.

Sicherheitsziel	Zilegruppe
Überwachung	Integrator / Wartungsdienstleister

Anleitung

Behalten Sie den Überblick über die SPS-Betriebszeit

- in der SPS selbst (wenn die Betriebszeit eine Systemvariable in der SPS ist.
- in der SPS selbst, wenn diese über MIB-2 / eine beliebige SNMP-Implementierung verfügt
- extern mittels z.B. SNMP

Wenn die SPS SNMP mit MIB-2 hat, was sehr häufig vorkommt, ist die OID für die Betriebszeit "sysUpTimeInstance(0)" 1.3.6.1.2.1.1.3. Das Zurücksetzen der Betriebszeit ist ein wichtiger Indikator für SPS-Neustarts. Stellen Sie sicher, dass das HMI bei jeder Art von SPS-Neustart warnt.

Die Betriebszeit mit Fehlercodes zu korrelieren, ist eine gute Diagnosemethode.

Beispiel

/

Begründung

Vorteilhaft für	Begründung
Sicherheit	Der grundlegendste Angriffsvektor für eine SPS besteht darin, sie zum Absturz und / oder Neustart zu zwingen. Für viele SPSen ist das gar nicht so schwer, denn viele SPSen kommen mit unerwarteten Eingaben oder zu viel Verkehr nicht gut zurecht. So können unerwartete Neustarts ein Indikator dafür sein, dass die SPS auf ungewöhnliche Aktionen.
Zuverlässigkeit	SPS-Neustarts eignen sich auch gut für die Diagnose bei Ausfällen und für die Überwachung, an welcher SPS zu welchem Zeitpunkt gearbeitet wird.
Instandhaltung	/

Referenzen

Standard / Framework	Mapping
MITRE ATT&CK für ICS	**Taktik** : TA0107 Inhibit Response Function **Technik:** T0816 Device Restart Shutdown
ISA 62443-3-3	**SR 7.6**: Netzwerk- und Sicherheitskonfigurationseinstellungen
ISA 62443-4-2	**CR 7.6**: Netzwerk- und Sicherheitskonfigurationseinstellungen
MITRE CWE	**CWE-778:** Unzureichende Protokollierung

18. Protokollieren Sie SPS-Hardstopps und trenden Sie sie auf dem HMI

Speichern Sie SPS-Hard-Stop-Ereignisse von Fehlern oder Abschaltungen für den Abruf durch HMI-Alarmsysteme, um sie vor dem Neustart der SPS zu konsultieren. Zeitsynchronisierung für genauere Daten.

Sicherheitsziel	Zilegruppe
Überwachung	Integrator / Wartungsdienstleister

Anleitung

Fehlerereignisse zeigen an, warum eine SPS heruntergefahren wurde, damit das Problem vor einem Neustart behoben werden kann.

Einige SPS haben möglicherweise Fehlercodes aus dem letzten Fall, in dem die SPS fehlerhaft war oder nicht ordnungsgemäß heruntergefahren wurde. Notieren Sie diese Fehler, und löschen Sie sie dann. Es kann eine gute Idee sein, diese Fehler als Informationsdaten an das HMI oder vielleicht an einen Syslog-Server zu melden, wenn diese Funktionen und diese Infrastruktur vorhanden sind.

Die meisten SPSen verfügen auch über eine Art First-Scan-Funktion, die Ereignisse generiert. Es ist ein Verhalten, das fast alle SPS-Geräte in irgendeiner Form haben. Es handelt sich im Grunde um ein oder mehrere Flags oder eine bestimmte Routine, die beim ersten Scan einer SPS ausgeführt wird, nachdem sie "aufgeweckt" wurde. Dieser erste Scan sollte protokolliert und nachverfolgt werden.

Beispiel

/

Begründung

Vorteilhaft für	Begründung
Sicherheit	Protokolle ermöglichen die Fehlerbehebung im Falle eines Vorfalls. Bevor eine SPS in Betrieb genommen wird, insbesondere nach aufgetretenen Problemen, ist es wichtig, sicherzustellen, dass sie vertrauenswürdig ist.
Zuverlässigkeit	Protokolle sind auch gute Quellen für das Debuggen, wenn das Ereignis nicht böswillig verursacht wurde.
Instandhaltung	/

Referenzen

Standard / Framework	Mapping
MITRE ATT&CK für ICS	**Taktik** : TA0107 Inhibit Response Function **Technik:** T0816 Device Restart Shutdown
ISA 62443-3-3	**SR 7.6**: Netzwerk- und Sicherheitskonfigurationseinstellungen
ISA 62443-4-2	**CR 7.6**: Netzwerk- und Sicherheitskonfigurationseinstellungen
MITRE CWE	**CWE-778:** Unzureichende Protokollierung

19. Überwachen Sie die SPS-Speicherauslastung und zeigen Sie einen Trend auf dem HMI

Messen und stellen Sie eine Baseline für die Speicherauslastung für jeden Controller bereit, der in der Produktionsumgebung eingesetzt wird, und stellen Sie einen Trend auf dem HMI bereit.

Sicherheitsziel	Zilegruppe
Überwachung	Integrator / Wartungsdienstleister Asset Owner

Anleitung

Da die Zunahme von Codezeilen in der Logik auch zu einem erhöhten Speicherverbrauch zur Laufzeit führen kann, empfiehlt es sich für SPS-Programmierer, jede Abweichung von der Baseline zu verfolgen und diesem Ereignis eine Alarmklasse zu widmen.

Beispiel

In Rockwell Allen Bradley-SPSen kann eine Baseline auf einer Steuerung erstellt und die Speicherauslastung mit dem RSLogix 5000 Task Monitor Tool verfolgt werden. Nicht nur der Hauptspeicher, sondern auch der I/O-Speicher und der Ladder/Tag-Speicher können anhand von Trends verfolgt werden.

Begründung

Vorteilhaft für	Begründung
Sicherheit	Eine erhöhte Speicherauslastung kann ein Indikator dafür sein, dass die SPS geänderten Code ausführt
Zuverlässigkeit	Die Verfolgung der Speicherauslastung für die laufenden Programme kann nützlich sein, um den Gesamtspeicherverbrauch und einen eventuellen Fehlerzustand für die SPS-Steuerung zu vermeiden.
Instandhaltung	Die Nachverfolgung der Speicherauslastung kann zum Optimieren und Ermitteln der besten Scanzeit für den überwachten Controller, aber auch zur Fehlerbehebung bei Problemen und Problemen im Zusammenhang mit fehlerhaften Zuständen verwendet werden.

Referenzen

Standard / Framework	Mapping
MITRE ATT&CK für ICS	**Taktik** : TA0xx Ausführung
ISA 62443-3-3	**SR 3.4**: Software- und Informationsintegrität
ISA 62443-4-2	**EDR 3.2**: Schutz vor Schadcode

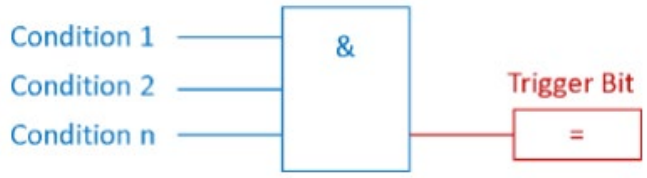
20. Fangen Sie falsch negative und falsch positive Ergebnisse für kritische Warnungen ab

Identifizieren Sie kritische Warnungen und programmieren Sie eine Falle für diese Warnungen. Legen Sie den Trap fest, um die Auslösebedingungen und den Alarmstatus für Abweichungen zu überwachen.

Sicherheitsziel	Zilegruppe
Überwachung	Integrator / Wartungsdienstleister

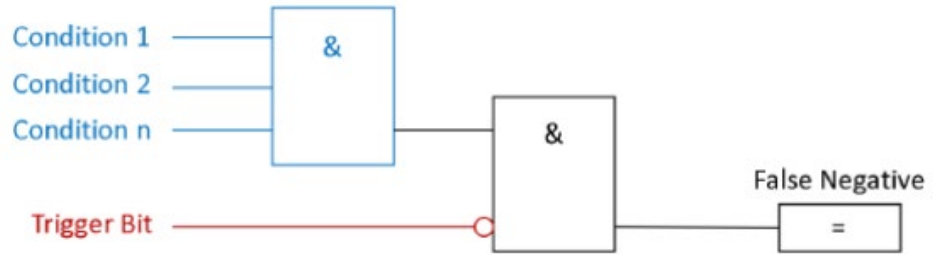
Anleitung

In den meisten Fällen sind Warnungszustände boolesch (True, False) und werden durch bestimmte Bedingungen ausgelöst, wie unten dargestellt. Z.B. wird das Trigger-Bit für den Alarm 'Überdruck' TRUE, wenn Bedingung 1 'Druckschalter 1', Bedingung 2 'Drucksensorwert über kritischer Schwelle' bis n. TRUE sind.



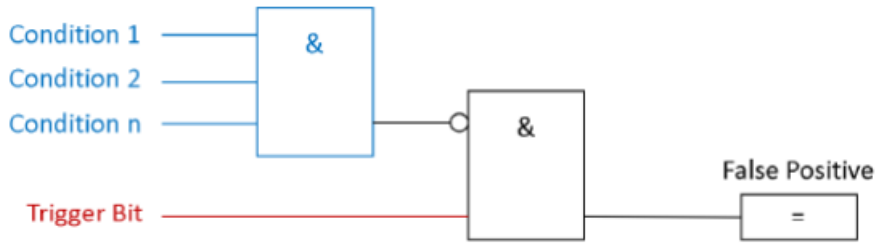
Um einen Angriff zu verschleiern, könnte ein Angreifer das Alert-Trigger-Bit unterdrücken und ein falsch negatives Ergebnis verursachen.

Ein Trap für falsch negative Ergebnisse überwacht die Bedingungen für das Trigger-Bit und das negierte Trigger-Bit selbst. Mit diesem einfachen Aufbau wird ein falsch negatives Ergebnis erkannt. Siehe das folgende Bild:



In anderen Fällen könnte ein Angreifer absichtlich Fehlalarme verursachen, um die Aufmerksamkeit des Prozessbedieners zu beeinträchtigen.

Auf die gleiche Weise wie bei der Falsch-Negativ-Falle können auch Falsch-Positive erkannt werden, indem das Alarm-Trigger-Bit überwacht wird und ob die Trigger-Bedingungen erfüllt sind. Wenn die Bedingungen NICHT erfüllt sind, aber das Trigger-Bit aktiv ist, wird ein falsch positives Ergebnis erkannt: Siehe folgendes Bild:



Beispiel

Beispiel 1: Siemens bietet in seinen Siemens S7-1200/1500 Produkten einen Webserver mit einer Vielzahl von Funktionen, wie z.B. Anzeige des SPS-Zustands, der Zykluszeit oder der Umfangsdatensätze. Es hat auch die Möglichkeit, Datentabellen und Variablen anzuzeigen und zu ändern. Die Zugriffsrechte auf den Webserver können in den SPS-Hardware-Einstellungen geändert werden. Bei falsch konfigurierten Zugriffsrechten könnte sich ein Angreifer Zugriff auf die SPS-Variablen und Datenblöcke verschaffen. Um ein falsch positives Ergebnis zu erzeugen, wählt der Angreifer ein Alert-Trigger-Bit aus und ändert den Zustand.

Beispiel 2: Bei der Triton/Trisys/HatMan-Attacke unterdrückte bösartiger Code die Warnzustände.

Beispiel 3: Ein Bus-Injection-Angriff könnte eine falsch positive Warnung an einen übergeordneten SCADA-Client senden.

Begründung

Vorteilhaft für	Begründung
Sicherheit	Mindert falsch negative oder falsch positive Ergebnisse kritischer Warnmeldungen, die durch einen Angreifer verursacht werden, der seinen Angriff verschleiert (z. B. nicht autorisierter Code, Bus-Injection, Manipulation von zugänglichen SPS-Statustabellen auf ungesicherten Webservern).
Zuverlässigkeit	/
Instandhaltung	/

Referenzen

Standard / Framework	Mapping
MITRE ATT&CK für ICS	**Taktik :** TA0107 Inhibit Response Function **Technik:** T0878 Alarm Suppresion, T0838 Modify Alarm Settings
ISA 62443-3-3	**SR 3.5**: Eingabe-Validierung
ISA 62443-4-2	**CR 3.5**: Validierung der Eingaben
ISA 62443-4-1	**SI-1**: Überprüfung der Sicherheitsimplementierung
MITRE CWE	**CWE-754:** Unsachgemäße Prüfung auf ungewöhnliche oder außergewöhnliche Bedingungen

Über das Projekt "Sichere SPS-Programmierung"

Seit vielen Jahren sind speicherprogrammierbare Steuerungen (SPS) von Haus aus unsicher. Nach mehreren Jahren der Anpassung und Anwendung von Best Practices aus der IT entstanden sichere Protokolle, verschlüsselte Kommunikation, Netzwerksegmentierung usw. Bisher wurde jedoch nicht darauf geachtet, die charakteristischen Merkmale von SPS (oder SCADA/DCS) für die Sicherheit zu nutzen oder SPS unter Berücksichtigung der Sicherheit zu programmieren. Dieses Projekt – inspiriert von den bestehenden Secure Coding Practices für die IT – füllt diese Lücke.

Wer sollte die Secure PLC Coding Practices lesen und implementieren?

Diese Praktiken wurden für Ingenieure geschrieben. Das Ziel dieses Projekts ist es, Ingenieuren, die Software (Kontaktplanlogik, Funktionsdiagramme usw.) erstellen, Richtlinien zur Verfügung zu stellen, um die Sicherheitslage industrieller Steuerungssysteme zu verbessern. Diese Praktiken nutzen die nativ verfügbaren Funktionen in der SPS/DCS. Für die Implementierung dieser Praktiken sind wenig bis gar keine zusätzlichen Softwaretools oder Hardware erforderlich. Sie alle lassen sich in den normalen SPS-Programmier- und Betriebsablauf integrieren. Für die Umsetzung dieser Praktiken ist mehr als Sicherheitsexpertise erforderlich, sondern auch eine gute Kenntnis der zu schützenden SPSen, ihrer Logik und des zugrunde liegenden Prozesses.

Was ist der Geltungsbereich dieser Liste / wie definiert man SPS-Codierung?

Um in den Umfang der Liste der 20 besten sicheren SPS-Codierungspraktiken zu passen, müssen die Praktiken Änderungen beinhalten, die direkt an einer SPS vorgenommen werden. Was Sie in diesem Dokument sehen, ist eine Top-20-Auswahl einer größeren Anzahl potenzieller sicherer SPS-Codierungspraktiken. Es gibt auch zusätzliche Entwurfspraktiken, die sich auf die Gesamtarchitektur, HMIs oder die Dokumentation beziehen. Diese passen nicht in den Bereich der sicheren SPS-Codierung, könnten aber auf einer zukünftigen Liste für sichere SPS-Umgebungen stehen.

Was sind die Vorteile der Anwendung von Secure PLC Coding Practices?

Die Verwendung dieser Praktiken hat eindeutig Sicherheitsvorteile – in der Regel wird entweder die Angriffsfläche reduziert oder eine schnellere Fehlerbehebung ermöglicht, wenn ein Sicherheitsvorfall auftreten sollte. Viele Praxen haben jedoch über die Sicherheit hinaus weitere Vorteile. Einige machen den SPS-Code auch zuverlässiger, einfacher zu debuggen und zu warten, einfacher zu kommunizieren und möglicherweise auch schlanker. Darüber hinaus helfen die sicheren SPS-Codierungspraktiken nicht nur den Benutzern im Falle eines böswilligen Angreifers, sondern machen den SPS-Code auch robuster, um versehentlichen Fehlkonfigurationen oder menschlichem Versagen standzuhalten.

Wer steckt hinter diesem Projekt?

Angefangen hat alles mit Jake Brodskys S4x20-Vortrag "Secure Coding Practices for PLC's".

Im Anschluss an die Konferenz initiierte Dale Peterson das Top-20-Projekt. Jake Brodsky und Sarah Fluchs verbrachten mehrere Stunden am Telefon, um Jakes vorgeschlagene sichere SPS-Codierungspraktiken zu Papier zu bringen.
Danach richteten Dale, Jake und Sarah mit Unterstützung der ISA GCA eine Plattform bei top20.isa.org ein, um zu strukturieren und zusätzlichen Input aus den ICS-Sicherheits- und Ingenieursgemeinschaften zu sammeln.

Die Diskussionen und die Konsolidierung der Praxistexte sowie die Zusammenstellung einer Liste der relevantesten Top-20-Praxen dauerten etwa ein Jahr; Der Prozess wurde

beschleunigt von Vivek Ponnada, der nicht nur Inhalte beisteuerte und überprüfte, sondern auch regelmäßige Anrufe organisierte, bis alle Kommentare zu den Praktiken geklärt waren, Mohamed Abdelmoez Sakesli, der alle Standardreferenzen in einem großen Aufwand hinzufügte, das MITRE CWE-Team, das die CWE-Referenzen in letzter Minute zur Verfügung stellte, Sarah, die das Dokument zusammengestellt hat, das Sie jetzt lesen, und Jake, Dale, John Cusimano, Dirk Rotermund, Josh Ruff, Thomas Rabenstein, Gus Serino, Walter Speth, Agustin Valencia Gil-Ortega, Marcel Rick-Cen und Al Ratheesh R, die während der regelmäßigen Telefonate Beiträge geleistet haben.

Liste der Unterstützer

Das Secure PLC Coding Project ist und bleibt eine echte Gemeinschaftsleistung, die ohne unzählige Mitwirkende, die ihre Zeit und ihr SPS-/Sicherheitswissen großzügig zur Verfügung gestellt haben, nicht möglich gewesen wäre. Insgesamt 943 Nutzer haben sich auf der Plattform registriert, um zu diskutieren und Beiträge zu leisten. Hier ist eine alphabetische Liste aller Personen, die ausdrücklich zugestimmt haben, namentlich genannt zu werden. Vielen Dank an alle, die sich die Zeit genommen haben, dieses Projekt zu unterstützen!

Aagam Shah
Adam Patray
Agustín Valencia Gil-Ortega
- Aitor García Almiñana
Alec Summers
Al Ratheesh. Wer
Andreas Falk
Anton Shipulin
Arkaitz produzierte
Carlos Olave
Chris van den Hooven
Chris Sistrunk
Christos Alexopoulos
Cris DeWitt
Dale Peterson
Dene Yandle
Dennis Verschoor
Dirk Rotermund
Edorta Echave García
Gananand Kini
George Alex Holburn
Gus Serino
Hakija Agic
Hector Medrano
Josie Houghton
Jozef Sulwinski
Juan Pablo Angel Espejo
Khalid Ansari
Marc Weber
Marcel Rick-Cen
Martin Huddleston
Massimiliano Zonta
Matthew Loong
Matthias Müller
Michael Thompson
Michal Stepien
Miguel Ángel Frías
Mohamed Abdelmoez Sakesli
Mond Sieben Monde
Nahuel Iglesias
Nalini Kanth
Narasimha S. Himakuntala
Omar Morando
Oscar J. Delgado-Melo
Päivi Brunou
Peter Donnelly
Peter Jackson
Ravindra Deshkulkarni

Heiko Rudolph	Rick Booij
Isiah Jones	Robert Albach
Jacob Brodsky	Russischer Priester
Javier Pérez Quezada	Sarah Fluchs
J-D Bamford	Sergej Biberdorf
Joe Weiss	Stephan Beirer
John Cusimano	Steve Christey Coley
John Hoyt	Thomas Rabenstein
John Powell	Tim Gale
John Kingsley	Vivek Ponnada
Joseph J. Januszewski	Vytautas Butrimas
Josh Ruff	Walter Speth

Besonderer Dank geht an diese Organisationen, die großzügig Infrastruktur für das Projektteam zur Verfügung gestellt haben, wie z.B. Domains, Hosting, Webdesign und Grafikdesign:

Deutsche Übersetzung:

Rudolf Preuß Koordinator IT/OT

rudolf.preuss@actemium.de

Actemium Controlmatic West GmbH

www.actemium.de