softScheck
we identify vulnerabilities others don't

Ihre Strategie ist falsch

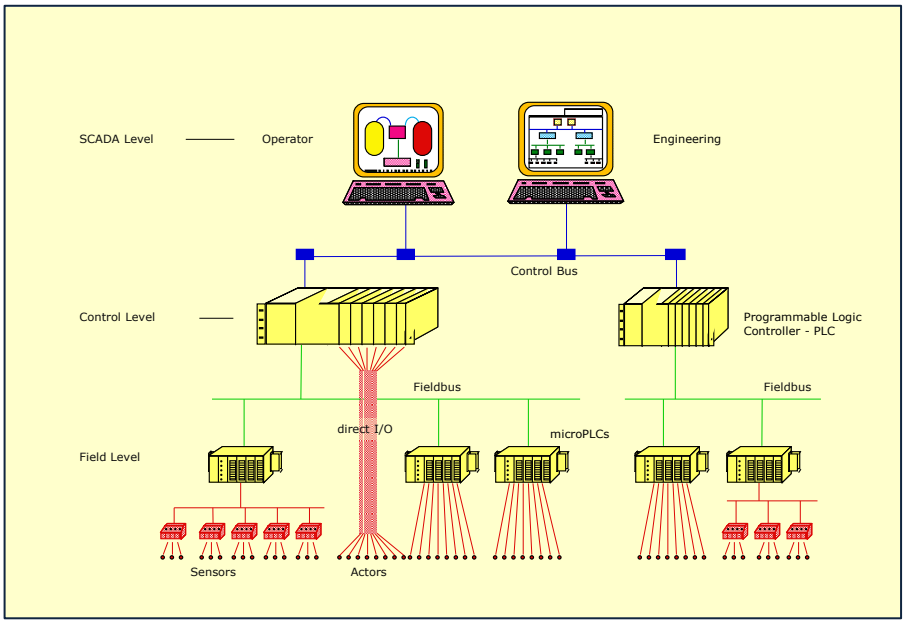
Industrial Control Systems Security

Hartmut Pohl[1]

Seit Stuxnet, Duqu, Flame, Mahdi, Gauss, Shamoon etc. und deren Nachfolger, Varianten und Derivate sind Industriesteuerungen in aller Munde – aber es gibt noch viele Punkte zu bedenken, um zu zukunftssicheren Lösungen zu kommen. Dabei wird der Punkt Sicherheit eher zurückhaltend erwähnt. Dies gilt z.B. für Produktionsanlagen, Leittechnik und modernes Gebäudemanagement – und auch Fernwartungs- und Fernsteuerungsanlagen. Wichtig aber ist: Jeder Sicherheitsvorfall ist kostenträchtig und jeder bekannt gewordene erfolgreiche Angriff ist imageschädigend.

Allerdings ist es völlig falsch, nur die aggressiver werdenden Angriffe zu analysieren. Richtig ist vielmehr, die von den Angriffen ausgenutzten bekannten Sicherheitslücken zu patchen und insbesondere die noch nicht veröffentlichten Sicherheitslücken zu identifizieren und zu beheben.

Die Risiken liegen im fehlerhaften Ablauf der Software, (leicht) fahrlässiger Bedienung der Software bis hin zu gezielten Angriffen aus dem Unternehmen oder - z.T. mit erheblichem Aufwand - auch aus dem Internet. Ein Internet-Anschluss ist aber für einen erfolgreichen Angriff gar nicht notwendig - wie die erfolgreichen Angriffe auf Industriesteuerungen mit dem Wurm Stuxnet ebenfalls zeigen.



Industrial Control System – ICS

Sicherheit ist also nicht nur ein theoretisches Problem. Für die Zukunft muss also erwartet werden, dass nicht nur weit entfernte Uran-Zentrifugen angegriffen werden sondern unsere eigenen Prozesssteuerungen.

Industrial Control Systems (ICS) – also Automatisierungs-, Prozesssteuerungs- oder Leitsysteme –bestehen aus einer aufgabenspezifischen Anlage und aus standardisierten Automatisierungsgeräten wie speicherprogrammierbaren Steuerungen (SPS), Human-Machine-Interfaces (HMI) und Industrie-PCs (IPC), deren Software und Steuerung (Hardware) fehlerbehaftet sein kann.

Unsere Untersuchungen zeigen, dass häufig ältere, fehlerbehaftete Versionen weit verbreitete Speicher-programmierbare Steuerungen (SPS) eingesetzt werden.

---

[1] Prof. Dr. Hartmut Pohl, geschäftsführender Gesellschafter softScheck GmbH, Köln - Büro: Sankt Augustin
www.softScheck.com

Besonders gefährdet sind SPS mit älteren Firmware-Versionen, da oftmals bereits Exploits veröffentlicht sind, die die Sicherheitslücken ohne umfangreiches Expertenwissen ausnutzen. Häufig kann damit ein kritischer Angriff auf das SCADA-System gestartet werden, der z.B. das Verhalten von Antrieben manipuliert oder die Geräte sogar abschaltet.

Tests mit gängigen Vulnerability Assessment Tools zeigen viele angreifbare Sicherheitslücken an; z.T. überschneiden sich die Ergebnisse allerdings so stark, dass der Einsatz weniger Tools hinreichend ist. Angezeigt werden z.B. Cross-Site-Scripting und SNMP-Sicherheitslücken mit hohem bis sehr hohem Gefährdungsgrad.

Außerdem wurden erfolgreich DoS-Angriffe auf SPS mit aktueller Firmware durchgeführt, sodass weiterhin Aktualisierungsbedarf besteht.

Die beiden folgenden Schritte sind unverzichtbarer Bestandteil einer Sicherheitsstrategie zur Erhöhung des Sicherheitsniveaus eines Industrial Control Systems:

1. Das Aufspielen der möglichst jüngsten SPS-Firmware-Version (in der erkannte Sicherheitslücken behoben sind) ist also genauso unverzichtbar wie die Behebung oder auch Umgehung (z.B. Sperrung von Ports) veröffentlichter - aber noch nicht vom Hersteller gepatchter - Sicherheitslücken.

   Identifizierte Sicherheitslücken resultieren auch aus werksseitig (zu wenig sicher?) vorkonfigurierten Geräten. Hier sollten Anwender nicht die benutzerfreundliche Einstellung für die SPS übernehmen sondern eine sicherere Einstellung vornehmen.

2. Gleichermaßen muss die Anwendungs- und Individualsoftware auf bisher nicht erkannte (unveröffentlichte) Sicherheitslücken überprüft werden; die erfolgreiche Ausnutzung solcher Sicherheitslücken haben Stuxnet & Co. vorgeführt.

Zur Identifizierung bisher nicht-bekannter Sicherheitslücken werden insbesondere die im Folgenden kurz erläuterten drei Verfahren eingesetzt:

1. Architectural Analysis - Threat Modeling

Bereits im Design muss Sicherheit berücksichtigt werden: Insoweit beginnt die Identifizierung und der Nachweis von Sicherheitslücken mit der Analyse der Dokumentation. Dazu gehört auch eine Untersuchung der Programmablaufpläne und der Datenflussdiagramme von und zu allen Kommunikationspartnern – vgl. die Abbildung.

2. Static Source Code Analysis

Analysiert wird der Source Code (White-Box-Test) der Zielsoftware ohne ihn auszuführen – bis hin zur semantischen Analyse. Damit ist es möglich, auch komplexe Fehler, die etwa auf Race Conditions, Deadlocks oder falscher Pointerverwaltung basieren, zu identifizieren.

3. Dynamic Analysis mit „Fuzzing"

Mit dieser Black-Box-Technik (also ohne Kenntnis des Quellcodes) werden bisher nicht erkannte Sicherheitslücken kostensparend in Anwendungen oder Anwendungskomponenten identifiziert. Dabei werden geeignete Testdaten in das Zielprogramm eingespeist. Führt die Verarbeitung dieser Daten zu einem fehlerhaften Verhalten (Crash, hoher Verbrauch an Ressourcen wie Rechenzeit) des Zielprogramms, so wird dieses anomale Verhalten mit Hilfe eines - im Fuzzer integrierten oder separaten - Monitoring-Tools detektiert, protokolliert und voranalysiert, so dass damit Sicherheitslücken nachgewiesen werden können.

Erst durch den Einsatz eines umfassenden Sicherheitskonzepts, dass die drei Verfahren Architectural Analysis - Threat Modeling, Static Source Code Analysis und Dynamic Analysis - Fuzzing mit geeigneter Tool-Unterstützung adressiert, können die Forderungen nach vertrauenswürdiger, Sicherheitslücken-freier Software erfüllt werden.