Security-Testing für industrielle Automatisierungssysteme

ISuTest: Ein modulares Framework zur automatisierten Unterstützung bei der Entwicklung von sicheren Automatisierungskomponenten

Dipl.Inform. **S. Pfrang**, M.Sc. **D. Meier**,
Fraunhofer IOSB, Karlsruhe

Kurzfassung

Am Fraunhofer IOSB in Karlsruhe wurde das Security-Testing-Framework ISuTest entworfen und in einem Prototypen umgesetzt. ISuTest ermöglicht das automatisierte Überprüfen der Security-Eigenschaften von Automatisierungssystemen mithilfe von definierten Testfällen und baut auf dem auf Office-IT ausgerichteten „OpenVAS“ (Open Vulnerability Assessment System) auf. ISuTest ermöglicht Angriffe auf Automatisierungssysteme durchzuführen und daraus resultierende Reaktionen zu prüfen. Anhand dieser kann die Auswirkung auf die Funktionsfähigkeit und damit die Anfälligkeit gegenüber einem Angriff untersucht werden. Durch sein offenes und modulares Design ermöglicht ISuTest einfaches und effizientes Testen nahezu beliebiger Automatisierungskomponenten.

Abstract

At the Fraunhofer IOSB in Karlsruhe a Security-Testing-Framework called ISuTest was designed and implemented. ISuTest enables the automated testing of the security properties of industrial automation equipment by using predefined test cases. It is compatible to the „OpenVAS” (Open Vulnerability Assessment System) security testing framework. ISuTest makes it possible to conduct attacks on automation systems and capture the resulting reactions. This reveals the impact of attacks and possible vulnerabilities of the tested systems. Easy and efficient testing of arbitrary automation systems is enabled by the open and modular design of the ISuTest-Framework.

1. Motivation

Die fortschreitende Vernetzung in der Automatisierung unter dem Einfluss von „Industrie 4.0“ und dem „(Industrial) Internet of Things“ führt zu einem größeren Bedrohungsfeld im Bereich der IT-Sicherheit. Immer einfacher zugängliche Systeme und Komponenten haben dabei

einen großen Einfluss auf die IT-Security von Gesamtanlagen. Die Sicherheitseigenschaften einzelner Komponenten gewinnen daher immer mehr an Bedeutung für ihren Einsatz in modernen Industrieanlagen. Diese Eigenschaften umfassen dabei beispielsweise die Unterstützung von sicheren Kommunikationsprotokollen, welche Integrität, Verfügbarkeit und Vertraulichkeit ermöglichen. Auch ein robustes Verhalten im Falle von ungeplanten Eingaben gehört zu den Sicherheitseigenschaften, da solche Eingaben durch Fehlfunktionen entstehen können, aber auch gezielt von Angreifern provoziert werden können. Darüber hinaus müssen alle von einer Komponente angebotenen Dienste auf ihre Sicherheitseigenschaften untersucht werden, also sowohl Dienste aus dem klassischen IT-Bereich (z. B. Webserver), als auch Dienste aus der Automatisierung (z. B. industrielle Kommunikationsprotokolle). Um die Sicherheit von Automatisierungskomponenten bestimmen zu können, braucht es daher Systeme, die diese Sicherheitseigenschaften automatisiert überprüfen können. Aufgrund der Komplexität und Spezialisierung von Automatisierungskomponenten, gestaltet sich dies im industriellen Umfeld schwieriger als beispielsweise im Bereich der Office-IT, für die entsprechende Werkzeuge und Frameworks bereits existieren. Die Vielzahl an unterschiedlichsten Schnittstellen und Standards industrieller Komponenten stellen solche Systeme vor besondere Herausforderungen.

2. Grundlagen

Industrielle Anlagen und Systeme weisen andere Eigenschaften auf als Systeme der traditionellen Office-IT. Dies hat Auswirkungen auf den Umgang mit IT-Sicherheitsanforderungen. In der unterschiedlichen Priorisierung der klassischen Informationssicherheitsziele wird dieser Unterschied deutlich.

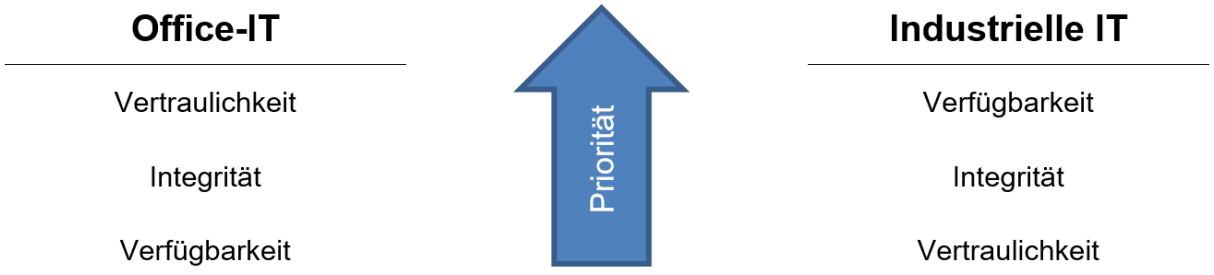

Bild 1: Unterschiede in der Priorität der Schutzziele der Informationssicherheit in der industriellen und klassischen Office-IT

Nach dem Schutzziel der Vertraulichkeit dürfen nur autorisierte Stellen Informationen aus dem System gewinnen können. Dieses Schutzziel kann durch die Verwendung von Datenverschlüsselung erreicht werden. Integrität dient der Sicherstellung, dass Informationen nicht durch unautorisierte Stellen verändert werden können bzw. eine mögliche Veränderung erkannt wird. So kann das Fälschen von Daten unterbunden werden. Mit der Verfügbarkeit wird sichergestellt, dass unautorisierte Stellen nicht die Benutzung von Systemen und Diensten durch autorisierte Stellen verhindern können.

Wie in Bild 1 dargestellt, wird im Gegensatz zur Office-IT, der Vertraulichkeit in der industriellen IT eine geringere Priorität als der Integrität und Verfügbarkeit zugeschrieben. Im industriellen Umfeld hat die Verfügbarkeit des Systems die höchste Priorität. Daher ist es möglich, dass die anderen Schutzziele vernachlässigt werden, falls die Verfügbarkeit des Systems durch diese beeinträchtigt werden könnte. Dies folgt aus der Tatsache, dass die Effektivität in der industriellen Anlage direkt mit deren aktiven Betrieb verknüpft ist und daher Stillstandzeiten zu vermeiden sind. Auch können viele industrielle Prozess nicht einfach unterbrochen werden oder das Wiederanfahren des Prozesses nimmt weitere Zeit in Anspruch. Diese veränderte Priorität der Schutzziele muss bei der Entwicklung von Sicherheitstests für industrielle Systeme Beachtung finden.

Sicherheitstests dienen der Identifizierung von Schwachstellen im untersuchten System. Allgemein kann zwischen drei Arten von Schwachstellen unterschieden werden:

1. **Schwachstellen im Entwurf** gründen auf Fehlern, die bereits bei der Spezifizierung eines Systems oder Protokolls entstanden sind. Folgen Implementierungen dem Entwurf, ist auch in diesen zwangsläufig der Fehler enthalten. Diese Art von Schwachstellen zählen zu den am schwersten behebbaren, da Lösungen zu Inkompatibilitäten mit alten Implementierungen führen können.
2. **Schwachstellen in der Implementierung** entstehen durch die nicht korrekte Umsetzung eines Entwurfs. Ein häufiger Fehler für diese Schwachstellen ist der Umgang mit im Entwurf nicht spezifizierten Vorgängen.
3. **Schwachstellen in der Konfiguration** gehören zu den häufigsten Fehlerquellen. Sie können, trotz eines sicher entworfenen und implementierten Systems, zu Sicherheitslücken führen. Diese Art von Schwachstellen gehört zu den am einfachsten behebbaren.

Zur Identifizierung von Schwachstellen der verschiedenen Arten sind unterschiedliche Vorgehensweisen bei den Testverfahren notwendig.

Das ISA Security Compliance Institute (ISASecure) beschreibt Tests zur Zertifizierung von industriellen Systemen nach dem IEC 62443 Standard [1]. Die Testverfahren sind dabei in

drei Teile eingeteilt: dem Software Development Security Assurance (SDSA), dem Functional Security Assessment (FSA) und dem Communication Robustness Testing (CRT). Dabei konzentrieren sich die CRTs auf das Testen von Systemverhalten im Hinblick auf grundlegende Netzwerkprotokolle, wie Ethernet, TCP und IP. Für die Protokolle ist eine feste Anzahl von Testfällen definiert, die vorgeben, welche Pakete an das System gesendet werden müssen und welche Reaktionen zu erwarten sind. Es ist darüber hinaus festgelegt, welche Reaktionen ein Nicht-Bestehen des Tests zur Folge haben. Das Bestehen der Tests wird auch mithilfe der Überprüfung sogenannter *Essential Services* ermittelt. Dabei werden vom zu testenden System angebotene Dienste fortlaufend getestet um zu ermitteln, ob die Testfälle die Funktion dieser Dienste ungewollt beeinflussen. Die CRTs decken dabei nur die grundlegenden Kommunikationsprotokolle ab und umfassen keine speziell im industriellen Umfeld verbreiteten Protokolle.

Die meisten Testumgebungen folgen einem einfachen, grundlegenden Aufbau. Dieser sieht ein System vor, welches für die Durchführung der Tests verantwortlich ist (*Test-Device*). Dieses Test-Device sendet dann *Eingaben* an das zu testende Gerät oder System, das sogenannte *Device under Test (DUT)*. Die *Ausgaben* des DUT werden vom Test-Device aufgenommen und ermöglichen die Auswertung der Tests. Dieser Aufbau ist in Bild [2] dargestellt.

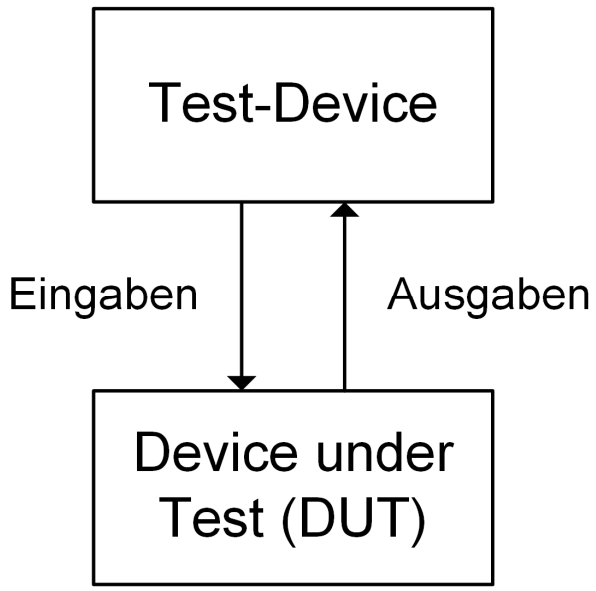

Bild 2: Grundlegender Aufbau einer Testumgebung, bei der das Test-Device Eingaben an das Device under Test (DUT) sendet und die Ausgaben des DUT analysiert.

3. Stand der Technik

Im Office-Bereich wird das Testen von Systemen auf Sicherheitsschwachstellen bereits seit längerem durch etablierte Lösungen ermöglicht. In der industriellen IT haben sich solche Systeme noch nicht in gleicher Weise durchgesetzt.

TOCSSiC (Teststand On Control System Security At Cern) ist ein Testaufbau, welcher in [2] vorgestellt wird. In diesem Aufbau werden die Security-Testing Programme Nessus und Netwox verwendet, um Automatisierungskomponenten mit den gleichen Tests wie in Office-IT-Systeme zu überprüfen. Dabei werden Angriffe auf Standardkommunikationsprotokolle durchgeführt. Auf die speziellen Eigenschaften und Technologien von Automatisierungssystemen wird dabei nicht eingegangen.

Die Achilles Plattform von Wurldtech [3] hat das Ziel des Testens und Zertifizierens von Automatisierungssystemen und besteht aus einer speziellen Hardware- und Softwareeinheit. Es handelt sich hierbei um ein kommerzielles und geschlossenes System und ist dadurch schwer erweiterbar. Die Achilles Plattform ist zudem zur Zertifizierung nach dem ISASecure Conformity Assessment Program [1] zugelassen.

OpenVAS (Open Vulnerability Assessment System) [4] stellt ein offenes Framework zur Durchführung von Sicherheitsuntersuchungen von Office-IT-Systemen dar und ist der freie Nachfolger des bereits erwähnten Nessus-Systems. Es ermöglicht die Verwaltung von Scanaufträgen und die Bereitstellung von Berichten und bietet bereits eine Großzahl an verfügbaren Tests an. Es kann durch eine offene Schnittstelle um weitere Scanner erweitert werden.

Das Testsystem VuTAT ist Ergebnis des Forschungsprojekts „Vulnerability Tests of Automation Technology components“ [5]. Das Testsystem erweitert OpenVAS und macht so die Ausführung von Python-Skripten möglich, die auch digitale und analoge I/O-Signale an ein DUT senden können. Durch die verwendete Einbindung in OpenVAS ist VuTAT nicht kompatibel mit neueren OpenVAS-Versionen. Durch die Aufteilung von Dateneingabe und Datenerfassung sind keine Testskripte möglich, die beide Definitionen enthalten. Außerdem ist VuTAT stark auf das vorgestellte Referenzsystem angepasst und kann daher schwieriger auf andere Plattformen konvertiert werden.

4. Anforderungen

Die bei dem Aufbau eines Testsystems relevanten Anforderungen ergeben sich aus den grundlegenden Anforderungen an eine Testplattform, den speziellen Eigenschaften der industriellen IT, sowie den Erfahrungen aus den vorhandenen Projekten. Für das ISuTest-Framework ergeben sich daraus folgende Anforderungen:

1. **Anpassbarkeit:** Automatisierungssysteme basieren auf den unterschiedlichsten Technologien und sind von verschiedensten Herstellern erhältlich. Daher muss das Test-Framework Unterstützung für unterschiedliche Technologien bieten und dementsprechend anpassbar sein.
2. **Dynamische Eingabe:** Zum Testen von Systemen werden diese mit unterschiedlichen Eingaben versorgt und die Ausgaben und Reaktionen des Systems darauf erfasst. Da Automatisierungssysteme die unterschiedlichsten Eingabemöglichkeiten haben, muss das Test-Framework über ein System zur Eingabedatenerstellung sowie Einspeisung verfügen.
3. **Ausgabeerfassung:** Das Test-Framework muss die Ausgaben des getesteten Gerätes erfassen und auswerten können, um die Reaktion feststellen zu können.
4. **Beschreibung von Testfällen:** Das Testen folgt vorgegebenen Testfällen. Das Test-Framework muss die Abarbeitung solch vorgegebener Testfälle unterstützen.
5. **Erweiterbarkeit:** Damit das System auch mit zukünftigen Technologien verwendet werden kann, muss es erweiterbar für neue Ein- und Ausgabemöglichkeiten sowie neue Testszenarien sein.

Zu den grundlegenden Anforderungen kommen noch zusätzliche Anforderungen hinzu, die die Verwendung des Test-Frameworks erleichtern sollen:

6. **Testrepositorium:** Testfälle sollen über Repositorien verwaltet werden, von denen sie abgerufen werden können. Das ermöglicht den einfachen Austausch dieser Testfälle.
7. **Automatisierung:** Die Durchführung von Tests soll vollautomatisch erfolgen und Vorgänge, wie das Zurücksetzen des DUT zwischen verschiedenen Tests, sollen durch das Test-Framework durchgeführt werden.
8. **Open Source:** Der Quellcode des Frameworks soll zur freien Verfügung stehen um die Erweiterung durch eine große Gemeinschaft zu fördern und damit eine breite Abdeckung von Technologien und vielfältige Testfälle zu ermöglichen.

5. Aufbau des Frameworks

Die zentrale Komponente des Frameworks ist der *ISuTest-Scanner*. Dieser hat die Aufgabe, Tests durchzuführen und die Testergebnisse auszuwerten. Die durchzuführenden Tests werden in *Test-Definitionen* spezifiziert und beschrieben. Die Test-Definitionen werden als Skripte in einer eigenen Domain Specific Language (DSL) erstellt, die speziell auf die Anforderungen und Fähigkeiten von ISuTest ausgelegt ist. Die Anweisungen der Test-Spezifikationen werden durch den Scanner mithilfe von *Modulen* ausgeführt. Diese Module ermöglichen die Ein- und Ausgaben des Frameworks an und vom DUT. Die Ergebnisse kön-

nen anschließend mithilfe von *Evaluations* ausgewertet werden. Der ISuTest-Scanner ist über das *OpenVAS Scanner Protocol* an OpenVAS angeschlossen und kann darüber Scanaufträge von OpenVAS annehmen und Ergebnisse zur Verfügung stellen. Die für das Durchführen der Eingaben und Erfassen der Ausgaben eines DUT zuständigen Module ermöglichen ISuTest den wirksamen Einsatz im Automatisierungsumfeld. Sie ermöglichen den Umgang mit verschiedensten Technologien, Kommunikationsprotokollen und Datentypen. Das Erzeugen der Eingaben und Erfassen der Ausgaben wird dabei in den Modulen durch *Commands* abgebildet. Diese führen Operationen aus und können über Parameter angepasst werden. Sie ähneln klassischen Funktionen und Methoden, die in der Programmierung verwendet werden. Eine weitere Eigenschaft der *Commands* ist, dass diese zeitgerichtet sind. Dies bedeutet, dass mehrere *Commands* vorbereitet werden können und dann in einer vorgegebenen Sequenz abgearbeitet werden. Dadurch können z. B. Simultanitäten in der Eingabe von Daten oder der Erfassung abgebildet werden. Auch ist es so möglich, Befehle, deren Verarbeitung unterschiedlich viel Zeit in Anspruch nehmen können, trotzdem zu gezielten Zeitpunkten auszuführen.
Die Ausgaben von getesteten Systemen müssen nach der Erfassung verarbeitet werden, um eine Aussage über das erfolgreiche Absolvieren von Tests treffen zu können. Bei Tests ist dies oft der Vergleich zwischen der aufgezeichneten Reaktion eines DUT und dem erwarteten Verhalten. Aufgrund der Unterschiedlichkeit von Testzielen und der auszuwertenden Daten, wird diese Funktionalität durch einen eigenen Teil des Frameworks abgedeckt, den *Evaluations*. Diese sorgen für den passenden Vergleich der erwarteten und aufgezeichneten Daten und geben ein entsprechendes Ergebnis zurück.

6. Evaluation

Das in Abschnitt 5 beschriebene ISuTest-Framework wurde prototypisch in einem Demonstrationsaufbau umgesetzt und evaluiert, wie in [6] beschrieben. Der schematische Aufbau ist in Bild 3 dargestellt. Der zugehörige Hardwareaufbau ist in Bild 4 dargestellt. Auf dem Trägergestellt befinden sich auf der linken Seite das Netzteil zur Versorgung der Komponenten mit 24V. Daran schließt sich ein steuerbarer Schaltaktor an. Dieser ermöglicht das Trennen der Stromversorgung des DUT während eines Testzyklus. Das Trennen dient dazu, das DUT automatisiert durch das Zurücksetzen in einen definierten Ausgangszustand zu bringen. Daneben schließt eine speicherprogrammierbare Steuerung (SPS) an, mit deren Hilfe die I/O-Schnittstellen des DUT angesprochen werden und dessen I/O-Ausgaben wieder erfasst werden können.

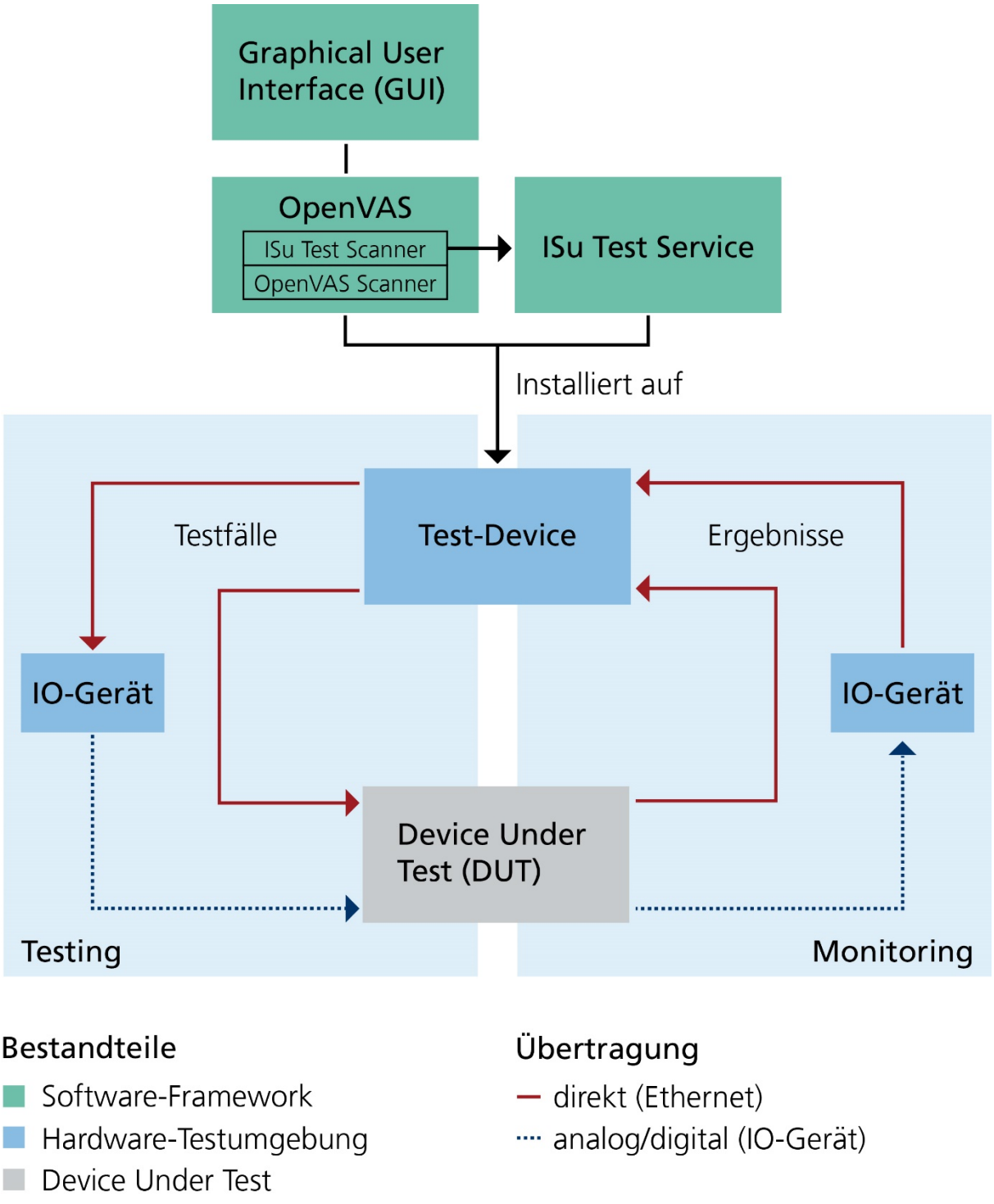

Bild 3: Aufbau des ISuTest-Frameworks und Anbindung an OpenVAS.

Neben einem Netzwerkswitch auf der rechten Seite, welcher zur Anbindung der einzelnen Komponenten dient, befindet sich das Test-Device. Das Test-Device ist ein PC-System, auf

dem das ISuTest-Framework installiert ist. Vor den Test-Komponenten befindet sich das DUT, im abgebildeten Fall eine SPS (Bild 4).

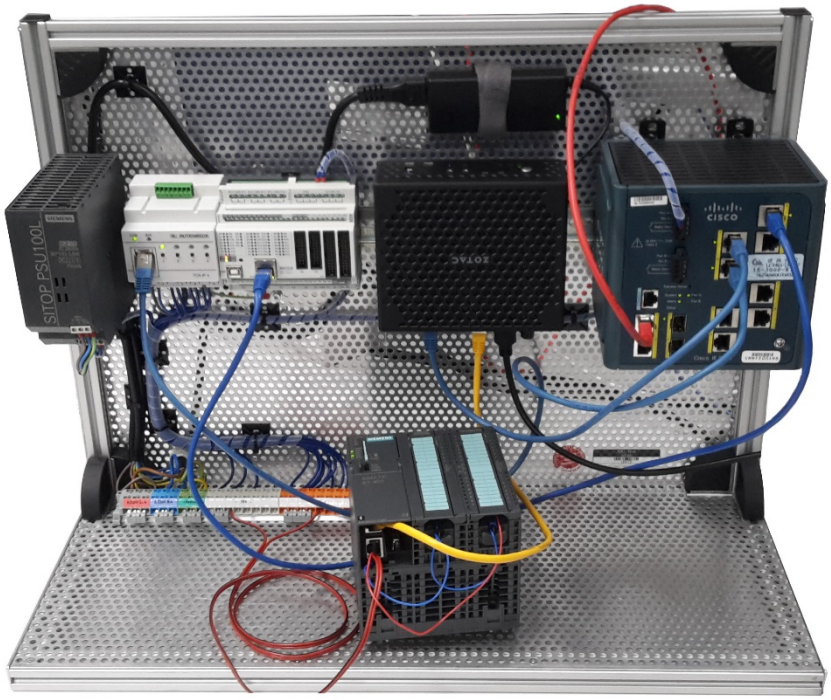

Bild 4: Hardwareaufbau einer ISuTest-Demonstrationsumgebung.

Zur Evaluierung der Funktionalität des ISuTest-Frameworks wurden Testfälle des CRT in Testskripte umgesetzt. Diese wurden anschließend automatisch über einen Scanauftrag des auf dem Test-Device laufenden OpenVAS durchgeführt. Dabei wurden zur Ermittlung des Testergebnisses Essential Services des DUT ausgewertet um Fehlfunktionen festzustellen.
Durch diese prototpyische Umsetzung konnte die Funktionalität des ISuTest-Frameworks überprüft werden. Alle in Abschnitt 4 aufgestellten grundlegenden Anforderungen konnten erfüllt werden. Durch den Aufbau des ISuTest-Frameworks aus Modulen, Evaluationen und flexiblen Testskripten können unterschiedlichste Technologien abgedeckt werden (1). Die Module ermöglichen die Ein- und Ausgabe von beliebigen Daten (2, 3), unabhängig von der Übertragungsform, wie Netzwerkpakete oder direkte I/O-Signale. Durch die Beschreibung der Testabläufe durch Testskripte können beliebig komplizierte Testszenarien umgesetzt werden (4). Durch die Verwendung offener Technologien und den modularen Aufbau des Frameworks kann das System einfach erweitert werden (5). Die Verwendung von dedizierter Repositorien für Module, Evaluationen und Testskripte ermöglicht den einfachen Austausch

und die Weiterentwicklung des Frameworks (6). Durch den Demonstrationsaufbau konnte auch die Fähigkeit zur automatischen Durchführung von Testsequenzen gezeigt werden (7). Da das System nur freie und offene Komponenten enthält, kann der Quellcode frei zur Verfügung gestellt werden (8). Dies ist für eine zukünftige Revision des Frameworks vorgesehen.

7. Zusammenfassung und Ausblick

Die voranschreitende Vernetzung in industriellen Anlagen stellt die Untersuchung von Automatisierungssystemen und IT-Sicherheitsaspekten in ein neues Licht. Die Komplexität und Variantenvielfalt dieser Systeme stellt dabei eine große Herausforderung für Testsysteme in diesem Bereich dar. Die in Abschnitt 4 beschriebenen Anforderungen sind nötig, um eine effektive und effiziente Test-Umgebung in diesem Bereich etablieren zu können. Das vorgestellte ISuTest-Framework ist nach diesen Anforderungen entwickelt worden und deckt diese bereits größtenteils ab. Seine modulare und offene Struktur macht es leicht weiterentwickelbar. Durch den in Abschnitt 6 gezeigten prototypischen Aufbau konnte das System auch im praktischen Einsatz getestet werden und hat dort seine Funktionsfähigkeit unter Beweis gestellt. Die Plattform kann nun dafür verwendet werden, Automatisierungskomponenten, wie speicherprogrammierbare Steuerungen oder Peripheriesysteme, auf Sicherheitseigenschaften und Verwundbarkeiten zu testen.

Der präsentierte Stand des ISuTest-Frameworks stellt dabei die initiale Version der Plattform dar und soll auch zukünftig funktional erweitert werden. Zu den derzeitigen Arbeiten zählen:

- **Fuzzy-Testing-Erweiterung:** Unter Fuzzy-Testing versteht man das Testen eines Systems durch zufällige Eingaben. Dadurch können insbesondere Implementierungsfehler aufgedeckt werden. Effektive Fuzzing-Werkzeuge zeichnen sich dadurch aus, dass sie gezielt Änderungen an gültigen Eingaben vornehmen um Randfälle der Implementierung abzuprüfen.
- **Grafischer Testfall-Editor:** Um das Beschreiben und Erstellen von Testfällen weiter zu vereinfachen wird ein grafischer Editor erstellt. Mithilfe dieses Werkzeuges wird es möglich sein, Testfälle basierend auf Modulen und Evaluationen in einfacher Weise zu erzeugen. Damit wird es auch Personen ohne weitere Programmierkenntnisse möglich sein, Testfälle zu erstellen.

[1] EDSA-310 ISA Security Compliance Institute - Embedded Device Security Assurance - ISASecure certification requirements, Spezifikation, Automation Standards Compliance Institute, 2015

[2] Lüders, S.: Control systems under attack?, 10th ICALEPCS Int. Conf. on Accelerator & Large Expt. Physics Control Systems, Genf 2005

[3] Achilles Test Platform Fact Sheet, Produktblatt, Wurldtech, 2014

[4] Intevation GmbH. About OpenVAS. http://openvas.org/about.html, 2017. [Online; Zugriff 09.04.2017].

[5] Heiss, S. und Brand, J.-C. und Doehring, T.: Schwachstellenanalyse von Automatisierungskomponenten - Entwicklung eines Frameworks zur automatisierten Untersuchung von Ethernet-basierten Komponenten der Automatisierungstechnik (AT) zur Aufdeckung von IT-Sicherheitsschwachstellen (Abschlussbericht), 2013

[6] Kautz, V.: Konzeption und Implementierung eines auf OpenVAS basierenden Security-Testing-Frameworks für industrielle Automatisierungssysteme, Karlsruher Institut für Technologie / Fraunhofer IOSB Masterarbeit. Karlsruhe 2016