Effizienter Schutz der IT-Sicherheit auf der Feldebene von Automatisierungssystemen

Von der Fakultät Informatik, Elektrotechnik und Informationstechnik
der Universität Stuttgart zur Erlangung der Würde eines
Doktor-Ingenieurs (Dr.-Ing.) genehmigte Abhandlung

Vorgelegt von
Felix Gutbrodt
aus Stuttgart

Hauptberichter:	Prof. Dr.-Ing. Dr. h. c. Peter Göhner
Mitberichter:	Prof. Dr.-Ing. Andreas Kirstädter
Tag der Einreichung:	13.02.2009
Tag der mündlichen Prüfung:	18.01.2010

Institut für Automatisierungs- und Softwaretechnik
der Universität Stuttgart

2010

IAS-Forschungsberichte

Band 2/2010

Felix Gutbrodt

Effizienter Schutz der IT-Sicherheit auf der Feldebene von Automatisierungssystemen

D 93 (Diss. Universität Stuttgart)

Shaker Verlag
Aachen 2010

Bibliografische Information der Deutschen Nationalbibliothek
Die Deutsche Nationalbibliothek verzeichnet diese Publikation in der Deutschen Nationalbibliografie; detaillierte bibliografische Daten sind im Internet über http://dnb.d-nb.de abrufbar.

Zugl.: Stuttgart, Univ., Diss., 2010

Printed in Germany.

ISBN 978-3-8322-8908-9
ISSN 1610-4781

Shaker Verlag GmbH • Postfach 101818 • 52018 Aachen
Telefon: 02407 / 95 96 - 0 • Telefax: 02407 / 95 96 - 9
Internet: www.shaker.de • E-Mail: info@shaker.de

Die vorliegende Arbeit entstand während meiner Tätigkeit als wissenschaftlicher Mitarbeiter am Institut für Automatisierungs- und Softwaretechnik (IAS) der Universität Stuttgart.

Mein besonderer Dank gilt Herrn Prof. Dr.-Ing. Dr. h. c. Peter Göhner für den Freiraum bei der Auswahl des Dissertationsthemas, die fortwährende Unterstützung während der Entstehung der Arbeit, die zahlreichen wertvollen Anregungen und kritischen Fragen sowie die Übernahme des Hauptberichts.

Herrn Prof. Dr.-Ing. Andreas Kirstädter danke ich für das Interesse an meiner Arbeit und die Übernahme des Mitberichts.

Bei meinen ehemaligen Kolleginnen und Kollegen am IAS bedanke ich mich für die kollegiale Atmosphäre und die tolle Zusammenarbeit. Insbesondere danke ich Dr. Nasser Jazdi, der mich auf diesen Weg geführt und stets freundschaftlich begleitet hat, sowie den ehemaligen Kollegen, mit denen ich mein Dissertationsthema regelmäßig diskutieren durfte: Dr. Thomas Stiedl, Dr. Paul Linder und Simon Kunz. Ganz besonders bedanke ich mich bei Michael Wedel für die kritische und gründliche Durchsicht des Manuskripts.

Ebenfalls danke ich allen Studierenden, die im Rahmen ihrer Studien- und Diplomarbeiten einen großen Beitrag zu dieser Arbeit geleistet haben.

Sehr dankbar bin ich meinen Eltern für die langjährige Unterstützung während meiner gesamten Ausbildungszeit, insbesondere meiner verstorbenen Mutter, die mich stets, aber insbesondere auch während meiner Doktorandenzeit sehr unterstützt hat.

Schließlich danke ich meiner Freundin Martina ganz herzlich für ihr Verständnis, ihre Geduld und ihren Ansporn, der entscheidend zum Gelingen dieser Arbeit beigetragen hat.

Stuttgart, im Januar 2010

Felix Gutbrodt

Inhaltsverzeichnis

Abbildungsverzeichnis

Tabellenverzeichnis

Abkürzungsverzeichnis

µC	**Mikroc**ontroller
3DES	**Triple D**ata **E**ncryption **S**tandard
AES	**A**dvanced **E**ncryption **S**tandard
AT	**A**utomatisierungs**t**echnik
BACnet	**B**uilding **A**utomation and **C**ontrol **Net**work
CAN	**C**ontroller **A**rea **N**etwork
CPU	**C**entral **P**rocessing **U**nit
DSA	**D**igital **S**ignature **A**lgorithm
ECU	**E**lectronic **C**ontrol **U**nit
EIB/KNX	**E**uropean **I**nstallation **B**us / **Ko**n**nex**
EtherCAT	**Ether**net for **C**ontrol **A**utomation **T**echnology
FG	**F**eld**g**erät
GB	**G**iga**b**yte (10^9 Byte)
GHz	**G**iga**h**ert**z**
HART	**H**ighway **A**ddressable **R**emote **T**ransducer
ID	**Id**entifikation
IP	**I**nternet **P**rotocol
IT	**I**nformations**t**echnik
kB	**K**ilo**b**yte (10^3 Byte)
LIN	**L**ocal **I**nterconnect **N**etwork
LLC	**L**ogical **L**ink **C**ontrol
LON	**L**ocal **O**perating **N**etwork
MAC	**M**edia **A**ccess **C**ontrol
MB	**M**ega**b**yte (10^6 Byte)
MD5	**M**essage **D**igest **5**
MHz	**M**ega**h**ert**z**
MOST	**M**edia **O**riented **S**ystems **T**ransport

OTP	**O**ne **T**ime **P**ad
PC	**P**ersonal **C**omputer
PDA	**P**ersonal **D**igital **A**ssistant
PROFIBUS	**Pro**cess **Fi**eld **Bus**
PROFIBUS DP	**PROFIBUS D**ezentrale **P**eripherie
PROFIBUS PA	**PROFIBUS P**rozess**a**utomatisierung
PROFINET	**PROFI**BUS über Ether**net**
PROFINET IO	**PROFINET I**nput **O**utput
RAM	**R**andom **A**ccess **M**emory
RC4	**R**on's **C**ode **4**
ROM	**R**ead **O**nly **M**emory
RSA	**R**ivest, **S**hamir, **A**dleman
SHA	**S**ecure **H**ash **A**lgorithm
SSL	**S**ecure **S**ockets **L**ayer
TCP	**T**ransmission **C**ontrol **P**rotocol
TLS	**T**ransport **L**ayer **S**ecurity
UDP	**U**ser **D**atagram **P**rotocol
USB	**U**niversal **S**erial **B**us
VPN	**V**irtual **P**rivate **N**etwork
WiMax	**W**orldwide **I**nteroperability for **M**icrowave **Acc**ess
WLAN	**W**ireless **L**ocal **A**rea **N**etwork
XML	E**x**tensible **M**arkup **L**anguage

Begriffsverzeichnis

Aktor: Einheit zur Umsetzung von elektrischen oder optischen Stellsignalen in physikalische Stellgrößen, die zur Beeinflussung des vom Automatisierungssystem gesteuerten technischen Prozesses dienen.

Angriff: Intentionale (d. h. absichtliche, vorsätzliche, zielgerichtete) Durchführung einer Handlung, welche die IT-Sicherheit bedroht.

Authentifizierung: Überprüfung der von einer Entität angegebenen Identität.

Automatisierungssystem: Technisches System, Rechner- und Kommunikationssystem und Menschen zur Leitung und Bedienung des auf dem technischen System ablaufenden technischen Prozesses.

Autorisierung: Zuweisung von Rechten an identifizierte Entitäten.

Bedrohung: Mögliches Ereignis, durch das ein Schaden entstehen kann.

Buffer Overflow: Überlauf eines zu kleinen Speicherbereichs, wodurch andere Speicherbereiche überschrieben werden.

Byte: Zusammengehörige Datenmenge mit Länge 8 bit.

Feldbus: Serielles Kommunikationsmedium zum Austausch von Daten auf der Feldebene von Automatisierungssystemen.

Feldebene: Hierarchieebene des Automatisierungssystems, die dem technischen Prozess am nächsten ist.

Feldgerät: Systemelement der Feldebene, bestehend aus einem oder mehreren Sensoren bzw. Aktoren zur Erfassung bzw. Beeinflussung von Prozessgrößen, einem Rechner zur Verarbeitung der Prozessgrößen und einer Feldbusanbindung zur Kommunikation.

Gateway: Übergangspunkt zwischen technologisch verschiedenen Kommunikationssystemen.

Gefahr: Zustand eines Systems, in dem das Risiko für das Auftreten von Schäden größer ist als das größte noch vertretbare Risiko.

Gefährdung: Ereignis, das aufgrund einer Schwachstelle einen Schaden verursachen kann.

IT-Sicherheit: Informationstechnik-Sicherheit, siehe *Security*.

Kommunikation: Übertragung von Daten über ein physikalisches Medium.

Malware: Software, welche Funktionen enthält, deren Ausführung vom Benutzer eines Rechners nicht gewünscht wird, z. B. Computerviren, Computerwürmer und trojanische Pferde.

Mikrocontroller: Vollständiges Mikroprozessorsystem auf einem Chip, in den außer einer CPU auch Speicher und Peripheriekomponenten integriert sind.

Risiko: Produkt aus Wahrscheinlichkeit für das Auftreten eines Schadens multipliziert mit dem Ausmaß des Schadens. Risiken können quantitativ oder qualitativ angegeben werden.

Safety: Zustand eines Systems, in dem das Risiko für das Auftreten von Schäden durch Gefahren, die vom Betrieb des Systems ausgehen und Menschen oder Umwelt gefährden, geringer ist als das größte noch vertretbare Risiko. Die deutsche Bezeichnung für *Safety* lautet *technische Sicherheit*.

Schutzfunktionalität: Funktionalität zur Absicherung einer Schwachstelle eines IT-Systems.

Schutzmechanismus: Ablauf eines Vorgangs oder mehrerer ineinandergreifender Vorgänge mit dem Zweck, ein System zu schützen. Schutzmechanismen realisieren Schutzfunktionalitäten, indem sie konkrete Ablaufvorschriften umsetzen.

Schwachstelle: Eigenschaft eines Systems, durch die eine Bedrohung einen Schaden verursachen kann.

Security: Zustand eines Systems, in dem das Risiko für das Auftreten von Schäden, die aufgrund von Bedrohungen aus der Umgebung des Systems auftreten, geringer ist als das größte noch vertretbare Risiko. Sicherheit im Sinne von Security wird im Deutschen als *IT-Sicherheit* bezeichnet.

Sensor: Einheit zur Erfassung von physikalischen Prozessgrößen und deren Umwandlung in elektrische oder optische Messsignale mit dem Zweck der Verarbeitung dieser Messsignale durch Automatisierungscomputer.

Sicherheit: Zustand eines Systems, in dem das Risiko für das Auftreten von Schäden geringer ist als das größte noch vertretbare Risiko. Der Begriff Sicherheit umfasst sowohl Schäden, die vom Betrieb des Systems ausgehen (*Safety*), als auch Schäden, die aufgrund von Bedrohungen aus der Umgebung hervorgerufen werden (*Security*).

Smart Sensor/Actuator: Sensor/Aktor mit Fähigkeiten zur Informationsverarbeitung (z. B. zur Einheitenumwandlung) und Feldbusanbindung. Smart Sensors/Actuators sind somit *Feldgeräte*.

Standard-IT: Informationstechnische Systeme, die in Büros, Rechenzentren usw. eingesetzt werden.

Steuergerät: In der Domäne der Kraftfahrzeugelektronik verwendeter Begriff für Feldgerät.

Verlässlichkeit: Eigenschaft eines Systems, die es erlaubt, volles Vertrauen in die Funktion des Systems zu setzen.

Virtual Private Network: In einem Virtual Private Network werden Verbindungen eines Kommunikationsnetzes über ein anderes Netz geleitet („getunnelt“). Diese Verbindungen werden durch kryptografische Kommunikationsprotokolle geschützt.

Zusammenfassung

Die zunehmende Verwendung von vernetzten Rechnern auf der Feldebene von Automatisierungssystemen bedingt Schwachstellen in Bezug auf die IT-Sicherheit der Feldebene. Die Feldebene dient zur Anbindung eines Automatisierungssystems an den technischen Prozess, der in einem technischen System abläuft. Aufgrund dieser Aufgabe sind das technische System und die Feldebene eng miteinander gekoppelt und häufig örtlich weiträumig verteilt oder öffentlich zugänglich. Dadurch können unautorisierte Personen einen physischen Zugriff auf die Feldebene erlangen und Angriffe auf die IT-Sicherheit der Feldebene ausführen. Derartige Angriffe umfassen beispielsweise die Störung von Kommunikationsvorgängen oder die Veränderung von in Software realisierten Funktionen. Insbesondere ist es auch möglich, Funktionen zur Verhinderung gefährlicher Situationen, wie z. B. Notabschaltungen, zu manipulieren. Durch solche Manipulationen können Fehlfunktionen des Automatisierungssystems und somit auch des geführten technischen Prozesses absichtlich herbeigeführt werden. Derartige Fehlfunktionen bedrohen die technische Sicherheit, die Zuverlässigkeit sowie die Verfügbarkeit von Automatisierungssystemen und unterminieren dadurch die Eigenschaften, die erforderlich sind, um volles Vertrauen in die Funktion eines Automatisierungssystems setzen zu können. Aus diesem Grund ist der Schutz der IT-Sicherheit auf der Feldebene zwingend erforderlich.

Wegen der Exponiertheit und der räumlichen Ausdehnung der Feldebene von Automatisierungssystemen können Angriffe auf die IT-Sicherheit der Feldebene in vielen Fällen nicht verhindert werden. Stattdessen muss die Feldebene selbst so abgesichert werden, dass Angriffe auf die IT-Sicherheit der Feldebene keine Fehlfunktionen hervorrufen. Aufgrund von wirtschaftlichen Randbedingungen darf diese Absicherung keine oder nur geringe Kosten verursachen und sie muss wirksam sein gegen die umfangreichen Möglichkeiten der Manipulation, die einem zielgerichtet agierenden Angreifer zur Verfügung stehen. Das im Rahmen der vorliegenden Arbeit entwickelte Konzept des effizienten Schutzes der IT-Sicherheit auf der Feldebene adaptiert daher bewährte informationstechnische Funktionsprinzipien der IT-Sicherheit so, dass diese in Form von Software-Schutzmechanismen innerhalb von typischen Systemelementen der Feldebene ausgeführt werden können. Dies wird durch eine Softwarearchitektur erreicht, welche eine variable Zusammenstellung und eine Mehrfachverwendung der Schutzmechanismen ermöglicht. Dadurch kann der Schutz so an die vorliegenden Gefährdungen angepasst werden, dass ausschließlich der minimal benötigte Schutzumfang und die minimal benötigte Stärke dieses Schutzes eingesetzt werden müssen, womit ein sehr geringer Ressourcenverbrauch erreicht wird. Das Konzept ist zudem so ausgelegt, dass es Echtzeitanforderungen erfüllt und an unterschiedliche Technologien angepasst werden kann.

Abstract

The increasing application of networked computers at the field level of industrial automation systems causes weaknesses concerning the security of the field level. The field level realizes the connection of an industrial automation system to the technical process, which is executed in a technical system. Due to this task, the technical system and the field level are closely coupled and often widely distributed or publicly accessible. As a result, unauthorized persons may obtain physical access to the field level and conduct attacks against the security of the field level. Such attacks could be the distortion of communication or the change of functions that are realized as software. In particular, it is also possible to annul functions designed to prevent dangerous situations, e.g. emergency shutdown. This can result in intentionally effectuated malfunctions of the industrial automation system as well as of the technical system. Such malfunctions endanger the safety, the reliability, and the availability of industrial automation systems. Thus, they undermine all properties that are required to place full confidence in the functioning of an industrial automation system. For this reason, security protection of the field level is imperative.

Due to the exposure and the spatial distribution of many industrial automation systems, attacks on the security of the field level can in many cases not be prevented. Instead, the field level itself must be hardened in such a way that attacks against field level security do not cause malfunctions. Due to economical constraints, the protection has to entail no or only low costs and it has to be effective against the comprehensive possibilities of manipulation that are available to a determined attacker. The concept of efficient security protection at field level, which has been developed within this thesis, adapts well-proven functional principles in a way that they can be executed in the form of software protection mechanisms by typical system elements of the field level. This is achieved by a software architecture that allows for variable combination and multiple use of the protection mechanisms. Thereby, the protection can be adapted to the hazards at hand, so that only the minimally required scope and strength of protection can be applied and the associated resource overhead is minimized. Furthermore, the concept is designed to fulfill real-time requirements and can be adapted to different technologies.

1 Einleitung und Motivation

„The only way of discovering the limits of the possible
is to venture a little way past them into the impossible.“
(Arthur C. Clarke)

1.1 IT-Sicherheit als Eigenschaft verlässlicher Automatisierungssysteme

Am 28. November 2008 gelang es einem Unbekannten, eine der vier Turbinen des britischen Kohlekraftwerks Kingsnorth außer Gang zu setzen [Heis08a]. Sicherheitseinrichtungen im Wert von 12 Mio. Pfund, die das Kraftwerk zu dem am besten geschützten Kohlekraftwerk Großbritanniens machen [Vida08], konnten den Saboteur nicht aufhalten. Er überwand zwei jeweils drei Meter hohe, stacheldrahtbewehrte, elektrisch geladene Zäune und drang dann in die mit Überwachungskameras gesicherte Hauptturbinenhalle ein. Auswertungen der Kameraaufzeichnungen ergaben, dass der Saboteur „mit einem Gerät hantierte“ [Heis08a] und damit innerhalb von Minuten eine der 500-MW-Turbinen abschaltete [Knap08].

Kraftfahrzeugdiebe bedienen sich heute bei Diebstählen von Premium-Kfz nicht mehr ausschließlich rein mechanischer Hilfsmittel. Stattdessen werden modernste elektronische Werkzeuge eingesetzt, um Zugang zum Kfz-Innenraum zu erlangen [EKM+08] und Anti-Diebstahleinrichtungen, z. B. elektronische Wegfahrsperren, zu deaktivieren [Masc08].

Auch selbst vergleichsweise einfache Automatisierungssysteme sind heutzutage durch Manipulationen bedroht. So wies die Software eines Kaffeeautomaten der Firma Jura eine Schwachstelle auf, durch welche die Mixturen der bestellten Getränke von Hackern manipuliert werden konnten [Heis08b] [Wrig08].

Diese Beispiele zeigen ebenso wie weitere Untersuchungen [Bach03] [Bach04] [MoSc05] [CERT07], dass Manipulationen der IT-Sicherheit in der Automatisierungstechnik prinzipiell ebenso möglich sind wie in der Standard-IT, also beispielsweise im Bürobereich. Manipulationen der IT-Sicherheit eines Automatisierungssystems können wegen der Anbindung an ein technisches System physische Auswirkungen auf die Umgebung hervorrufen. Dadurch besteht die Gefahr, dass Schäden an technischen Einrichtungen verursacht werden, Umweltschäden auftreten oder Menschen verletzt oder getötet werden.

Durch Störungen der IT-Sicherheit von Automatisierungssystemen können die technische Sicherheit, die Zuverlässigkeit und die Verfügbarkeit beeinträchtigt werden. Beispielsweise können automatisierungstechnische Schutzeinrichtungen deaktiviert oder Parameter verändert

werden. Die Verlässlichkeit eines Automatisierungssystems erfordert somit zwingend auch die IT-Sicherheit des Automatisierungssystems.

1.2 IT-Sicherheit in der Automatisierungstechnik – Problembewusstsein und Herausforderungen

In der Automatisierungstechnik werden unterschiedliche Technologien zur Realisierung von Rechner- und Kommunikationssystemen verwendet. In den letzten Jahren werden zur Betriebsführung, Anlagenführung und Maschinenführung zunehmend Standard-IT-Bausteine anstelle spezialisierter Prozessleitsysteme verwendet [Enst02]. So kommen Personal Computer (PCs), Microsoft-Windows-Betriebssysteme und Kommunikationssysteme des Ethernet-Standards zum Einsatz. Zur Optimierung von organisatorischen Abläufen werden derartige Bausteine miteinander verbunden, dazu werden zunehmend auch Internet-Technologien verwendet [Scha05].

Bei derartigen Systemen der Automatisierungstechnik müssen die gleichen Bedrohungen der IT-Sicherheit betrachtet werden wie in anderen IT-Systemen auch [Naed04], beispielsweise Hacker-Angriffe über das Internet. Aus diesem Grund ist die IT-Sicherheit auch in der Automatisierungstechnik zu einer wichtigen Thematik geworden, die bei der Entwicklung und im Betrieb von Automatisierungssystemen berücksichtigt werden muss [BeMo07]. Bei den Systemelementen von Automatisierungssystemen, welche mit Standard-IT-Technologien umgesetzt sind, können die IT-Sicherheitskonzepte der Standard-IT mit geringen Anpassungen übernommen werden [Naed03].

Ein Automatisierungssystem führt stets einen technischen Prozess, u. a. durch Messungen von Prozessgrößen und durch Stelleingriffe in den technischen Prozess. Diese Aufgaben werden auf der *Feldebene* von Automatisierungssystemen durchgeführt. Die Feldebene realisiert die Anbindung des Automatisierungssystems an den technischen Prozess. Dabei herrschen auf der Feldebene spezialisierte Technologien (Feldgeräte, Feldbusse) vor und es gelten besondere Randbedingungen (z. B. geringe Ressourcen der eingesetzten Rechner).

In vielen Betrachtungen der IT-Sicherheit von Automatisierungssystemen wird die Feldebene heute nicht explizit berücksichtigt oder es wird davon ausgegangen, dass durch Absicherung der Standard-IT einschließlich der Verbindungspunkte, welche die Feldebene mit dem restlichen Automatisierungssystem verbinden, auch die IT-Sicherheit der Feldebene gewährleistet ist [Käst07]. Es existieren jedoch Bedrohungen, welche die IT-Sicherheit auf der Feldebene trotz dieser Absicherungen gefährden. Beispiele für derartige Bedrohungen sind die Veränderung von Kilometerständen und die Freischaltung von Motorfunktionen bei Kraftfahrzeugen [Grel05]. Solche Angriffe lassen sich einfach auf die Feldebene anderer Arten von Automatisierungssystemen, z. B. Fertigungssysteme oder verfahrenstechnische Anlagen, übertragen [Plew06] [Rusc06] [Pech07].

Da die Feldebene das Bindeglied zum technischen Prozess darstellt, kann durch die Manipulation der IT-Sicherheit auf der Feldebene das Verhalten des technischen Prozesses direkt beeinflusst werden, beispielsweise durch die Vorgabe falscher Stellgrößen. Durch die Deaktivierung oder Umgehung von automatisierungstechnischen Schutzeinrichtungen kann der technische Prozess schwere Schäden verursachen, beispielsweise Schäden an Menschen in der Umgebung des technischen Prozesses, Umweltschäden und finanzielle Schäden. Daher muss die IT-Sicherheit auch auf der Feldebene von Automatisierungssystemen geschützt werden.

Eine Absicherung der Feldebene muss die dort vorherrschenden Eigenschaften und Randbedingungen berücksichtigen, die sich von denen der Standard-IT unterscheiden, beispielsweise durch die geringen Ressourcen von Systemelementen der Feldebene und die Echtzeitanforderungen des technischen Prozesses. Aufgrund dieser Unterschiede können Schutzkonzepte, die in der Standard-IT zum Einsatz kommen, nicht auf die Feldebene übertragen werden. So wird beispielsweise davon ausgegangen, dass Feldgeräte ca. 10- bis 40-mal leistungsfähiger sein müssten, um einen derartigen Schutz ausführen zu können [John08]. Eine derartige Leistungssteigerung würde jedoch hohe Mehrkosten verursachen.

1.3 Ziel des Schutzes der IT-Sicherheit auf der Feldebene

Die Gefährdung von Menschen und der Umwelt sowie die Gefahr finanzieller Schäden durch unautorisierte Manipulationen an den Systemelementen der Feldebene von Automatisierungssystemen bilden den Ausgangspunkt für die vorliegende Arbeit. Die zu entwickelnden Konzepte und Verfahren sollen die Systemelemente der Feldebene auf eine Art und Weise schützen, dass Übergänge des Automatisierungssystems in unerwünschte (z. B. gefährliche), durch Angriffe auf die IT-Sicherheit der Feldebene hervorgerufene Fehlerzustände unterbleiben. Dabei sind die spezifischen Eigenschaften der Feldebene zu berücksichtigen und es ist zu beachten, dass die Fähigkeiten, die zur Erbringung der Aufgaben auf der Feldebene erforderlich sind (z. B. Echtzeitfähigkeit), nicht in negativer Weise beeinträchtigt werden. Innerhalb dieser Zielsetzung wird folgenden Aspekten ein besonderer Stellenwert beigemessen:

Wirksamkeit des Schutzes

Ein wesentliches Kriterium bei Maßnahmen zur Absicherung der IT-Sicherheit ist die Wirksamkeit des Schutzes, welche zwei Dimensionen abdecken muss. Zum einen sind alle möglichen Bedrohungen der IT-Sicherheit auf der Feldebene zu berücksichtigen. Zweitens ist im Gegensatz zu Problemstellungen der technischen Sicherheit mit Manipulationen zu rechnen, die zielgerichtet und böswillig durchgeführt werden. Ein Schutz gegen solche Bedrohungen muss auf eine Art und Weise konzipiert sein, dass er nicht ausgehebelt werden kann.

Systemunabhängigkeit des Schutzes

Auf der Feldebene werden zahlreiche unterschiedliche Technologien eingesetzt. Diese unterscheiden sich z. B. in ihren Funktionsweisen und den Ressourcen, die ihnen zur Verfügung stehen. Das Ziel der vorliegenden Arbeit ist daher, die Absicherung der IT-Sicherheit auf der Feldebene unabhängig von den eingesetzten Technologien zu ermöglichen. Dazu soll das zu entwickelnde Konzept von spezifischen Technologien abstrahieren und somit zum Schutz aller Technologien auf der Feldebene eingesetzt werden können.

Wirtschaftlichkeit des Schutzes

Die Hersteller von Automatisierungssystemen stehen untereinander im Wettbewerb. Daher muss der Schutz der IT-Sicherheit auf der Feldebene mit möglichst geringen Kosten umsetzbar sein. Um die Wettbewerbsfähigkeit zu gewährleisten, wird daher ein Ansatz benötigt, der sowohl bei dem Engineering zur Integration des Schutzes in die Feldebene eines spezifischen Automatisierungssystems als auch bei der Anschaffung und dem Betrieb der Systemelemente der Feldebene keine oder nur geringe Mehrkosten verursacht.

1.4 Gliederung der Arbeit

Die vorliegende Arbeit ist in 11 Kapitel untergliedert.

Kapitel 2 enthält die für das Verständnis der vorliegenden Arbeit erforderlichen Grundlagen der Automatisierungstechnik. Das Kapitel beschreibt den Aufbau von Automatisierungssystemen und geht dann detaillierter auf die Feldebene ein. Es werden deren Eigenschaften und Randbedingungen beschrieben und es wird dargelegt, aus welchen Systemelementen sich die Feldebene zusammensetzt.

Das dritte Kapitel gibt eine Einführung in die IT-Sicherheit. Dazu werden Begrifflichkeiten der IT-Sicherheit definiert, Gefährdungen der IT-Sicherheit dargestellt und es wird beschrieben, wie der Schutz der IT-Sicherheit prinzipiell realisiert wird.

Kapitel 4 stellt dar, wie die IT-Sicherheit auf der Feldebene angegriffen werden kann. Es werden dazu unterschiedliche Möglichkeiten des Zugriffs auf die Systemelemente der Feldebene und die Auswirkungen von Angriffen auf die IT-Sicherheit der Feldebene erläutert.

Kapitel 5 befasst sich mit existierenden Ansätzen zum Schutz der IT-Sicherheit. Dabei werden zunächst Anforderungen aufgestellt, welche von dem Schutz der IT-Sicherheit auf der Feldebene erfüllt werden müssen. Daraufhin bewertet das Kapitel die existierenden Ansätze bezüglich der Erfüllung dieser Anforderungen. Dabei wird sich herausstellen, dass keiner der existierenden Ansätze alle Anforderungen zum Schutz der IT-Sicherheit auf der Feldebene erfüllen kann. Daraus wird der Bedarf nach einem neuartigen Konzept abgeleitet.

In Kapitel 6 wird zunächst der Grundgedanke des Schutzes der IT-Sicherheit auf der Feldebene festgelegt. Basierend auf diesem Grundgedanken werden die Anforderungen so präzisiert, dass der Entwurf und die Validierung des neuartigen Konzepts ausgeführt werden können.

Im siebten Kapitel wird das Konzept zum Schutz der IT-Sicherheit auf der Feldebene hergeleitet. Dazu wird analysiert, welche Funktionalitäten zur Absicherung der Feldebene erforderlich sind und wie diese unter den gegebenen Randbedingungen der Feldebene prinzipiell umzusetzen sind. Es wird eine Architektur definiert, welche die Bausteine und deren Verbindungen festlegt, aus denen der Schutz aufgebaut wird.

Zur Anwendung des Schutzkonzepts auf die Feldebene eines konkreten Automatisierungssystems beschreibt das Kapitel 8 eine Vorgehensweise, mit welcher der Schutzbedarf systematisch ermittelt und der Schutz individuell angepasst werden kann. Zudem wird dargestellt, wie die Validierung des Schutzes durchzuführen ist.

In Kapitel 9 wird die Realisierung des Schutzes beschrieben. Dazu wird dargestellt, dass Softwarekomponenten über wesentliche Vorteile für die Implementierung des Schutzes verfügen. Darauf aufbauend wird eine Softwarekomponententechnologie ausgewählt und die Realisierung von Schnittstellen der Softwarekomponenten und der Softwarekomponenten selbst beschrieben. In diesem Kapitel wird auch der Ressourcenbedarf der Softwarekomponenten aufgeführt, die im Rahmen dieser Arbeit entwickelt wurden.

Das zehnte Kapitel diskutiert zwei Anwendungsfälle zum Zweck der Validierung des Schutzkonzepts gegen die Anforderungen an den Schutz der IT-Sicherheit auf der Feldebene. Es wird dazu die IT-Sicherheit auf der Feldebene zweier Demonstrationsanlagen geschützt. Dabei wird die in Kapitel 8 dargestellte Vorgehensweise herangezogen, um den jeweiligen Schutzbedarf zu ermitteln, die Absicherung durchzuführen und schließlich den Schutz hinsichtlich der Wirksamkeit und des Ressourcenkonsums zu evaluieren.

Kapitel 11 fasst schließlich die wesentlichen Aspekte der Arbeit zusammen, beschreibt die Voraussetzungen und Grenzen für den Einsatz des vorgeschlagenen Konzepts und führt Anknüpfungspunkte für weitere Arbeiten auf.

2 Grundlagen der Automatisierungstechnik

Dieses Kapitel führt in die Grundlagen der Automatisierungstechnik ein, die zum Verständnis dieser Arbeit erforderlich sind. Dazu wird zunächst der Aufbau von Automatisierungssystemen dargestellt und die Rolle der Feldebene in diesem Aufbau beschrieben. Daraufhin geht das Kapitel auf die Eigenschaften der Feldebene ein. Abschließend werden die Systemelemente der Feldebene eingeführt und deren Charakteristika zusammengefasst.

2.1 Aufbau von Automatisierungssystemen

Die Aufgabe von Automatisierungssystemen ist es, technische Prozesse in die Lage zu versetzen, mehr oder weniger selbsttätig abzulaufen. Technische Prozesse sind Vorgänge, wie z. B. die Produktion von Werkstücken oder die Beheizung eines Hauses, bei denen physikalische Größen mit technischen Mitteln erfasst und beeinflusst werden. Da technische Prozesse immer innerhalb eines technischen Systems ablaufen, bestehen Automatisierungssysteme aus drei Teilen bzw. Teilsystemen [LaGö99a]:

- Einem technischen System, in dem ein technischer Prozess abläuft.
- Einem Rechner- und Kommunikationssystem, in welchem die Informationsverarbeitung abläuft. Das Rechner- und Kommunikationssystem dient der Informationsverarbeitung, d. h. der Erfassung, der Verarbeitung, der Darstellung und der Ausgabe von Informationen mit dem Ziel, den Ablauf eines technischen Prozesses zu steuern.
- Dem Prozessbedienpersonal, welches das Prozessgeschehen verfolgt, Vorgänge leitet und in Ausnahmesituationen tätig wird.

Abbildung 2.1 stellt den Aufbau von Automatisierungssystemen aus den drei Teilsystemen technisches System, Rechner- und Kommunikationssystem sowie dem Betriebspersonal dar.

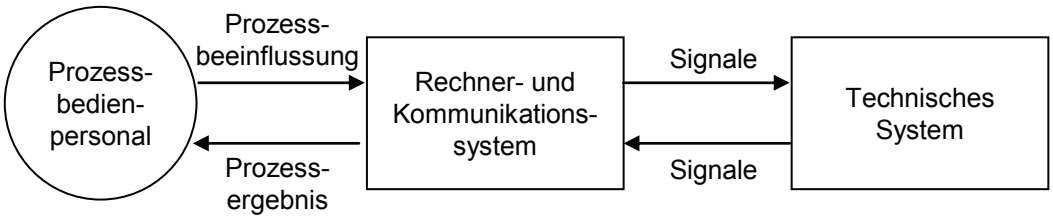

Abbildung 2.1: Aufbau von Automatisierungssystemen

Automatisierungssysteme lassen sich nach dem Umfang und der Komplexität des technischen Prozesses sowie der Art der Realisierung der Automatisierungsfunktionen in zwei Klassen von Automatisierungssystemen einteilen: einerseits in Systeme, bei denen der technische Prozess in

einem technischen Produkt abläuft (Produktautomatisierung), andererseits in Systeme, bei denen der technische Prozess in einer technischen Anlage abläuft (Anlagenautomatisierung).

Produktautomatisierungssysteme sind Geräte (z. B. Kaffeeautomaten) oder Maschinen (z. B. Aufzüge), bei denen es sich im Allgemeinen um Massenprodukte handelt, welche dedizierte Aufgaben erfüllen. Aus diesem Grund sind die Automatisierungsaufgaben in der Produktautomatisierung zumeist eng umgrenzt. Produktautomatisierungssysteme zeichnen sich üblicherweise durch das Vorkommen eines einzigen oder einiger weniger Automatisierungsrechner aus. Dabei werden als Automatisierungsrechner zumeist Mikrocontroller eingesetzt.

Anlagenautomatisierungssysteme werden zur Automatisierung industrieller Anlagen (z. B. Fertigungsanlagen, verfahrenstechnische Anlagen) eingesetzt, bei denen umfangreiche und komplexe Automatisierungsfunktionen auszuführen sind. Die Automatisierungsfunktionen können dabei zentral (d. h. auf einem einzigen Rechner) oder dezentral (d. h. auf mehrere Rechner verteilt) realisiert werden. Beide Ausführungen haben jeweils unterschiedliche Vor- und Nachteile in Bezug auf Kosten, Zuverlässigkeit, Änderbarkeit, Optimierbarkeit und Bedienbarkeit des Automatisierungssystems [LaGö99a].

Um die jeweiligen Vorteile einer Struktur zu nutzen und die jeweiligen Nachteile zu vermeiden, werden die Automatisierungsfunktionen bei komplexen Automatisierungssystemen in übereinanderliegende Hierarchieebenen gegliedert. Im Allgemeinen werden vier Hierarchieebenen unterschieden [LaGö99b]:

- Die langfristige Produktionsplanung, Kostenanalysen sowie Auftragsvergabe und Auftragsabwicklung werden auf der *Unternehmensleitebene* durchgeführt.
- Die Betriebsablaufplanung, Kapazitätsplanung, Auswertungen der Prozessergebnisse sowie Maßnahmen zur Qualitätssicherung werden auf der *Produktionsleitebene* ausgeführt.
- Die *Prozessleitebene* umfasst die Automatisierungsfunktionen zur Anlagenführung, also z. B. die Prozessüberwachung, die Prozessoptimierung und die Prozesssicherung sowie die Funktionen zur Maschinenführung, wie Steuern, Regeln, Grenzwertüberwachung sowie das Registrieren und die Anzeige von Prozessinformationen.
- Auf der *Feldebene* werden Prozessgrößen gemessen und Stelleingriffe in den technischen Prozess ausgeführt.

Um die Anwenderforderungen nach abgestufter Leistungsfähigkeit der einzelnen Ebenen, nach Flexibilität und Anpassbarkeit in hohem Maße zu erfüllen, nimmt die Anzahl der Automatisierungsrechner sowie deren Vernetzung von den oberen Ebenen zu den unteren Ebenen hin zu. Die hier betrachteten Hierarchieebenen werden in ihrer Gesamtheit auch als Automatisierungspyramide (AT-Pyramide) bezeichnet [Hauf06]. Abbildung 2.2 stellt die AT-Pyramide dar.

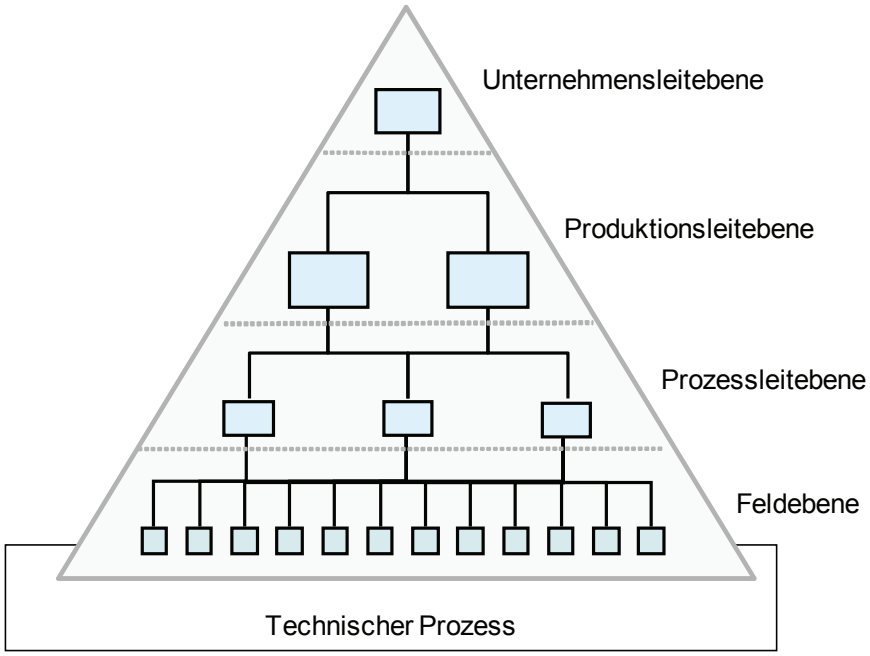

Abbildung 2.2: Automatisierungspyramide

Je nach Branche bestehen verfeinerte Ausprägungen der AT-Pyramide mit weiteren Ebenen und abweichenden Bezeichnungen. Ebenso existieren Automatisierungssysteme, bei denen die höheren Ebenen teilweise oder insgesamt wegfallen (z. B. Kraftfahrzeuge oder automatisierte Geräte). Aufgrund der erforderlichen Anbindung des Automatisierungssystems an den technischen Prozess bleibt die Feldebene jedoch stets in einem gewissen Mindestumfang bestehen [Eber05].

Auf den einzelnen Ebenen der Automatisierungspyramide werden unterschiedliche Technologien eingesetzt. Auf den Ebenen Unternehmensleitebene, Produktionsleitebene und Prozessleitebene von modernen Automatisierungssystemen werden zumeist keine Automatisierungstechnik-spezifischen Systemelemente eingesetzt. Stattdessen finden hauptsächlich Bausteine aus der Standard-IT Verwendung, z. B. Personal Computer, Ethernet-Netzwerke und Microsoft-Betriebssysteme [Enst02]. Um eine hohe Informationsdurchgängigkeit durch das gesamte Automatisierungssystem zu erreichen, existieren Bestrebungen, Standard-IT-Bausteine auch auf der Feldebene einzusetzen [Schw06]. Aufgrund der Nähe der Feldebene zum technischen Prozess unterscheiden sich die Eigenschaften der Feldebene jedoch stark von denen der übrigen Ebenen. Daher ist ein Einsatz von Standard-IT-Technologien auf der Feldebene in vielen Fällen nicht möglich, weshalb auf der Feldebene nach wie vor zahlreiche spezialisierte Technologien Verwendung finden. Die folgenden Abschnitte beschreiben die Eigenschaften und die Systemelemente der Feldebene näher.

2.2 Eigenschaften der Feldebene von Automatisierungssystemen

Technische Systeme sind aufgrund ihrer Aufgabe, Struktur, Position und Größe häufig leicht zugänglich. Beispielsweise befinden sich Kraftfahrzeuge meist auf öffentlichen Straßen. Bei Automatisierungsanlagen kann die räumliche Ausdehnung des technischen Systems bis zu mehreren Kilometern betragen [HiLi97]. Daher existiert oftmals eine große Anzahl von Personen, die sich Zugang zu dem technischen System der Anlage verschaffen können. Da die Systemelemente der Feldebene in das technische System eingebettet sind, sind sie, ebenso wie das technische System selbst, exponiert.

Die Abläufe in technischen Prozessen unterliegen stets zeitlichen Bedingungen. Sie dürfen nicht zu früh beginnen oder zu spät enden. Außerdem können in technischen Prozessen mehrere Vorgänge zum selben Zeitpunkt stattfinden. Daher ist es nicht ausreichend, wenn die Systemelemente der Feldebene nur die *Richtigkeit* von Ergebnissen gewährleisten können, es ist zusätzlich erforderlich, dass die Messung von Prozessgrößen und die Ausgabe von Stellsignalen zum jeweils korrekten Zeitpunkt erfolgen. Zudem muss es möglich sein, auf gleichzeitig eintretende Ereignisse zu reagieren bzw. gleichzeitige Aktionen auszulösen. Die Eignung, diese Anforderungen zu erfüllen, wird als Echtzeitfähigkeit bezeichnet. Bei technischen Prozessen treten häufig so genannte harte Echtzeitanforderungen [Cool03] auf, d. h., die Nichteinhaltung der Zeitschranken ist mit einem Versagen des Automatisierungssystems gleichzusetzen.

Die Randbedingungen, denen die Systemelemente der Feldebene unterworfen sind, können sich von Fall zu Fall beträchtlich unterscheiden. Beispielsweise können die Zuverlässigkeit von Datenübertragungen, die Widerstandsfähigkeit gegen Störeinstrahlungen oder die Einhaltung kurzer Zeitschranken je nach Anwendungsfall mehr oder weniger Relevanz aufweisen. Zudem können sich diese Anforderungen gegenseitig beeinflussen. Daher existiert auf der Feldebene auch nicht die *eine* Technologie, sondern eine heterogene Menge unterschiedlicher Technologien, welche für den Einsatz unter bestimmten Randbedingungen zugeschnitten sind [Eber05].

Die traditionellen Aufgaben der Feldebene bestanden ausschließlich in dem Messen von Prozessgrößen und dem Eingriff in den technischen Prozess. Dazu wurden die Systemelemente der Feldebene in den technischen Prozess integriert und somit örtlich verteilt (örtlich dezentrale Realisierung). Die Funktionalität zur Verarbeitung der Prozessgrößen wurde jedoch zentral auf einem Rechner auf der Prozessleitebene ausgeführt (funktional zentrale Realisierung). Zur Verbindung der Systemelemente mit dem zentralen Rechner wurden üblicherweise Zweidrahtleitungen verwendet, welche sternförmig zwischen dem zentralen Rechner und den einzelnen Systemelementen der Feldebene verlegt wurden.

Aufgrund steigenden Kostendrucks und zunehmender Komplexität ist die Struktur der Feldebene bei modernen Automatisierungssystemen in zwei Aspekten verändert [WGU03]. Bei

einer zunehmenden Anzahl an Teilprozessen und Funktionen ist es vorteilhaft, wenn die Automatisierung nicht nur örtlich, sondern auch funktional dezentral realisiert wird [LaGö99a]. Bei modernen Automatisierungssystemen sind die Funktionalitäten zur Verarbeitung der Prozessgrößen daher nicht mehr in einem zentralen Rechner auf der Prozessleitebene angesiedelt. Stattdessen sind diese Funktionalitäten in die Feldebene eingesickert [Serv05]. Die Verarbeitung der Prozessgrößen wird auf einer Vielzahl von Rechnern ausgeführt, die in den Teil des technischen Prozesses integriert sind, der von dem jeweiligen Rechner geführt wird (örtlich und funktional dezentrale Realisierung). Bei derartigen Automatisierungssystemen handelt es sich oftmals um *verteilte Automatisierungssysteme*, bei denen die Automatisierungsrechner ohne die Einschränkungen miteinander kommunizieren dürfen, die bei einer hierarchischen Strukturierung vorgegeben sind.

Die zweite Veränderung betrifft die Verbindung der Systemelemente. Die Systemelemente der Feldebene müssen Informationen austauschen, um die geforderten Funktionalitäten zu erbringen. Die Vernetzung einer großen Anzahl von Systemelementen erfordert jedoch einen hohen planerischen Aufwand und großen Materialeinsatz. Zudem sind Änderungen (z. B. Erweiterungen) schwierig und aufwändig. Bei modernen Automatisierungssystemen werden daher sogenannte Feldbusse eingesetzt, die es erlauben, mehrere Kommunikationsteilnehmer an ein einziges Busmedium (z. B. eine Zweidrahtleitung) anzubinden. Feldbusse werden dabei sowohl auf der Feldebene von hierarchischen Automatisierungssystemen als auch bei verteilten Automatisierungssystemen verwendet. Auch wenn die Kommunikationsbeziehungen in den beiden Fällen unterschiedliche Aufgaben erfüllen, wird die physische Vernetzung der Rechner mit Feldbussen gleichartig ausgeführt.

2.3 Systemelemente der Feldebene von Automatisierungssystemen

Die Funktionalitäten der Feldebene werden durch zwei unterschiedliche Arten von Systemelementen realisiert: solche, die Prozessinformationen verarbeiten, sowie solche, welche die Kommunikation bewerkstelligen. Die Systemelemente, welche Prozessinformationen verarbeiten, werden auf der Feldebene als *Feldgeräte*, die Systemelemente zur Kommunikation als *Feldbusse* bezeichnet.

2.3.1 Feldgeräte

Feldgeräte werden zur Erfassung von Prozesssignalen sowie zur Generierung von Stellsignalen eingesetzt. Typische Tätigkeiten, die im Rahmen dieser Aufgaben auftreten und die von Feldgeräten durchgeführt werden, sind die Verarbeitung der erfassten Messwerte, die Verteilung von Daten und zunehmend auch das Steuern und Regeln von Teilprozessen [Kast08]. Zur Erfüllung dieser Aufgaben sind Feldgeräte prinzipiell aus den folgenden Bestandteilen aufgebaut

[LiFä00], die je nach Einsatzzweck des Feldgeräts unterschiedlich vorhanden bzw. ausgeprägt sein können:

- Sensoren und Aktoren zur Interaktion mit dem technischen Prozess
- Rechner zur Ausführung der Funktionalitäten
- Feldbusschnittstelle zur Anbindung an Feldbussysteme
- Programmierschnittstelle zum Aufspielen der Feldgerätesoftware

Abbildung 2.3 stellt den schematischen Hardware-Aufbau eines Feldgeräts dar.

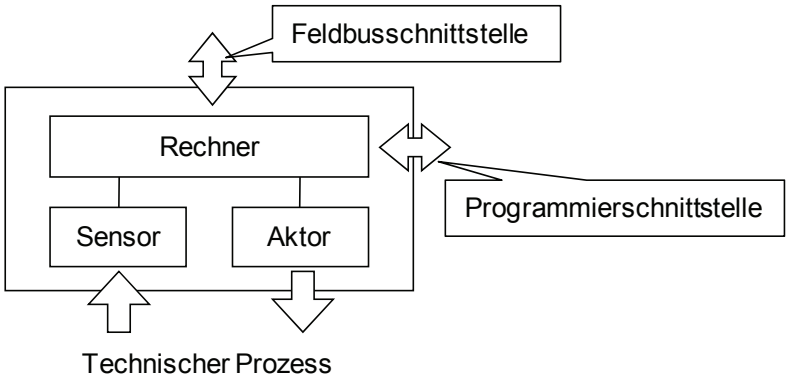

Abbildung 2.3: Schematischer Hardware-Aufbau von Feldgeräten

Anhand des eingesetzten Rechnertyps lassen sich die folgenden Arten von Feldgeräten unterscheiden:

- *Festverdrahtete Steuerungen* sind Feldgeräte, bei denen die gesamte Funktionalität unveränderlich in Hardware realisiert ist. Sie kommen hauptsächlich in Umgebungen zum Einsatz, bei denen ein hohes Maß an technischer Sicherheit gefordert wird [Seid04] [IEC61508-7]. Festverdrahtete Steuerungen werden heutzutage zumeist in Form von Field-Programmable-Gate-Arrays (FPGAs) oder Application-Specific-Integrated-Circuits (ASICs) realisiert [Meye07].
- *Industrie-PCs*, d. h. PCs, welche an den Einsatz in Umgebungen mit hoher Belastung durch Flüssigkeiten, Staub, Temperatur und elektromagnetischer Störstrahlung angepasst sind. Industrie-PCs verfügen zudem über entsprechende Schnittstellenkarten, welche die Anbindung an Feldbussysteme ermöglichen. Industrie-PCs werden aufgrund ihrer Größe und ihres hohen Preises zumeist dann eingesetzt, wenn eine hohe Rechenleistung erforderlich ist. Es existieren zahlreiche Industrie-PC-Typen, welche einen Touchscreen als Bedienelement enthalten. Diese werden oftmals dann eingesetzt, wenn eine einfache Bedienbarkeit von komplexen Funktionalitäten gefordert ist.

- *Mikrocontroller* (µC) enthalten Prozessor, Speicher sowie Ein-/Ausgabe-Kanäle in einem einzigen Baustein. Mikrocontroller verbinden eine hohe Flexibilität mit hoher Robustheit, geringen Ausmaßen und geringem Preis. Da auf der Feldebene von Automatisierungsanlagen üblicherweise zahlreiche Feldgeräte erforderlich sind, werden dort üblicherweise Mikrocontroller-basierte Feldgeräte herangezogen. Andere Rechnertypen kommen zum Einsatz, wenn die oben angeführten Forderungen (sehr hohes Maß an technischer Sicherheit, sehr hohe Rechenleistung und einfache Bedienbarkeit) gegeben sind. Mikrocontroller können als fertige Produkte erworben werden oder aus vorgefertigten Beschreibungen von Funktionsblöcken (IP-Cores) in Form von ASICs oder FPGAs realisiert werden [Meye07].

- *Speicherprogrammierbare Steuerungen* (SPS) sind Automatisierungsrechner, die sich durch ihren zyklischen Betriebsmodus sowie eine einfache Programmierbarkeit mit spezialisierten SPS-Programmiersprachen [EN61131-3] auszeichnen. Während „klassische" SPS-Systeme mit spezieller SPS-Hardware realisiert wurden [Frie94], bauen moderne SPS-Systeme auf Mikrocontrollern oder Industrie-PCs auf [Seit03]. Dabei wird der SPS-spezifische Betriebsmodus in Form von Software realisiert (SPS-Laufzeitumgebung), auf der die SPS-Programme aufsetzen.

Aufgrund der oben beschriebenen Eigenschaften, insbesondere der geringen Anschaffungskosten, stellen Mikrocontroller die häufigste Realisierungsform von Feldgeräterechnern dar [Dorr02] [Wagn07]. Die Funktionalität von Mikrocontroller-basierten Feldgeräten (einschließlich SPS und programmierbaren FPGAs/ASICs) wird durch Software realisiert. Aufgrund der zahlreichen Einsatzszenarios existiert eine große Anzahl unterschiedlicher Mikrocontrollertypen mit verschiedenen Architekturen auf dem Markt. Dabei wird eine große Bandbreite von Leistungsfähigkeiten abgedeckt. Die Leistungsfähigkeit eines Mikrocontrollers kann in erster Näherung durch seine Wortbreite angegeben werden.

Übliche Wortbreiten bei Mikrocontrollern sind 4 bit, 8 bit, 16 bit und 32 bit. Tabelle 2.1 listet typische Vertreter für Mikrocontroller und ihre Ressourcen auf [R4283G06] [RH8G03] [RM16C06] [RM32R07]. Daraus wird ersichtlich, dass die Ressourcen von Mikrocontrollern deutlich kleiner bemessen sind als die Ressourcen von Standard-IT-Rechnern, welche heutzutage üblicherweise über eine Wortbreite zwischen 32 bit und 64 bit, 300 GB nicht-flüchtigen Speicher, mehr als 1 GB flüchtigen Speicher und eine Taktfrequenz in der Größenordnung von 3 GHz verfügen [Dell07].

Tabelle 2.1: Größenordnungen der Ressourcen typischer Mikrocontroller

µC-Familie	Wortbreite	Speicher (ROM)	Speicher (RAM)	Maximale Taktfrequenz
Renesas 4283	4 bit	2 kB	32 Byte	4 MHz
Renesas H8/330	8 bit	32 kB	512 Byte	10 MHz
Renesas M16C	16 bit	384 kB	31 kB	24 MHz
Renesas M32R	32 bit	1 MB	176 kB	160 MHz

Während (Industrie-)PCs bereits seit vielen Jahren Festplatten als nichtflüchtiges, mehrfach beschreibbares Speichermedium verwenden, setzen erst moderne Mikrocontroller nichtflüchtigen Speicher ein, der mehrmals beschrieben werden kann (EEPROM/Flash-ROM), und somit Software-Updates ermöglicht. Man unterscheidet bei der Software von Feldgeräten zwischen Anwenderprogrammen (Automatisierungsprogrammen) und Systemprogrammen [LaGö99a].

Anwenderprogramme sind Applikationen, welche die Automatisierungsfunktionen ausführen, also Programme zur Prozesssteuerung, -überwachung, -regelung usw. Systemprogramme sind dagegen Programme, welche die Anwenderprogramme steuern und verwalten. Zu den Systemprogrammen zählen Laufzeitumgebungen und Betriebssysteme, wobei bei Feldgeräten insbesondere Echtzeit- bzw. Embedded-Betriebssysteme zum Einsatz kommen. Da die Funktionalität eines Feldgeräts auf dessen spezifische Aufgabe zugeschnitten ist, wird Feldgerätesoftware im Betrieb üblicherweise nur im Wartungsfall oder bei Modifikationen an dem Automatisierungssystem verändert.

Die funktionale Verteiltheit der Automatisierungsfunktionen auf der Feldebene erfordert die Kooperation mehrerer Feldgeräte. Daher sind Feldgeräte mit einer Schnittstelle (vgl. Abbildung 2.3) zur Anbindung an Feldbusse ausgestattet. Diese werden im nächsten Abschnitt vorgestellt.

2.3.2 Feldbusse

Feldbussysteme sind serielle Kommunikationssysteme, die zum Datenaustausch auf der Feldebene von modernen Automatisierungssystemen verwendet werden [SHW99] [Reiß02]. Feldbusse erfüllen dabei unterschiedliche Kommunikationsaufgaben. Einerseits werden mit Feldbussen Messwerte bzw. Stellgrößen zwischen Automatisierungsrechnern und Sensoren bzw. Aktoren übermittelt, welche direkt[1] oder mithilfe von Anschlussmodulen an einen Feldbus angebunden sind. Andererseits dienen Feldbusse zum Informationsaustausch zwischen den Automatisierungsrechnern der Feldebene untereinander sowie zwischen den Automatisierungsrechnern der Feldebene und den Rechnern übergeordneter Hierarchieebenen.

[1] Solche Sensoren/Aktoren werden als „intelligente“ Sensoren/Aktoren bezeichnet (*Smart Sensor/Actuator*).

Im Gegensatz zur konventionellen Kommunikationstechnik auf der Feldebene können bei Feldbussen mehrere Feldgeräte über ein gemeinsames Kommunikationsmedium verbunden werden. Dazu sind Protokolle erforderlich, welche die Codierung der Signale, den Zugriff auf das gemeinsam genutzte Medium und die Flusssteuerung festlegen. Solche Feldbuskommunikationsprotokolle lassen sich anhand des ISO/OSI-Referenzmodells [ISO7498-1] beschreiben, wobei die Informationen, welche zur Übermittlung von Nachrichten erforderlich sind, den Ebenen 1 (Bitübertragung) und 2 (Sicherung) zugeordnet werden. Die Ebene 2 ist wiederum in zwei Teilebenen unterteilt. Die Ebene 2a (Media Access Control, MAC) beschreibt die Art des Zugriffs (Arbitrierung) der Feldgeräte auf den Feldbus, die Ebene 2b (Logical Link Control, LLC) beschreibt die Datenflusssteuerung. Die übertragenen Nutzdaten sind der Ebene 7 (Anwendung) zugeordnet. Die verbleibenden Ebenen 3–6 des ISO/OSI-Referenzmodells befassen sich u. a. mit Routing-, Segmentierungs- und Sitzungsfunktionen. Diese Ebenen werden von Feldbusprotokollen nicht implementiert [Reiß02]. Abbildung 2.4 stellt dar, wie die Ebenen des ISO/OSI-Referenzmodells auf die Struktur von Feldbusnachrichten abgebildet werden.

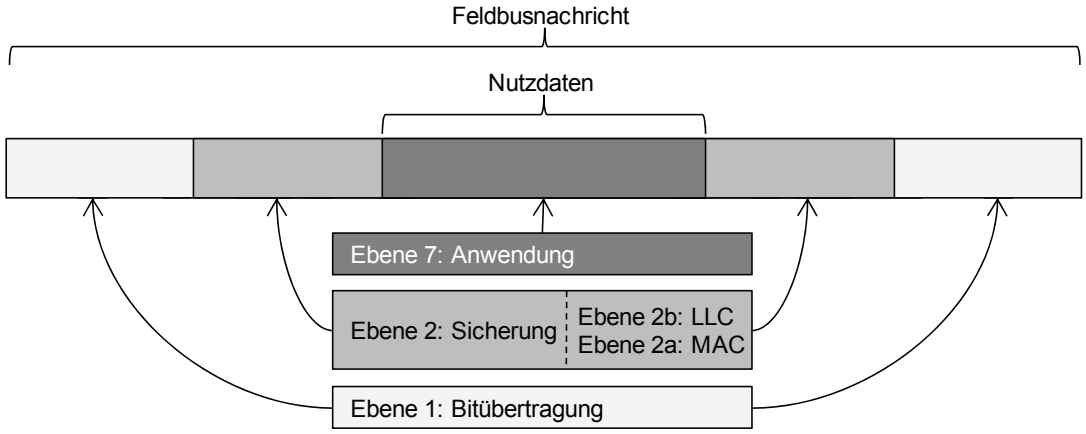

Abbildung 2.4: Struktur von Feldbusnachrichten

Je nach Einsatzgebiet existieren unterschiedliche Anforderungen an Feldbusse. Bei räumlich ausgedehnten Automatisierungssystemen ist es erforderlich, große Entfernungen mit Feldbuskabeln überbrücken zu können. In der chemischen und verfahrenstechnischen Industrie und der Gebäudeautomation können solche Entfernungen bis zu mehreren Kilometern betragen [Reiß02] [Buss96]. Die einzuhaltenden Zeitschranken sind ebenfalls unterschiedlich. Daher verfügen Feldbussysteme je nach Einsatzgebiet über unterschiedliche Latenzzeiten (Laufzeiten auf Übertragungsmedium und Verarbeitungszeiten durch Protokollautomaten). Auch die Bandbreite (Rate der übertragbaren Daten pro Zeiteinheit) von Feldbussystemen ist je nach zu übertragender Datenmenge unterschiedlich [HiLi97].

In nahezu allen Einsatzbereichen von Feldbussen ist eine hohe Widerstandsfähigkeit gegen Störungen erforderlich, da Feldbusse einerseits häufig in Bereichen mit starker elektromagnetischer Störstrahlung eingesetzt werden und andererseits, da fehlerhafte Datenübertragungen Übergänge des technischen Prozesses in Fehlerzustände verursachen können. Derartige Fehlerzustände

können in vielen Domänen – beispielsweise der chemischen Industrie oder bei Kraftfahrzeugen – erhebliche Schäden am technischen Prozess selbst sowie seiner Umgebung verursachen.

Aufgrund der unterschiedlichen, zum Teil widersprüchlichen Anforderungen an Feldbussysteme existiert heute nicht *ein* Feldbusstandard, sondern zahlreiche verschiedene. Dabei werden in den unterschiedlichen Domänen der Automatisierungstechnik bestimmte Feldbussysteme vorherrschend eingesetzt.

Für die chemische und verfahrenstechnische Industrie sind hier *PROFIBUS PA* [IEC61158] [EN61784], *Foundation Fieldbus* [IEC61158] und der *HART*-Feldbus [HART07] anzuführen, für die fertigungstechnische Industrie *PROFIBUS DB* [IEC61158] und *Interbus-S* [IEC61158] [EN50254].

In der Gebäudeautomatisierung werden vorwiegend die Feldbusse *LON* [EN14908-1], *BACnet* [EN16484-5] und *EIB/KNX* [EN50090-4] [IEC14543-3] eingesetzt.

Zur Vernetzung von Kfz-Elektronik wird oftmals *CAN* [ISO11898] verwendet, in jüngerer Zeit werden auch die Feldbussysteme *FlexRay* [Flex05a] [Flex06] für zeitlich deterministische Applikationen, *LIN* [LIN06] für einfachste Funktionalitäten und *MOST* [MOST06] für Multimedia-Anwendungen herangezogen.

Ein sich abzeichnender Trend besteht in der Verwendung von Ethernet als Basis für neue Feldbusstandards bzw. in der Adaption bestehender Feldbusstandards an Ethernet. Beispiele hierfür sind *PROFINET IO* und *EtherCAT* [Schw06]. Auch drahtlose Kommunikationstechnologien aus der Standard-IT und der Unterhaltungselektronikindustrie sollen ihren Weg in die Feldebene finden, z. B. *WLAN*, *Bluetooth*, *ZigBee* und *WiMax* [Hasc06] [Meie06].

2.4 Zusammenfassung der charakteristischen Eigenschaften der Feldebene von Automatisierungssystemen

Tabelle 2.2 fasst die Eigenschaften zusammen, welche für die Feldebene von modernen Automatisierungssystemen charakteristisch sind.

Tabelle 2.2: Eigenschaften der Feldebene von modernen Automatisierungssystemen

Eigenschaft	**Ausprägung bei modernen Automatisierungssystemen**
Verteiltheit	Oftmals hohe örtliche und funktionale Verteilung aufgrund der Integration der Systemelemente in den technischen Prozess.
Exponiertheit	Oftmals hohe Exponiertheit aufgrund der räumlichen Lage des Automatisierungssystems.
Echtzeitfähigkeit	Oftmals harte Echtzeitfähigkeit durch den technischen Prozess gefordert, jedoch unterschiedlich große Zeitgrenzen aufgrund der Art der technischen Prozesse.
Vorherrschende Technologie	Unterschiedliche Technologien aufgrund der unterschiedlichen Randbedingungen.
Ressourcenkapazität	Unterschiedliche Ressourcenkapazitäten, aus Kostengründen zumeist sehr gering im Vergleich zur Standard-IT.
Veränderlichkeit der Software	Sehr geringe Veränderlichkeit der Software, da nur im Wartungsfall bzw. bei Modifikationen des Automatisierungssystems Veränderungen vorgenommen werden.

Dieses Kapitel befasste sich mit den automatisierungstechnischen Grundlagen für die vorliegende Arbeit. Automatisierungssysteme verfügen stets über Elemente, welche die Anbindung an ein technisches System realisieren. Diese Elemente werden der *Feldebene* von Automatisierungssystemen zugeordnet. Die Aufgaben der Systemelemente der Feldebene (Feldgeräte und Feldbusse) bestehen in der Verarbeitung von erfassten Messwerten, in der Verteilung von Daten und zunehmend auch in der Steuerung und der Regelung von Teilprozessen. Aufgrund der Integration in den technischen Prozess sind die Systemelemente der Feldebene oftmals örtlich weiträumig verteilt und entsprechend exponiert. Um den unterschiedlichen Aufgaben der Feldebene und den unterschiedlichen Anforderungen technischer Prozesse zu genügen, existieren zahlreiche Arten von Feldgeräten und Feldbussen. Verglichen mit Standard-IT-Technologien weisen Feldgeräte und Feldbusse zumeist eine sehr geringe Ressourcenkapazität auf. Zum Verständnis der Problemstellung, der sich diese Arbeit widmet, sind zudem Grundkenntnisse über das Fachgebiet der IT-Sicherheit erforderlich. Diese werden im folgenden Kapitel vermittelt.

3 Grundlagen der IT-Sicherheit

Während sich das vorangegangene Kapitel mit den automatisierungstechnischen Grundlagen befasste, werden in diesem Kapitel die Grundlagen der IT-Sicherheit betrachtet. Zum Verständnis dieser Thematik wird zunächst die IT-Sicherheit von der technischen Sicherheit abgegrenzt und es werden Begriffe aus dem Fachgebiet der IT-Sicherheit eingeführt. Um die unterschiedlichen Arten von Bedrohungen der IT-Sicherheit und die verschiedenen Verursacher dieser Bedrohungen nachvollziehen zu können, werden sie in diesem Kapitel diskutiert. Da im Rahmen der vorliegenden Arbeit ein Konzept zum Schutz der IT-Sicherheit erarbeitet werden soll, wird zudem dargestellt, wie der Schutz der IT-Sicherheit prinzipiell erbracht werden kann und es werden die Tätigkeiten beschrieben, welche zur Realisierung von Schutz grundsätzlich durchzuführen sind.

3.1 Definition und Abgrenzung von Begriffen

3.1.1 Sicherheit, Safety, Security

Der Begriff *Sicherheit* bezeichnet den Zustand eines Systems, in dem das Risiko für das Auftreten eines Schadens geringer ist als das größte noch vertretbare Risiko (Grenzrisiko) [Kona05]. Das Risiko beschreibt dabei die Wahrscheinlichkeit für das Auftreten eines Schadensfalls multipliziert mit dem Ausmaß des Schadens [Gott02] [IEC61508-4].

In der deutschen Sprache umfasst *Sicherheit* dabei zwei unterschiedliche Bedeutungen, die in der englischen Sprache mit den Bezeichnungen *Safety* und *Security* belegt sind. Sicherheit im Sinne von *Safety* bezeichnet im Kontext der Automatisierungstechnik die Freiheit von Gefahren, die von Automatisierungssystemen ausgehen können und welche Menschen oder Umwelt bedrohen. Der Begriff *Security* bezieht sich dagegen auf Ereignisse, welche von Menschen oder der Umwelt hervorgerufen werden können und den Betrieb von Automatisierungssystemen gefährden [LaGö99a]. Sicherheit im Sinne von *Security* wird häufig auch als IT-Sicherheit (Informationstechnik-Sicherheit) bezeichnet [Beth90].

3.1.2 Bedrohung, Schwachstelle, Gefährdung

Eine *Bedrohung* ist ein mögliches Ereignis, durch das prinzipiell ein Schaden entstehen kann [GGK+04]. Eine Systemeigenschaft, durch welche eine Bedrohung wirksam wird, d. h. einen Schadensfall tatsächlich hervorrufen kann, wird als *Schwachstelle* des Systems bezeichnet. Ein System ist sicher im Sinne von *Security*, wenn keine Schwachstellen existieren, da Bedrohungen in diesem Fall nicht wirksam werden können, oder wenn keine Bedrohungen auftreten können.

Ein Ereignis, das aufgrund einer existierenden Schwachstelle einen Schaden hervorrufen kann, wird als *Gefährdung* bezeichnet [GGK+04]. Abbildung 3.1 stellt den Zusammenhang zwischen Bedrohung, Schwachstelle und Gefährdung für die Sicherheit im Sinne von Security dar.

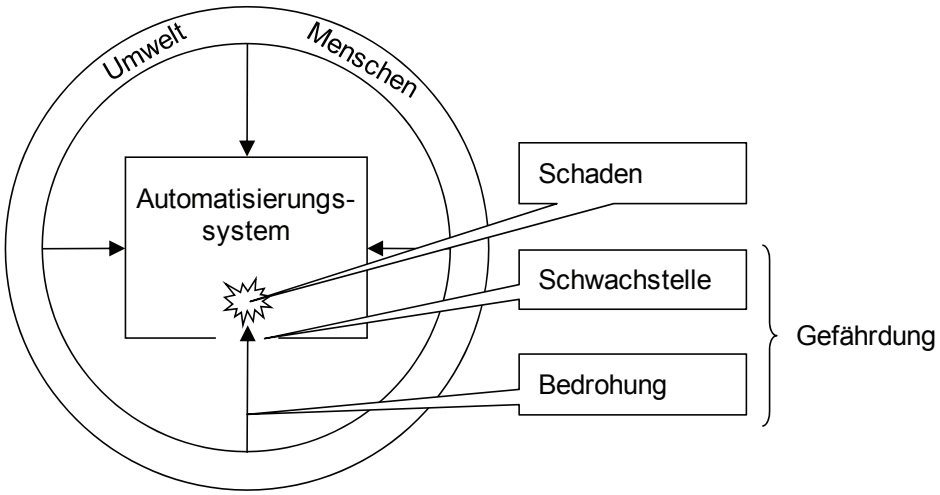

Abbildung 3.1: Zusammenhang zwischen Bedrohung, Schwachstelle und Gefährdung

3.2 Bedrohungen der IT-Sicherheit

3.2.1 Grundbedrohungen der IT-Sicherheit

Es existieren zahlreiche Ursachen für die Bedrohung der IT-Sicherheit, z. B. höhere Gewalt, Fahrlässigkeit, technisches Versagen, organisatorische Mängel und Vorsatz [EsAt06]. Die aus diesen Ursachen hervorgehenden Bedrohungen lassen sich bezüglich ihrer Wirkungsweise in zwei Klassen einteilen: Bedrohungen aufgrund von Manipulationen der Hardware und Bedrohungen durch die Manipulation von Software bzw. Daten [Ecke03].

Auswirkungen aufgrund von Manipulationen an IT-Systemen treten bei Automatisierungssystemen grundsätzlich dann auf, wenn die zur Führung des Prozesses erforderlichen Informationen nicht oder nicht korrekt erfasst, verarbeitet, transportiert oder ausgegeben werden können. Hat eine Manipulation von informationsverarbeitender Hardware eines Automatisierungssystems keine Auswirkungen auf die Informationsverarbeitung selbst, so treten auch keine Auswirkungen auf das Automatisierungssystem auf.

Da im Rahmen dieser Arbeit die Absicherung der IT-Sicherheit keinem Selbstzweck dient, sondern dem Schutz eines Automatisierungssystems, werden im Folgenden nur Manipulationen an Informationen betrachtet, unabhängig davon, ob diese durch Manipulationen an Hardware oder Software hervorgerufen werden. Bei Informationsmanipulationen lassen sich drei grundsätzliche Bedrohungen unterscheiden [Dier04]:

- Unautorisierte Informationsgewinnung
- Unautorisierte Manipulation von Kommunikation (einschließlich unautorisierte Informationsproduktion)
- Unautorisierte Manipulation von Funktionalität

3.2.2 Intentionale und nichtintentionale Bedrohungen der IT-Sicherheit

Bei Bedrohungen lässt sich unterscheiden, ob diese intentional (absichtlich, vorsätzlich, gezielt) oder nichtintentional (zufällig, fahrlässig, unvermeidbar) herbeigeführt werden [Dier04]. Die intentionale Durchführung einer Handlung, welche die IT-Sicherheit bedroht, wird auch als *Angriff* auf die IT-Sicherheit bezeichnet. Der wesentliche Unterschied zwischen Angriffen und nichtintentional auftretenden Ereignissen liegt darin, dass bei einem Angriff sowohl mit der gezielten Ausnützung von Schwachstellen des angegriffenen Systems zu rechnen ist als auch mit der Umgehung oder der Manipulation von vorhandenem Schutz, der eine Schwachstelle absichern soll. Das folgende Beispiel soll dies verdeutlichen.

Zum Schutz vor nichtintentionalen Modifikationen von Nachrichten bei der Übertragung über einen Kommunikationskanal, beispielsweise durch fehlerhafte Übertragung aufgrund elektromagnetischer Störungen, werden heutzutage u. a. Prüfsummen eingesetzt [BMB+05]. Dabei ist die Wahrscheinlichkeit, dass eine Störung eine Nachricht so ändert, dass diese Veränderung anhand der Prüfsumme nicht erkannt werden kann, gering. Bei einem Angriff mit dem Ziel, eine übertragene Nachricht zu verfälschen, kann ein Angreifer den Nachrichteninhalt jedoch beispielsweise so manipulieren, dass die Prüfsumme auch für die veränderte Nachricht gültig ist. Dadurch ist es dem Empfänger nicht möglich, die Manipulation zu erkennen.

Intentionale und nichtintentionale Bedrohungen unterscheiden sich also nicht in ihrer prinzipiellen Wirkungsweise (z. B. der Modifikation von Informationen). Intentionale Bedrohungen stellen jedoch höhere Anforderungen an den Schutz der IT-Sicherheit. Diese Anforderungen schließen dabei den Schutz vor nichtintentionalen Bedrohungen stets mit ein. Aus diesem Grund werden im Folgenden ausschließlich intentionale, die IT-Sicherheit bedrohende Handlungen – Angriffe – betrachtet.

3.2.3 Arten und Motivationen von Angreifern

Angriffe auf die IT-Sicherheit werden per Definition absichtlich und zielgerichtet von Menschen ausgeführt. Um angemessen auf solche Angriffe reagieren zu können, erläutert dieser Abschnitt die unterschiedlichen Arten von Angreifern und deren Motivationen. In der Literatur werden üblicherweise drei Angreifertypen unterschieden [Ecke03] [Rent04] [SMR00].

Skript-Kids (*Skript-Kiddies*[2]) sind Angreifer, oftmals Jugendliche, welche nicht über vertiefte Kenntnisse über IT-Sicherheit und die Funktionsweise der Systeme verfügen, welche sie bedrohen. Sie verwenden daher für ihre Angriffe vorgefertigte (Software-)Werkzeuge. Die Motivation von Skript-Kids umfasst sowohl Neugier und Spieltrieb als auch Vandalismus sowie finanzielle Motivationen.

Die Begriffe *Hacker* und *Cracker* bezeichnen technisch versierte Angreifer mit guten Kenntnissen über die Funktionsweise von IT-Systemen und über IT-Sicherheit. Dies ermöglicht es diesen Angreifertypen, Schwachstellen von IT-Systemen zu finden und auszunützen. Cracker unterscheiden sich insofern von Hackern, dass Cracker ihre Kenntnisse dazu verwenden, IT-Systeme tatsächlich zu manipulieren. Das Ziel von Hackern ist dagegen primär die Identifikation von Schachstellen mit dem Ziel der Absicherung von IT-Systemen. Die Motivation von Hackern und Crackern besteht einerseits aus Neugierde, sie wird bei Crackern aber zunehmend auch durch finanzielle Anreize verursacht [BMB+05], beispielsweise durch Spionage, Erpressung oder Diebstahl.

Nachrichtendienste befassen sich professionell mit der Beschaffung von Informationen und anderen Aktivitäten, wie z. B. Sabotage. Die Motivation für diese Tätigkeiten ergibt sich aus einer staatlichen oder privatwirtschaftlichen Beauftragung. Nachrichtendienste sind üblicherweise sehr gut für ihre Aufgaben gerüstet, da sie zahlreiche, gegebenenfalls hoch qualifizierte Experten für IT-Systemtechnik und IT-Sicherheit beschäftigen können und teilweise über sehr gute technische und organisatorische Infrastrukturen verfügen.

Aufgrund ihrer jeweiligen Eigenschaften weisen die verschiedenen Angreifertypen eine unterschiedliche Gefährlichkeit auf [Schn01]. Während Skript-Kids bereits vor der Manipulation leicht gesicherter IT-Systeme kapitulieren müssen, ist zur Abwehr von Angriffen durch Hacker/Cracker aufgrund ihrer Kenntnisse ein stärkerer Schutz erforderlich. Zur Absicherung gegen Nachrichtendienste empfiehlt sich der stärkste verfügbare Schutz. Tabelle 3.1 fasst die verschiedenen Angreifertypen sowie die Stärken des Schutzes zusammen, die zur Abwehr der Angreifertypen erforderlich sind.

Tabelle 3.1: Angreifertypen und erforderliche Schutzstärken

Angreifertyp	**Erforderliche Schutzstärke**
Skript-Kid	Gering
Hacker/Cracker	Mittel
Nachrichtendienst	Hoch

[2] Der Begriff *Kid* bzw. *Kiddie* bezieht sich auf das jugendliche Alter und die geringe Kompetenz dieses Angreifertyps.

3.3 Schutz der IT-Sicherheit

In den vorangegangenen Abschnitten wurde beschrieben, auf welche Weise die IT-Sicherheit kompromittiert werden kann und welche Arten von Angreifern existieren, die motiviert sind, Angriffe auf die IT-Sicherheit durchzuführen. Um das Auftreten von Schäden durch derartige Manipulationen von informationsverarbeitenden Systemen zu unterbinden, müssen diese entsprechend geschützt werden.

3.3.1 Schutzfunktionalitäten und Schutzmechanismen

Um IT-Systeme vor Gefährdungen durch Angriffe zu schützen, gibt es zwei Ansatzpunkte: die Beseitigung der Bedrohungen der IT-Sicherheit und die Beseitigung der Schwachstellen des zu schützenden Systems. Da die Bedrohungen (d. h. die möglichen Aktivitäten von Angreifern) im Allgemeinen nicht dem Einfluss der Betreiber von IT-Systemen unterliegen, wird zumeist versucht, an den Schwachstellen von IT-Systemen anzusetzen. Ausgehend von den abzusichernden Schwachstellen des Systems werden Schutzfunktionalitäten definiert, welche den Schutz des Systems bezüglich einer oder mehrerer Schwachstellen gewährleisten sollen [Zitt07] [GaNa00]. Der Begriff Schutzfunktionalität beschreibt dabei das Ziel (die Absicherung einer Schwachstelle), nicht aber, auf welche Weise dieses Ziel zu erreichen ist.

Schutzfunktionalitäten werden durch einen oder mehrere Schutzmechanismen [Roed02] [UrhG07] erbracht. Ein Schutzmechanismus ist ein Ablauf, der einen festgelegten Vorgang bzw. mehrere festgelegte, ineinandergreifende Vorgänge beschreibt mit dem Ziel, Schutz für ein System zu realisieren, beispielsweise ein kryptografischer Algorithmus [Sail99]. Da eine große Anzahl unterschiedlicher Schutzmechanismen existiert, denen gleichartige Funktionsweisen zugrunde liegen, lassen sich Schutzmechanismen zu Klassen zusammenfassen [Eder05]. Zum Einsatz in einem IT-System müssen Schutzmechanismen realisiert werden, wobei die festgelegten Vorgänge in Software implementiert oder in Hardware synthetisiert werden. Abbildung 3.2 stellt den Zusammenhang zwischen Schutzfunktionalitäten und Schutzmechanismen dar.

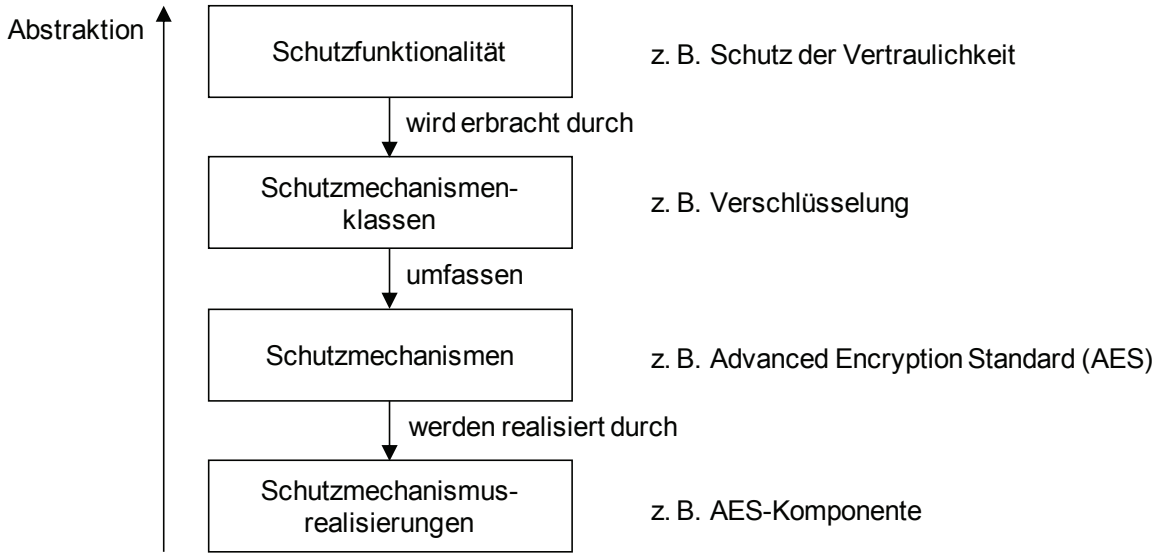

Abbildung 3.2: Zusammenhang zwischen Schutzfunktionalitäten und Schutzmechanismen

Schutzmechanismen basieren auf einer oder mehreren Wirkungsweisen: Prävention, Detektion oder Reaktion [Jahn03]. Absicherung durch Prävention wird durch konstruktive Maßnahmen erreicht, welche Schwachstellen gar nicht erst entstehen lassen oder verhindern, dass Bedrohungen auf Schwachstellen einwirken können. Bei einer Absicherung durch Detektion wird nicht verhindert, dass Bedrohungen auf ein IT-System einwirken, es ist aber möglich, derartige Ereignisse zu erkennen. Um den Eintritt von Schadensfällen zu verhindern, wird Schutz durch Detektion üblicherweise mit Schutz durch Reaktion kombiniert, welcher auf die Bedrohungshandlungen reagiert.

Sicherheitsbezogene Maßnahmen sind immer aufgrund der jeweiligen Gefährdung anzuwenden [Ecke03] [Ande01a]. Daher sind Aktivitäten zur Ermittlung der Schutzfunktionalitäten, welche zur Absicherung der IT-Sicherheit in einem konkreten Fall benötigt werden, sowie zur Umsetzung der Schutzfunktionalitäten mit Schutzmechanismen erforderlich. Diese Aktivitäten werden unter dem Begriff *Security-Engineering* zusammengefasst. Sie werden im folgenden Abschnitt erläutert.

3.3.2 Security-Engineering

Das Security-Engineering umfasst die Tätigkeiten, die erforderlich sind, um sichere Systeme (im Sinne von IT-Sicherheit) zu konstruieren. Um den Ablauf der Tätigkeiten des Security-Engineerings zu beschreiben, werden zumeist Vorgehensmodelle aus der Disziplin des Software-Engineerings herangezogen (insb. Wasserfallmodelle und evolutionäre Modelle) [Ande01a]. Die Tätigkeiten des Security-Engineerings umfassen üblicherweise eine *Bedrohungsanalyse*, eine *Risikobewertung*, einen Schritt für *Schutzentwurf und -realisierung* sowie eine *Schutzvalidierung* [Ande01a]. Abbildung 3.3 stellt die Tätigkeiten des Security-Engineerings in Form eines Wasserfallmodells dar.

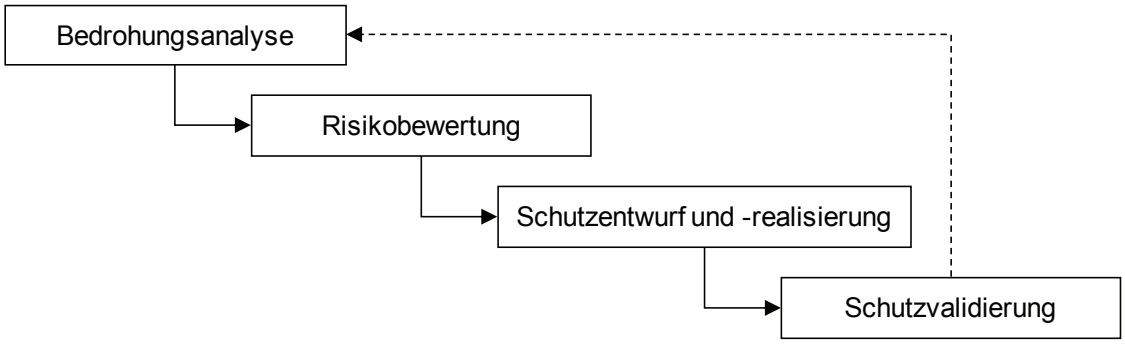

Abbildung 3.3: Tätigkeiten des Security-Engineerings

Im Rahmen der Bedrohungsanalyse werden Bedrohungen der IT-Sicherheit identifiziert. Bedrohungen, die aufgrund einer Schwachstelle wirksam werden können, gefährden das abzusichernde IT-System. Prinzipiell könnte bei der Absicherung der IT-Sicherheit versucht werden, Schutz vor allen ermittelten Gefährdungen zu realisieren. Dies ist in der Praxis aus wirtschaftlichen Gründen jedoch meist nicht möglich [Ande01b].

In der Phase Risikobewertung werden daher die Risiken bestimmt, die sich aus den gefundenen Bedrohungen ergeben. Ist das ermittelte Risiko für das Auftreten eines Schadensfalls durch eine Bedrohung größer als das größte noch vertretbare Risiko (Grenzrisiko), muss das Risiko dieser Bedrohung so weit reduziert werden, bis es kleiner als das Grenzrisiko ist. Dazu werden im Schritt Schutzentwurf und -realisierung geeignete Schutzfunktionalitäten und Schutzmechanismen bestimmt und in das zu schützende System integriert.

Um die Wirksamkeit des eingesetzten Schutzes zu überprüfen, wird das abzusichernde System den Bedrohungen ausgesetzt, vor denen es geschützt werden soll. Dazu werden im Rahmen der Validierung des Schutzes vor der Inbetriebnahme bzw. in Wartungszeiträumen simulierte Angriffe durchgeführt, sogenannte Penetrationstests [Ecke03] [BSI03]. Widersteht der Schutz den Penetrationstests, wird daraus auf die Wirksamkeit des Schutzes geschlossen.

Anders als die Disziplin des Software-Engineerings, welches sich mit der ingenieurmäßigen Entwicklung von Software befasst und dafür Prinzipien, Methoden, Konzepte, Notationen und Werkzeuge bereitstellt [Balz01], werden die durchzuführenden Tätigkeiten im Rahmen des Security-Engineerings als heute noch nicht „methodisch herausgearbeitet“ bezeichnet [Ecke03]. Mangels eigener Security-Engineering-Methoden werden daher häufig Methoden aus dem Bereich der technischen Sicherheit herangezogen, beispielsweise werden in [Ande01a] und [Ecke03] die Methoden *Failure Modes and Effects Analysis* (FMEA) [IEC60812] und *Fault Tree Analysis* (FTA) [IEC61025] genannt.

IT-Sicherheit befasst sich, anders als die Sicherheit im Sinne von *Safety*, mit einer Problemstellung, die sich über die Zeit verändert (Moving-Target-Eigenschaft der IT-Sicherheit). So kann sich die Bedrohungslage ändern oder die eingesetzten Schutzmechanismen können aufgrund neuer Erkenntnisse kompromittiert und somit wirkungslos werden. Aus diesem Grund empfiehlt

es sich, das Security-Engineering regelmäßig während der Lebensdauer des zu schützenden Systems durchzuführen [Ande01a].

In diesem Kapitel wurden die Grundlagen der IT-Sicherheit betrachtet. Es wurde dargestellt, dass sich das Fachgebiet der IT-Sicherheit mit Ereignissen befasst, welche von Menschen oder der Umwelt hervorgerufen werden und den Betrieb von Automatisierungssystemen gefährden. Bedrohungen der IT-Sicherheit können sowohl nichtintentionaler (d. h. zufälliger, fahrlässiger, unvermeidbarer) als auch intentionaler (d. h. absichtlicher, vorsätzlicher, gezielter) Art sein. Da die Betrachtung intentionaler Bedrohungen die Betrachtung nichtintentionaler Bedrohungen stets mit einbezieht, wurde der Fokus der weiteren Arbeit auf die intentionalen Bedrohungen (Angriffe) gelegt. Schutz vor Angriffen wird üblicherweise dadurch erbracht, dass die Schwachstellen, auf welche die Angriffe zielen, beseitigt werden. Dazu werden Schutzfunktionalitäten zur Absicherung einer oder mehrerer Schwachstellen definiert. Schutzfunktionalitäten beschreiben dabei das Ziel der Absicherung, nicht aber, auf welche Weise die Absicherung erbracht werden soll. Die Art und Weise der Absicherung wird von Schutzmechanismen festgelegt, welche zur Erbringung der Schutzfunktionalitäten herangezogen werden, z. B. kryptographische Algorithmen. Zur Absicherung eines konkreten Systems werden die Tätigkeiten des Security-Engineerings durchgeführt: Bedrohungsanalyse und Risikobewertung zur Ermittlung der erforderlichen Schutzfunktionalitäten und Schutzmechanismen, Schutzentwurf und -realisierung zur Umsetzung des Schutzes und Schutzvalidierung, um zu prüfen, ob der Schutz gegen die ermittelten Bedrohungen wirksam ist. Das folgende Kapitel führt die Inhalte der beiden vorangegangenen Kapitel zusammen, indem es zunächst analysiert, wie die Bedrohungen der IT-Sicherheit auf der Feldebene wirksam werden können. Daraufhin wird betrachtet, welche Auswirkungen durch Angriffe auf die IT-Sicherheit der Feldebene hervorgerufen werden können.

4 Gefährdung der IT-Sicherheit auf der Feldebene von Automatisierungssystemen

Die Berücksichtigung von Gefährdungen der IT-Sicherheit auf der Feldebene ist eine vergleichsweise neue Thematik in der Automatisierungstechnik. Die traditionellen Technologien der Feldebene, z. B. analoge Signalübertragungen, können zwar prinzipiell ebenfalls durch Angreifer manipuliert werden, beispielsweise durch die Veränderung einer elektrischen Spannung, welche den Betrag einer gemessenen physikalischen Größe angibt. Dadurch ist eine Beeinflussung von Prozessgrößenmessungen und von Stelleingriffen an einzelnen Stellen möglich. Aufgrund der informationsverarbeitenden Systemelemente auf der Feldebene von modernen Automatisierungssystemen lassen sich jedoch, ausgehend von Erfahrungen in der Standard-IT, wesentlich weitreichendere Möglichkeiten zur Manipulation der IT-Sicherheit vermuten. Dieses Kapitel untersucht daher die Gefährdung der IT-Sicherheit auf der Feldebene von Automatisierungssystemen, indem es die Grundlagen der Kapitel 2 und 3 zusammenführt. Somit werden die Gefährdungen der IT-Sicherheit ermittelt, welche sich aufgrund der spezifischen Eigenschaften der Feldebene von Automatisierungssystemen ergeben.

4.1 Zugriff auf die Informationen der Feldebene

Um Angriffe auf die IT-Sicherheit der Feldebene von Automatisierungssystemen durchführen zu können, welche die Abläufe in einem Automatisierungssystem beeinflussen, muss zunächst Zugriff auf die Informationen der Feldebene erlangt werden. Aufgrund der Position der Feldebene in der Automatisierungspyramide existieren dazu zwei prinzipielle Möglichkeiten: einerseits durch Zugriff, der von einer höheren Ebene der Automatisierungspyramide ausgeht, und andererseits durch physischen Zugriff, d. h. eine Ankopplung an die Systemelemente der Feldebene [WWP04] [GuGö06]. Abbildung 4.1 stellt diese beiden Möglichkeiten zum Zugriff auf die Feldebene dar.

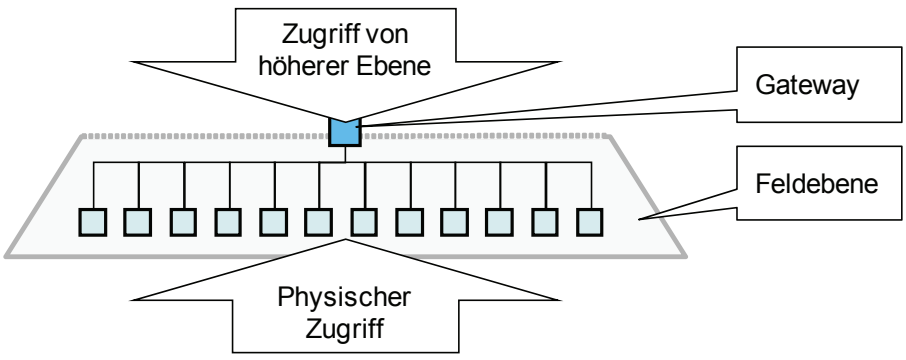

Abbildung 4.1: Möglichkeiten zum Zugriff auf die Feldebene

Aufgrund neuer Anforderungen an Automatisierungssysteme, wie z. B. des Wunsches nach Teleservices [Trau07], nimmt die Vernetzung der einzelnen Ebenen der Automatisierungspyramide immer mehr zu [Hauf06] [Enst02]. In diese Bestrebungen zur vertikalen Integration ist auch die Feldebene einbezogen, d. h., es ist bei modernen Automatisierungssystemen möglich, von höheren Ebenen aus Nachrichten an Feldgeräte zu senden oder von diesen zu empfangen. Da auf der Feldebene andere Kommunikationsprotokolle verwendet werden als auf den darüberliegenden Ebenen, werden Gateways zur Übersetzung zwischen den unterschiedlichen Protokollen eingesetzt (vgl. Abbildung 4.1). Die Vernetzung zwischen den Ebenen der Automatisierungspyramide ermöglicht es jedoch nicht nur legitimen Benutzern, Informationen mit Systemelementen der Feldebene auszutauschen, sondern auch Angreifern. Ein Angreifer, der Zugang zu einer höheren Ebene hat, kann somit über die bestehende Anbindung der Feldebene Angriffe auf die IT-Sicherheit der Feldebene durchführen.

Derartige Anbindungen werden mit Standard-IT-Rechnern realisiert, welche um Feldbusschnittstellen erweitert sind. Somit können diese mit existierenden Mitteln der Standard-IT abgesichert werden [Naed03] [MoSc05], wodurch der Zugriff von Angreifern auf die Feldebene unterbunden wird. Dadurch werden Angriffe auf die IT-Sicherheit der Feldebene, die von höheren Ebenen ausgehen, verhindert. Daher muss die Möglichkeit des Zugriffs über die höheren Ebenen im Rahmen dieser Arbeit nicht weiter betrachtet werden.

Bei physischem Zugang zur Feldebene ist es möglich, beispielsweise Notebook-Computer oder Personal-Digital-Assistants (PDAs) mit Systemelementen der Feldebene zu verbinden. Es existieren am Markt zahlreiche Produkte, mit welchen auf diese Weise Zugriff auf die Informationen gewonnen werden kann, die von Feldbussen übertragen oder in Feldgeräten gespeichert sind, z. B. [Vect07] [Holl04] [Soft03]. Derartige Produkte stehen neben den legitimen Anwendern auch den Angreifern zur Verfügung. Dadurch ist es möglich, Angriffe mit physischem Zugriff auf die Kommunikation oder Funktionalität der Feldebene durchzuführen. Abbildung 4.2 zeigt ein Beispiel für den physischen Zugriff eines Angreifers auf die Systemelemente der Feldebene durch die Ankopplung eines Rechners an einen Feldbus.

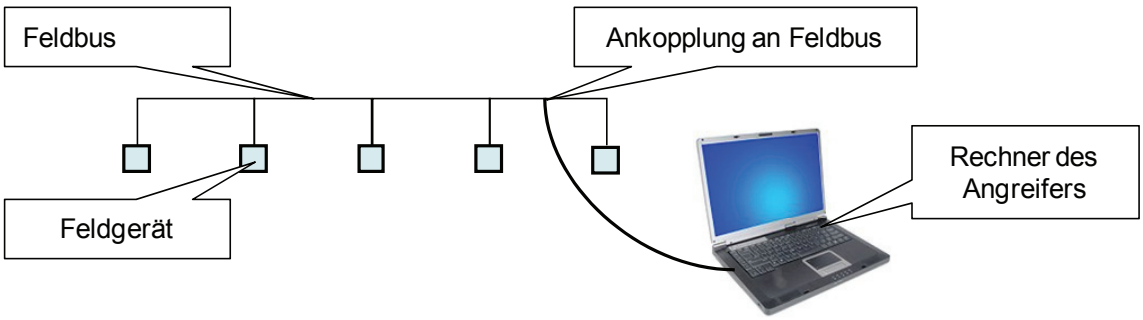

Abbildung 4.2: Beispiel für physischen Zugriff auf die Systemelemente der Feldebene

Bei der Informationsverarbeitung auf der Feldebene werden zwar die Umgebungsbedingungen der Feldebene berücksichtigt (z. B. elektromagnetische Störungen), nicht aber die Möglichkeit

einer böswilligen, beabsichtigten Manipulation. Unterschiedliche Studien belegen, dass die Grundbedrohungen der IT-Sicherheit bei physischem Zugriff auf die Systemelemente der Feldebene wirksam werden können und Auswirkungen auf den technischen Prozess verursachen können, z. B. [WWP04] [Plew06] [Rusc06] [Pech07].

4.2 Auswirkungen von Angriffen auf die IT-Sicherheit der Feldebene

Im Unterschied zu Standard-IT-Systemelementen sind Systemelemente der Feldebene in einen technischen Prozess integriert. Daher betreffen die Auswirkungen von Angriffen auf die IT-Sicherheit auf der Feldebene auch den technischen Prozess.

Durch unbefugten Informationsgewinn (Abhören von Feldbuskommunikation, Auslesen von Feldgerätefunktionalität) werden die Informationen, die auf der Feldebene gespeichert oder übertragen werden, nicht beeinflusst, der technische Prozess verbleibt in einem fehlerfreien Zustand, verhält sich also so, als habe keine Manipulation der IT-Sicherheit stattgefunden. Ein Angreifer kann durch derartige Aktionen aber Informationen gewinnen, welche weitere Manipulationen der Feldgerätefunktionalität oder Feldbuskommunikation erst ermöglichen. Zudem können vertrauliche Informationen gewonnen werden, beispielsweise Rezepturen in der Arzneimittelproduktion.

Durch die Manipulation von Feldgerätefunktionalität durch einen Angreifer weicht die tatsächliche Funktionsweise des Feldgeräts von seiner für die Erfüllung der Anforderungsspezifikation erforderlichen Funktionsweise ab – das Feldgerät wird fehlerhaft. Da das Feldgerät so nicht mehr in der Lage ist, die geforderte Funktion zu erfüllen, können Ausfälle auftreten, d. h. Übergänge von der Korrektheit zu einem Fehler. Diese Ausfälle haben wegen der Integration der Feldgeräte in den technischen Prozess direkten Einfluss auf diesen. Das Automatisierungssystem kann in diesem Fall in einen Fehlerzustand übergehen.

Bei einer Manipulation der Kommunikation verlieren die Feldgeräte zwar nicht ihre *Fähigkeit*, die geforderte Funktion zu erfüllen. Zahlreiche Funktionalitäten, die von Feldgeräten ausgeführt werden, erfordern aber Informationen von anderen Systemelementen. Falls diese Informationen aufgrund einer unautorisierten Modifikation für eine gegebene Situation nicht korrekt sind, handelt das Feldgerät zwar wie spezifiziert, bezogen auf die konkrete Situation aber möglicherweise nicht richtig. Das Automatisierungssystem kann in einem solchen Fall, ebenso wie bei der Manipulation der Feldgerätefunktionalität, in einen Fehlerzustand übergehen.

Ein Fehlerzustand, in den ein Automatisierungssystem aufgrund eines Angriffs auf die IT-Sicherheit übergeht, kann sowohl gefährlich als auch ungefährlich sein [Kona05] [VDI3542-4]. Bei ungefährlichen Fehlerzuständen des Automatisierungssystems wird eine spezifizierte Funktionalität nicht erbracht, die Zuverlässigkeit bzw. die Verfügbarkeit des Automatisierungssys-

tems ist also reduziert. In einem solchen Zustand gehen jedoch keine Gefahren von dem Automatisierungssystem aus. Geht das Automatisierungssystem dagegen in einen gefährlichen Fehlerzustand über, können Unfälle auftreten, welche Personen-, Sach- oder Umweltschäden verursachen [LaGö99a]. Abbildung 4.3 stellt die Zustände und Ereignisse dar, die bei der Manipulation der IT-Sicherheit der Feldebene bei technischen Prozessen auftreten können.

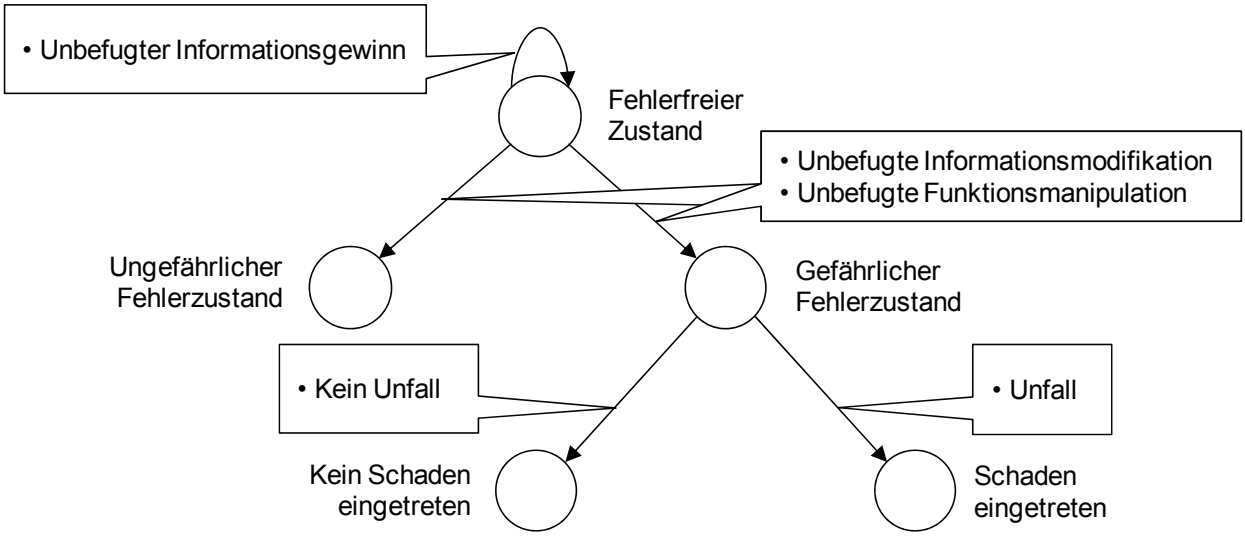

Abbildung 4.3: Zustände und Ereignisse von technischen Prozessen bei Manipulationen der IT-Sicherheit auf der Feldebene

Ein wesentliches Merkmal von Manipulationen der IT-Sicherheit auf der Feldebene ist die Entkopplung des Orts, an dem die Manipulation durchgeführt wird, von dem Ort, an dem die Auswirkungen der Manipulation auftreten. Bei Manipulationen der Feldbuskommunikation können prinzipiell alle an den Feldbus angebundenen Feldgeräte betroffen sein, wodurch diese möglicherweise aufgrund der ihnen vorliegenden Informationen in fehlerhafter Weise in den technischen Prozess eingreifen. Bei einer Manipulation von Feldgerätefunktionalität kann die Auswirkung in dem Teil des technischen Prozesses auftreten, der von dem betroffenen Feldgerät geführt wird. Die Auswirkung kann jedoch auch in einem anderen Teil des technischen Prozesses auftreten, falls dieser von einem Feldgerät geführt wird, welches Informationen verwendet, die von einem Feldgerät mit manipulierter Funktionalität fehlerhaft erzeugt wurden. Abbildung 4.4 stellt die örtliche Entkopplung der Manipulation von Funktionalität sowie der Manipulation von Kommunikation und der Auswirkungen auf den technischen Prozess dar.

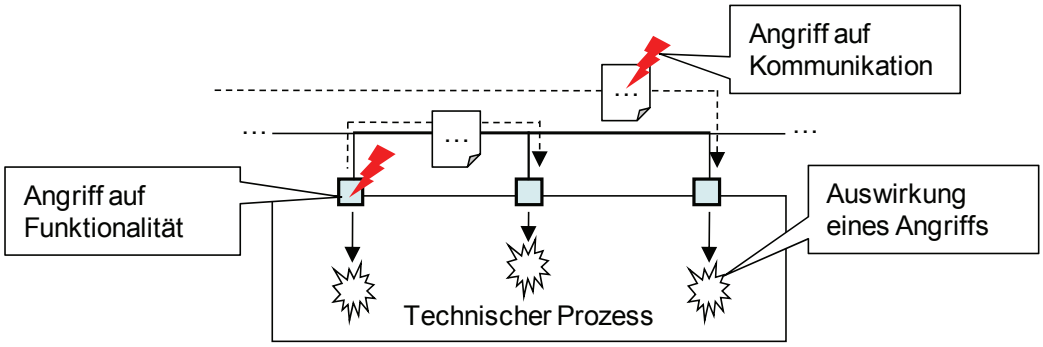

Abbildung 4.4: Auswirkungen von Angriffen auf die IT-Sicherheit der Feldebene

In diesem Kapitel wurden zwei Möglichkeiten dargestellt, durch welche Angreifer den für Manipulationen erforderlichen Zugriff auf die Informationen der Feldebene erlangen können: einerseits mittels Zugriff von „höheren“ Ebene der Automatisierungspyramide, andererseits durch physische Ankopplung an die Systemelemente der Feldebene. Da der unautorisierte Zugriff von „höheren“ Ebenen der Automatisierungspyramide auf die Informationen der Feldebene mit bestehenden Schutzkonzepten der Standard-IT unterbunden werden kann, wird diese Zugriffsform im Folgenden nicht weiter betrachtet. Der unautorisierte Zugriff durch physische Ankopplung an die Feldebene ermöglicht es Angreifern jedoch, Angriffe auf die IT-Sicherheit der Feldebene durchzuführen. Solche Manipulationen bedrohen die technische Sicherheit, die Zuverlässigkeit und – bei reparierbaren Systemen – die Verfügbarkeit von Automatisierungssystemen. Die Manipulierbarkeit der IT-Sicherheit der Feldebene unterminiert dadurch sämtliche Eigenschaften, die erforderlich sind, um volles Vertrauen in die Funktion eines Automatisierungssystems setzen zu können. Aus diesem Grund ist der Schutz der IT-Sicherheit auf der Feldebene zwingend erforderlich. Im folgenden Kapitel werden daher bestehende Ansätze zum Schutz der IT-Sicherheit vorgestellt und auf ihre Eignung zur Absicherung der Feldebene hin untersucht.

5 Ansätze zum Schutz der IT-Sicherheit

Nachdem in den vorangegangenen Kapiteln die Notwendigkeit für einen Schutz der IT-Sicherheit auf der Feldebene begründet wurde, erläutert dieses Kapitel die unterschiedlichen Ansätze zum Schutz der IT-Sicherheit, welche in der Literatur beschrieben und in der Praxis eingesetzt werden. Um den Ansatz zu identifizieren, der zur Absicherung der IT-Sicherheit auf der Feldebene am besten geeignet ist, werden alle Ansätze danach bewertet, ob sie bzw. wie gut sie die Anforderungen erfüllen, die an den Schutz der IT-Sicherheit auf der Feldebene gestellt werden müssen. Daraus ergibt sich der Aufbau des Kapitels: Im folgenden Abschnitt werden zunächst die genannten Anforderungen hergeleitet. Daraufhin werden die Ansätze vorgestellt und anhand ihrer Erfüllung der Anforderungen bewertet. Schließlich werden die Bewertungsergebnisse zusammengefasst und der Bedarf nach einem neuartigen Konzept zum Schutz der IT-Sicherheit auf der Feldebene abgeleitet.

5.1 Anforderungen an den Schutz der IT-Sicherheit auf der Feldebene

Die Anforderungen an den Schutz der IT-Sicherheit auf der Feldebene ergeben sich aus den zwei Themenbereichen dieser Arbeit. Der erste Anforderungskomplex folgt aus der Forderung nach IT-Sicherheit, ohne dass dabei ein bestimmter Anwendungskontext zugrunde gelegt wird. Die zweite Gruppe von Anforderungen ergibt sich aus den Eigenschaften der Feldebene, die von einem Ansatz, der die IT-Sicherheit auf der Feldebene absichern soll, berücksichtigt werden müssen.

5.1.1 Anforderungen aus der allgemeinen IT-Sicherheit

Ein wesentliches Kriterium bei der Auswahl eines Ansatzes zum Schutz der IT-Sicherheit ist die Wirksamkeit des Ansatzes, d. h. die Fähigkeit, Schutz vor den Bedrohungen zu erbringen, welche für das jeweilige Angriffsziel als relevant identifiziert wurden. Im Rahmen dieser Arbeit wird sowohl von nichtintentionalen Einflüssen als auch von zielgerichtet agierenden Angreifern ausgegangen. Zielgerichtet handelnde Angreifer sind prinzipiell in der Lage, die Schwachstellen des IT-Systems und den eventuell bestehenden Schutz der IT-Sicherheit für ihre Zwecke auszunutzen. Daher muss der Schutz der IT-Sicherheit auf der Feldebene gegen nichtintentionale Manipulationen und intentionale Manipulationen durch zielgerichtete Angreifer wirksam sein.

Die Beschaffung von IT-Systemen unterliegt nahezu immer finanziellen Einschränkungen. Um den Verkaufspreis solcher Systeme gering zu halten und somit die Wettbewerbsfähigkeit des Herstellers zu erhalten bzw. zu erhöhen, darf der Schutz der IT-Sicherheit die Kosten für Ent-

wicklung und Produktion des IT-Systems nicht oder nur möglichst wenig erhöhen. Dies ist insbesondere in der heutigen Zeit erforderlich, in der das Problembewusstsein bezüglich IT-Sicherheit oftmals nicht in dem Maße ausgeprägt ist, dass ihr Schutz als Mehrwert erscheint, für den größere Geldausgaben angemessen erscheinen. Der Schutz der IT-Sicherheit auf der Feldebene muss daher wirtschaftlich umsetzbar sein.

5.1.2 Anforderungen aus den Eigenschaften der Feldebene

Die Funktionalitäten der Feldebene sind auf unterschiedliche Feldgeräte verteilt, welche über Feldbusse kommunizieren. Daher muss ein Schutz der IT-Sicherheit auf der Feldebene sowohl Angriffe auf die Funktionalität als auch auf die Kommunikation der Feldebene berücksichtigen.

Wegen der hohen Exponiertheit der Feldebene bei vielen Automatisierungssystemen muss ein Ansatz zum Schutz der IT-Sicherheit auf der Feldebene berücksichtigen, dass Angreifer prinzipiell in der Lage sind, sich an beliebiger Stelle an die Systemelemente der Feldebene anzukoppeln.

Eine weitere Eigenschaft der Feldebene ist die Echtzeitfähigkeit ihrer Systemelemente. Zusätzlich zur Richtigkeit von Informationen auf der Feldebene wird gefordert, dass Feldgerätefunktionalitäten und Feldbuskommunikationen rechtzeitig erbracht werden müssen. Zudem muss gewährleistet sein, dass auf gleichzeitig auftretende Ereignisse reagiert werden kann. Der Schutz der IT-Sicherheit darf die Systemelemente daher nur in solcher Weise beeinflussen, dass die bestehende Echtzeitfähigkeit der Systemelemente erhalten bleibt.

Auf der Feldebene werden unterschiedliche Arten von Feldgeräten und Feldbussen eingesetzt. Diese sind im Allgemeinen nicht zu anderen Feldgerätetypen bzw. Feldbussystemen kompatibel. Da das Ziel der vorliegenden Arbeit nicht die Absicherung eines bestimmten Feldgeräte- oder Feldbusstandards, sondern der Feldebene allgemein ist, wird daher verlangt, dass der Schutz unabhängig von solchen Standards sein muss.

5.2 Vorstellung und Bewertung der Ansätze

Die Beherrschung von nicht autorisierten Informationsmanipulationen ist eine der ältesten Herausforderungen in der elektronischen Datenverarbeitung. Derartige Störungen der IT-Sicherheit können beispielsweise aufgrund von Angriffen, aber auch physikalischen Umgebungseinflüssen oder fehlerhaften Systemelementen (einschließlich Software) hervorgerufen werden. Dadurch können übertragene oder gespeicherte Informationen beispielsweise verändert, fehlerhaft erzeugt oder gelöscht werden. Zum Schutz vor derartigen Störungen existieren in der elektronischen Datenverarbeitung unterschiedliche Ansätze, die abhängig von den gegebenen Randbedingungen eingesetzt werden, gegebenenfalls werden auch mehrere Ansätze kombiniert. Die folgenden Abschnitte stellen die unterschiedlichen Ansätze vor und bewerten diese in Hinblick

auf die Erfüllung der oben dargestellten Anforderungen. Zudem wird mit PROFIsafe beispielhaft ein Protokoll betrachtet, das spezifisch für einen Feldbusstandard konzipiert ist und mehrere der dargestellten Ansätze verwendet.

5.2.1 Physische Zugangsbegrenzung

Beschreibung

Physische Zugangsbegrenzung verhindert, dass Angreifer Zugang zu einem bedrohten System erlangen. Somit ist auch die physische Ankopplung an Elemente dieses Systems nicht möglich und alle dadurch möglichen Angriffe werden bereits im Vorfeld verhindert. Beispiele für physische Zugangsbegrenzung sind Absperrungen von Gebäuden, Räumen und Bereichen sowie Wachposten, welche den Zugang von unbefugten Personen verhindern können.

Bewertung

Aus der Sicht der IT-Sicherheit ist es optimal, Angreifern den physischen Zugang zu verwehren, da somit Manipulationen bereits im Vorfeld verhindert werden. Es kann somit keinerlei Zugriff auf die Informationen des IT-Systems genommen werden. Physische Zugangsbegrenzungen sind von den eingesetzten Technologien unabhängig, da keine Kopplung mit dem zu schützenden System besteht. Aus demselben Grund beeinträchtigt physischer Schutz auch nicht die Echtzeitfähigkeit des abgesicherten IT-Systems.

Der Nachteil der physischen Zugangsbegrenzung zur Absicherung der IT-Sicherheit auf der Feldebene erwächst aus der Exponiertheit und der oftmals weiträumigen Verteiltheit der Systemelemente der Feldebene. Da aufgrund der Entkopplung von Zugriffsort des Angreifers und Auftrittsort der Auswirkungen alle Systemelemente der Feldebene vollständig abzudecken sind, muss die physische Zugangsbegrenzung oftmals einen räumlich weit ausgedehnten Bereich absichern. Dadurch ist eine wirksame physische Zugangsbegrenzung nur sehr kostenintensiv zu realisieren.

Zudem ist die Verwendung physischer Zugangsbegrenzungen nicht möglich, wenn drahtlose Kommunikationssysteme verwendet werden, da elektromagnetische Wellen physische Schutzeinrichtungen üblicherweise durchdringen. Eine weitere Schwäche der physischen Zugangsbegrenzung ist, dass es trotz solchen Schutzes oftmals noch vielen Personen möglich ist, physisch auf die Feldebene von Automatisierungssystemen zuzugreifen, beispielsweise Besuchern und Angestellten des Unternehmens, welches das System betreibt. Eine wirksame und umfassende Absicherung aller Systemelemente der Feldebene, auf die diese potenziellen Angreifer zugreifen können, ist mit physischem Schutz praktisch nicht zu realisieren.

5.2.2 Lineare Fehlererkennung

Beschreibung

Um Veränderungen von Daten zu erkennen, können Ansätze der linearen Fehlererkennung eingesetzt werden. Diese Ansätze verwenden Fehlersicherungsinformationen, welche z. B. vom Sender einer Nachricht erzeugt und vom Empfänger der Nachricht verifiziert werden können. Ergibt sich ein Widerspruch aus dem Abgleich der Fehlersicherungsinformation mit der Nachricht, kann auf eine Veränderung der Nachricht geschlossen werden. Es lassen sich drei unterschiedliche Arten linearer Fehlererkennung unterscheiden [BMB+05] [Will93] [Kowa06]:

- *Paritätsbits* sind einzelne Bitzeichen, die nach einer bestimmten Anzahl von Datenbits in die Daten eingefügt werden.
- Bei *Prüfsummenverfahren* werden die Werte von übertragenen Zeichen addiert.
- Der Grundgedanke von *Cyclic-Redundancy-Checks* (CRCs) ist die Division zweier Polynome, deren Koeffizienten durch die Bits der zu sichernden Daten und einer festgelegten Zahl gebildet werden. Der Divisionsrest wird als Fehlersicherungsinformation verwendet.

Bewertung

Alle Arten der linearen Fehlererkennung sind sehr effizient sowohl in Hardware als auch in Software realisierbar. Daher stellen diese keine großen Anforderungen an die Leistung der Rechner, von denen sie ausgeführt werden, und werden daher auf der Feldebene häufig verwendet. Da keine zusätzlichen oder leistungsfähigeren Feldgeräte benötigt werden, kann die lineare Fehlererkennung sehr wirtschaftlich realisiert werden. Die Berechnung erfolgt über zeitlich deterministische Algorithmen, sodass die Verwendung linearer Fehlererkennung einer bestehenden Echtzeitfähigkeit nicht entgegensteht. Aufgrund ihrer Effizienz kann die lineare Fehlererkennung auch bei sehr kurzen Zeitschranken verwendet werden. Lineare Fehlererkennung ist zudem von bestimmten Systemen unabhängig, sie kommt heutzutage bei einer Vielzahl von Kommunikationssystemen zum Einsatz, um die Wahrscheinlichkeit von Fehlübertragungen zu minimieren [Reiß02] [Buss96].

Betrachtet man lineare Fehlererkennungsmechanismen aus dem Blickwinkel eines intelligenten Angreifers, so stellen sich diese als nicht wirksam heraus. Ein Angreifer kann eine übertragene Nachricht oder ein abgespeichertes Datum trotz linearer Fehlererkennung manipulieren, ohne dass dies bei einer Prüfung festgestellt werden kann. Ein Angreifer, der in der Lage ist, den Inhalt einer Nachricht zu verändern, kann dazu entweder die an die Nachricht angehängte Fehlersicherungsinformation ebenfalls so manipulieren, dass diese zu dem manipulierten Nachrichteninhalt zu „passen“ scheint, oder den Nachrichteninhalt so verändern, dass die sich aus dem veränderten Inhalt ergebende Fehlersicherungsinformation nicht von der des unveränderten In-

halts unterscheidet. In beiden Fällen kann eine so manipulierte Nachricht nicht von einer unverfälschten Nachricht unterschieden werden. Abbildung 5.1 stellt die Manipulation des Nachrichteninhalts und der Fehlersicherungsinformation einer durch CRC-32 [IEEE802.3] geschützten Nachricht dar.

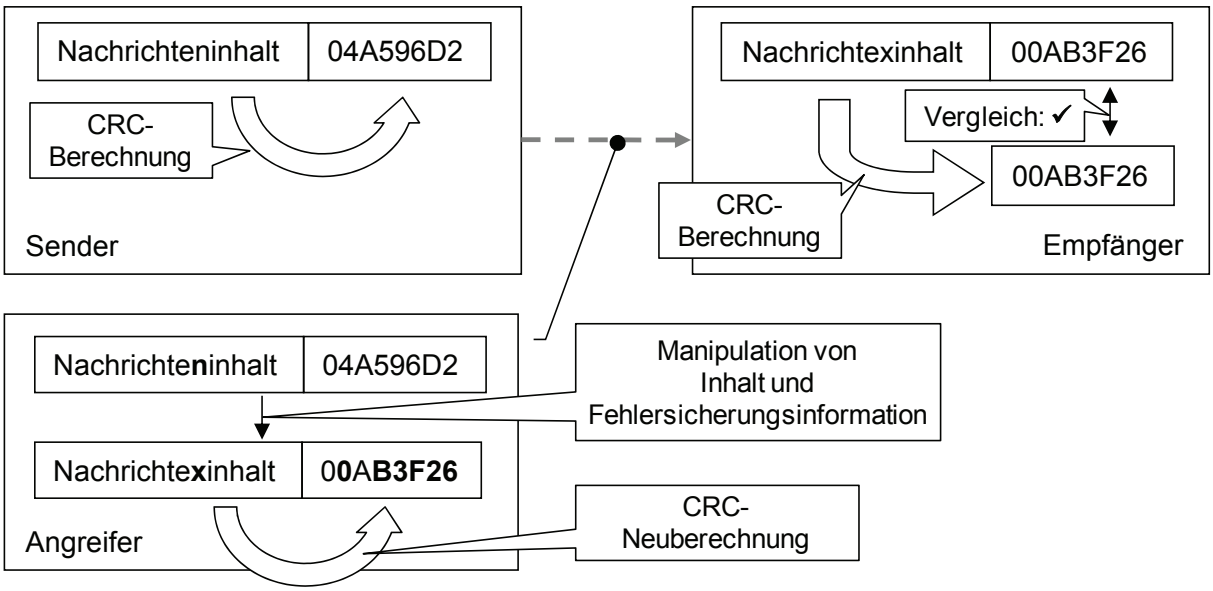

Abbildung 5.1: Manipulation einer mit linearer Fehlererkennung geschützten Nachricht

5.2.3 Zeitliche Überwachung

Beschreibung

Bei Echtzeitsystemen ist neben der Richtigkeit von Informationen auch die Korrektheit ihrer zeitlichen Eigenschaften erforderlich. Störungen dieser zeitlichen Eigenschaften bestehen aus Verzögerung, Löschung (entspricht unendlich langer Verzögerung), Veränderung der Reihenfolge oder Wiederholung von Nachrichten. Derartige Störungen können durch drei unterschiedliche Einflüsse hervorgerufen werden. So können physikalische Effekte beispielsweise Verdopplungen von Nachrichten aufgrund von Echo-Effekten durch unzureichend terminierte Busabschlüsse hervorrufen. Eine weitere Möglichkeit der Störung zeitlicher Eigenschaften ergibt sich aus den Eigenschaften von Kommunikationsprotokollen, welche ein Routing von Nachrichten über unterschiedliche Wege durchführen, wodurch z. B. die Nachrichtenreihenfolge verändert werden kann. Die dritte Möglichkeit besteht in einer Manipulation der Übertragungsstrecke. Um solche Beeinträchtigungen von zeitlichen Eigenschaften erkennen zu können, werden Ansätze zur zeitlichen Überwachung herangezogen. Dabei kommen zum Schutz vor diesen Störungen drei unterschiedliche Arten zeitlicher Überwachung zum Einsatz:

- *Explizite Zeitmarkierungen* sind, analog zu den im vorigen Abschnitt beschriebenen Fehlersicherungsinformationen, Informationen, welche an eine zu übertragende Nachricht angehängt werden und deren Absendezeitpunkt, Gültigkeitsdauer oder relative Po-

sition in einer Nachrichtensequenz angibt, was vom Empfänger ausgewertet werden kann [EN50159-2].

- *Implizite Zeitmarkierungen* werden bei Nachrichten eingesetzt, die in bekannten Zeitabständen eintreffen müssen, beispielsweise Token-Nachrichten, welche den Teilnehmern eines Kommunikationssystems zyklisch das Senderecht einräumen. Trifft eine solche Nachricht nicht zu dem erwarteten Zeitpunkt ein, kann der Empfänger folgern, dass die zeitlichen Eigenschaften der Nachricht gestört wurden [Schr06].

- *Bus-Guardians* sind Bauelemente, welche bei zeitgesteuerten Kommunikationssystemen den Buszugriff von Kommunikationsteilnehmern zu den Zeiten unterbinden, in denen der jeweilige Teilnehmer kein Senderecht besitzt [Flex05b] [Flex05c]. Bus-Guardians verhindern eine Störung der Buskommunikation, wenn ein Kommunikationsteilnehmer zu falschen Zeitpunkten senden möchte, z. B. aufgrund von Softwarefehlern des Teilnehmers (Babbling-Idiot).

Bei allen genannten Ansätzen zur zeitlichen Überwachung der Buskommunikation ist es erforderlich, dass die Mechanismen über eine synchronisierte Zeitbasis verfügen, anhand derer überprüft werden kann, ob die Zeitmarkierungen oder die Sendezeiten der Teilnehmer gültig sind.

Bewertung

Die dargestellten Arten des Schutzes der zeitlichen Eigenschaften von Nachrichten sind einfach in Hardware und Software realisierbar. Sie stellen, ebenso wie die lineare Fehlererkennung, keine großen Anforderungen an die Rechner, auf denen sie ausgeführt werden, und können somit ebenfalls sehr wirtschaftlich realisiert werden. Wegen der einfachen und deterministischen Algorithmen zur Prüfung von zeitlichen Eigenschaften sind diese Ansätze für Echtzeitsysteme verwendbar. Explizite Zeitmarkierungen sind technologieunabhängig und können durch jedes Kommunikationssystem verwendet werden, implizite Zeitmarkierungen erfordern dagegen zyklisch gesendete Nachrichten. Bus-Guardians setzen ein zeitgesteuertes Kommunikationssystem voraus.

Zeitmarkierungen können analog zur linearen Fehlererkennung ebenfalls von Angreifern manipuliert werden, da ein Angreifer zusätzlich zur Veränderung der zeitlichen Eigenschaften auch die Zeitmarkierung entsprechend manipulieren kann, wodurch beispielsweise eine verzögerte Nachricht als rechtzeitig erscheint. Bus-Guardians sind wirkungslos gegen Angriffe, die mittels physischen Zugriffs durchgeführt werden, da Bus-Guardians nur die Sendeversuche unterbinden können, bei denen die Nachricht den Bus-Guardian passieren muss (Abbildung 5.2).

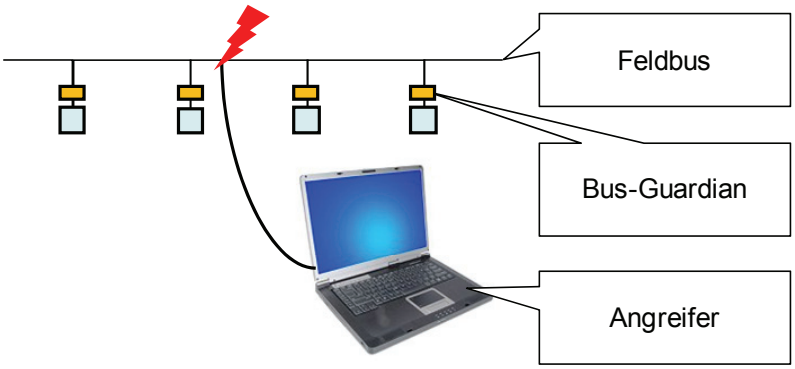

Abbildung 5.2: Unautorisierter Zugriff auf Feldbus mit Bus-Guardians

Zudem sind alle beschriebenen Ansätze der zeitlichen Überwachung prinzipiell gegen Angriffe anfällig, bei denen die Zeitbasis selbst manipuliert wird, beispielsweise durch Manipulation des Synchronisationsmechanismus.

5.2.4 Redundanz

Beschreibung

Eine weitere Möglichkeit, veränderte Informationen zu erkennen, besteht in einer redundanten Auslegung der Struktur des IT-Systems. Dazu werden Module des IT-Systems mehrfach vorgehalten, sodass entweder grundsätzlich mehrere Module eine Aufgabe gleichzeitig durchführen und Informationsveränderungen durch Vergleich erkannt werden (statische Redundanz), oder es werden Module eingesetzt, welche die Aufgaben anderer Module im Fall eines erkannten Modulfehlers übernehmen können (dynamische Redundanz) [LaGö99b].

Unabhängig von der Art, wie die redundante Struktur ausgelegt ist, lässt sich Redundanz sowohl bei Hardware-Bauteilen als auch bei Software-Modulen bzw. bei Informationen allgemein anwenden:

- Bei *Hardware-Redundanz* werden mehrere Hardware-Bauteile, welche dieselbe Aufgabe erfüllen, in das System integriert. Die einzelnen Bauteile können dabei identisch ausgeführt sein. Manipulationen an Informationen werden daran erkannt, dass die manipulierten Informationen nicht mit den Informationen übereinstimmen, die durch die anderen Bauteile erzeugt/kommuniziert werden.

- Bei *Informationsredundanz* werden Informationen mehrfach vorgehalten bzw. erzeugt. Softwareprogramme werden dabei üblicherweise verschiedenartig entworfen und implementiert, damit sich Entwurfs- oder Programmierfehler nicht auf alle Instanzen des Programms auswirken und somit erkennbar werden (Diversität). Bei der Erzeugung von Messwerten in technischen Systemen können Messgrößen an unterschiedlichen Stellen gemessen oder aus abhängigen Messgrößen abgeleitet werden (Messwert-Redundanz).

Ebenso ist es möglich, Messwerte mehrfach in bestimmten Zeitabständen abzufragen (Zeit-Redundanz).

Bewertung

Bei einer Verwendung von Hardware-Redundanz zum Schutz vor Angriffen mit physischem Zugriff müsste jedes Systemelement, auf das ein Angreifer zugreifen kann, mehrfach vorgesehen werden, damit eine Diskrepanz zwischen unmanipulierten und manipulierten Informationen erkennbar wird. Abbildung 5.3 stellt die Detektierbarkeit eines Angriffs auf die Feldebene mit redundanten Systemelementen dar.

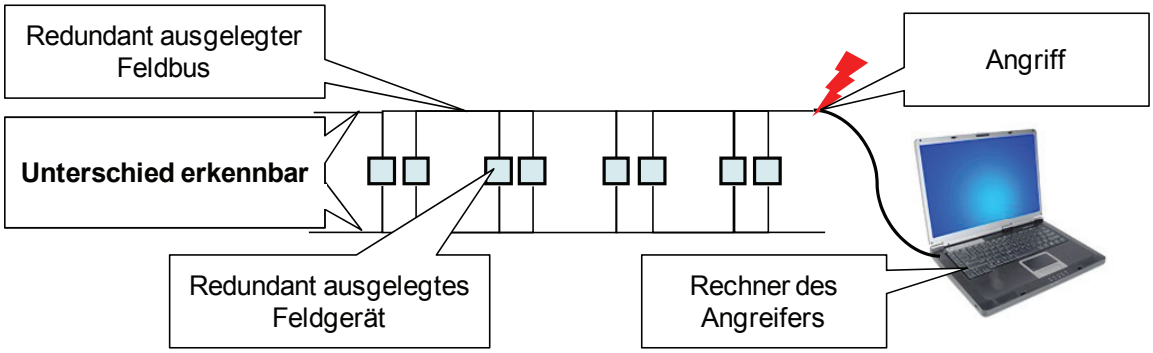

Abbildung 5.3: Detektierbarkeit eines Angriffs auf die Feldebene mit redundanten Systemelementen

Da jedes Systemelement, an das sich ein Angreifer ankoppeln kann, mehrfach vorgesehen werden muss, verursacht der Einsatz von Hardware-Redundanz zur Absicherung der IT-Sicherheit hohe Kosten. Zudem ist es möglich, Angriffe parallel an den redundant ausgelegten Systemelementen durchzuführen, wodurch die Wirksamkeit der Hardware-Redundanz gegen Angriffe mit physischem Zugriff nicht gegeben ist. Abbildung 5.4 stellt dar, wie ein solcher Angriff durch physische Ankopplung an ein redundant ausgelegtes Feldbussystem durchgeführt werden kann.

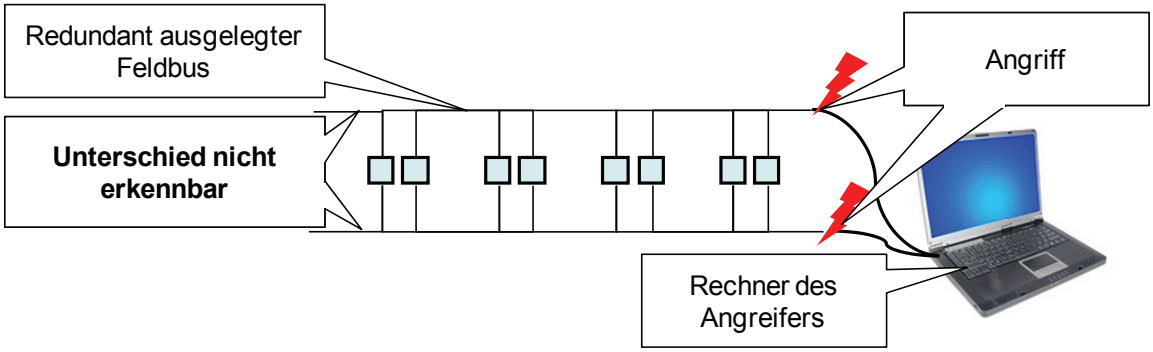

Abbildung 5.4: Angriff auf ein redundant ausgelegtes Feldbussystem

Die Verwendung von Software-Redundanz bietet keinerlei Wirksamkeit gegen unautorisierte Manipulationen, da es bei einer Veränderung von Softwarefunktionalität nicht relevant ist, ob

diese auf anderen Algorithmen beruhen als andere Applikationen, welche dieselbe Funktionalität aufweisen. Messwert- und Zeit-Redundanz sind nur dann gegen Angriffe mit physischem Zugriff wirksam, wenn der Angreifer nicht auf beide Kopien zugreifen kann.

5.2.5 Firewall-Systeme

Beschreibung

Moderne Betriebssysteme erbringen eine Vielzahl von Diensten, welche über Internet-Kommunikationsprotokolle angesprochen werden können. Dadurch ist es bei Rechnern, die an das Internet angebunden sind, einfach möglich, Funktionen durch Aufrufe über das Internet auszuführen. Daher soll im Allgemeinen kein Zugriff (z. B. bei Arbeitsplatz-PCs) oder nur eingeschränkter Zugriff (z. B. bei Web- oder E-Mail-Servern) auf die Dienste eines Rechners ermöglicht werden.

Um den Zugriff auf Dienste von Rechnern zu beschränken, werden Firewall-Systeme eingesetzt. Diese unterteilen Netze in einzelne Zonen mit unterschiedlichen Vertrauenswürdigkeiten und filtern Nachrichten, welche das Firewall-System passieren müssen, sodass kein unautorisierter Zugriff von Zonen niedriger Vertrauenswürdigkeit auf Dienste in Zonen höherer Vertrauenswürdigkeit stattfinden kann [Less06] [Roed02]. Im Fall von unautorisierten Zugriffsversuchen werden geeignete Reaktionen eingeleitet, z. B. Einträge in eine Logdatei zur späteren Auswertung oder die Alarmierung von Systemadministratoren. Abbildung 5.5 stellt eine typische, durch ein Firewall-System geschützte Konfiguration von Zonen mit mehreren Arbeitsplatz-Rechnern und einem Server-Rechner dar.

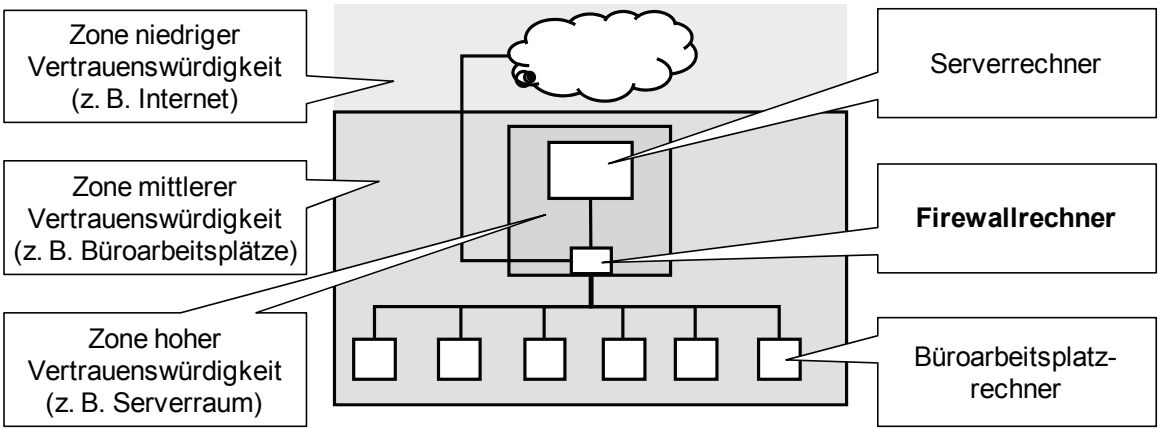

Abbildung 5.5: Typische Firewall-Konfiguration

Firewalls erkennen unautorisierte Nachrichten anhand von Filterregeln, gegen welche die in den Nachrichten enthaltenen IP-Adressen (welche den Ausgangs-/Zielrechner bezeichnen), TCP/UDP-Ports (welche den gewünschten Dienst auf einem Rechner adressieren) sowie weitere Informationen der Internet-Kommunikationsprotokolle geprüft werden [Coul01]. Je nach Ausprägung des Firewall-Systems werden auch Verbindungsinformationen verwertet (zustandsba-

sierte Firewall-Systeme) oder der Inhalt der Applikationsprotokollebene berücksichtigt (Proxy-Firewalls). Die Firewall-Software, welche die oben beschriebene Funktionalität realisiert, wird üblicherweise auf einem eigenen Rechner oder einer spezialisierten Hardware-Plattform (Appliance) ausgeführt, welche in das zu schützende Netz integriert ist (netzbasiertes Firewall-System, vgl. Abbildung 5.5). Zusätzlich zu netzbasierten Firewall-Systemen existieren auch Firewalls, die als Software direkt auf Arbeitsplatz-PCs ausgeführt werden (Personal-Firewall). Die Aufgabe solcher Personal-Firewalls ist die Filterung des eingehenden und ausgehenden Nachrichtenverkehrs auf dem zu schützenden Rechner.

Softwareprogramme für netzbasierte Firewalls sind z. B. *netfilter/iptables* [Netf07] für Linux, *ipfw* [Free07] für FreeBSD und MacOS X sowie der *Microsoft Internet Security and Acceleration Server* [Micr06] für Windows. Verbreitete Firewall-Appliances sind beispielsweise die Watchguard *Firebox*-Serie [Watc07] und die *Firewall-1* der Firma CheckPoint [Chec07a]. Bekannte Personal-Firewalls sind die *Norton Personal Firewall* [Syma07a], *ZoneAlarm* von CheckPoint [Chec07a] und die *Windows-Firewall* von Microsoft [Micr04].

Mit dem zunehmenden Einsatz von Industrial-Ethernet und Internet-Kommunikationsprotokollen in der Automatisierungstechnik werden vermehrt Firewalls für Automatisierungssysteme angeboten. Diese Firewall-Systeme funktionieren ebenso wie „normale“ netzbasierte Firewall-Systeme, beinhalten aber bereits ab Werk Filterregeln für IP-basierte Kommunikationsprotokolle der Automatisierungstechnik, wie PROFINET IO oder EtherCAT. Zudem garantieren sie die Einhaltung von Echtzeitanforderungen. Als Beispiele seien die *SCALANCE-S*-Produkte [Siem07] von Siemens und die *Industrial Firewall 1000* [Lind06] der Firma ads-tec angeführt.

Bewertung

Firewall-Systeme bieten guten Schutz gegen Angriffe, welche über das Internet ausgeführt werden und das Ziel verfolgen, unautorisiert Funktionen auf dem attackierten Rechner auszuführen. Daher sind netzbasierte Firewall-Systeme heute bei Rechnernetzen, die an das Internet angebunden sind, als Standard zu betrachten. Gelingt es einem Angreifer jedoch, ein netzbasiertes Firewall-System zu umgehen und sich physisch an Kommunikationsmedien oder Rechner hinter der Firewall anzukoppeln, können Firewall-Systeme keinen Schutz gegen unautorisierte Zugriffe auf Rechnerdienste bieten (Abbildung 5.6).

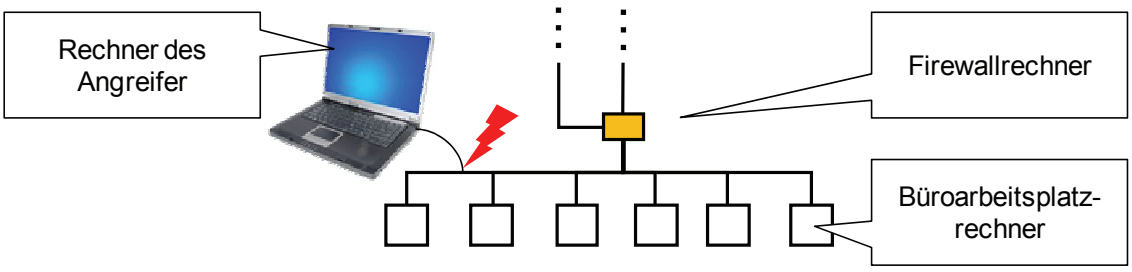

Abbildung 5.6: Physische Ankopplung eines Angreifers hinter der Firewall

Personal-Firewalls sind prinzipiell gut geeignet, unautorisierte Nachrichten abzufangen, bevor diese zur Verarbeitung an eine Software-Applikation übergeben werden und somit schädliche Auswirkungen hervorrufen können. Firewall-Systeme ziehen zur Erkennung solcher Nachrichten jedoch die Eigenschaften der Internet-Protokolle mit ihrem spezifischen IP-Adress/Port-Schema der TCP/IP-Protokolle heran. Diese Eigenschaften existieren bei Feldbussen mit Ausnahme von Industrial-Ethernet nicht. Außerdem sind Firewalls im Allgemeinen nicht in der Lage, inhaltliche oder zeitliche Manipulationen an legitimen Kommunikationsvorgängen festzustellen. Daher bieten Personal-Firewalls nur für eine Teilmenge der im Rahmen dieser Arbeit adressierten Bedrohungen wirksamen Schutz. Zudem bewegt sich die Programmgröße am Markt existierender Personal-Firewalls im Megabyte-Bereich, was die Ressourcenkapazität der meisten Feldgeräte überfordert.

5.2.6 Kryptografische Kommunikationsprotokolle

Beschreibung

Die meisten der heute verwendeten Internet-Kommunikationsprotokolle wurden zu einer Zeit entwickelt, als das Internet aus einer überschaubaren Anzahl von Rechnern bestand, welche von US-amerikanischen Militärangehörigen und Wissenschaftlern verwendet wurden. Wegen dieser vertrauenswürdigen Nutzergruppe war der Schutz der IT-Sicherheit keine Anforderung an die Kommunikationsprotokolle des Internets [Cerf93]. Seit den 1990er-Jahren ist das Internet ein Kommunikationssystem für eine große Zahl Menschen – eine heterogene Nutzergruppe, welche unterschiedliche Tätigkeiten über das Internet durchführt, auch solche, bei denen die Kommunikation gegen Manipulationen abgesichert sein muss.

Zahlreiche Softwareanwendungen erfordern daher die Gewährleistung von Vertraulichkeit und inhaltlicher Integrität bei Kommunikationsvorgängen über das Internet sowie der eindeutigen Identifizierung (Authentifizierung) der beteiligten Kommunikationspartner. Kryptografische Kommunikationsprotokolle verändern daher Nachrichten auf eine Art und Weise, die das Abhören erschweren und die Detektion von Nachrichtenmanipulationen ermöglichen.

Analog zu Firewall-Systemen existieren zwei unterschiedliche Ausprägungen von Kommunikationsprotokollen. Die erste besteht in der Absicherung von Kommunikationsvorgängen zwischen einzelnen Rechnern (z. B. Webseitenabrufe beim Online-Shopping). Dabei wird das kryptografische Kommunikationsprotokoll auf den einzelnen Rechnern ausgeführt. Die andere Ausprägung besteht in der Absicherung der gesamten Kommunikation zwischen lokalen Netzen (z. B. von Unternehmensniederlassungen, welche über das Internet verbunden sind), wobei die Schutzfunktionalitäten auf einem Gateway (Standard-PC oder Appliance) realisiert sind. Diese zweite Ausprägung von kryptografischen Kommunikationsprotokollen wird auch als *Virtual Private Network* (VPN) bezeichnet. Abbildung 5.7 stellt die Absicherung der Kommunikation zwischen Einzelrechnern sowie zwischen lokalen Netzen dar.

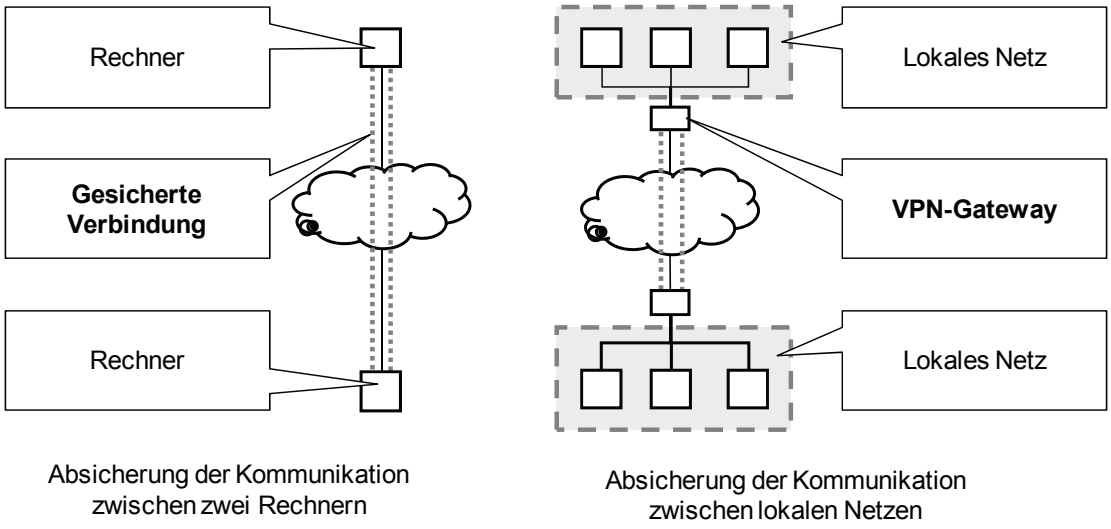

Abbildung 5.7: Kommunikation mit kryptografischen Kommunikationsprotokollen

Beispiele für kryptografische Kommunikationsprotokolle sind die SSL/TLS-Protokollfamilie [Resc01] und IPsec [RFC4301], welche beide sowohl für die Absicherung von Rechner-zu-Rechner-Kommunikation als auch für die Absicherung von VPNs herangezogen werden. Verbreitete Implementierungen von SSL/TLS sind z. B. *OpenSSL* [OSSL07] und *GnuTLS* [GNU07]. Bekannte VPN-Gateway-Softwareprogramme sind *OpenVPN* [OVPN07], *Hamachi* [Hama07] und *OpenSwan* [Xele06]. VPN-Appliances werden u. a. von den Unternehmen Juniper [Juni07], Aventail [Aven07] und F5 Networks [Ffiv07] angeboten.

Bewertung

Kryptografische Kommunikationsprotokolle stellen ein wirksames und bewährtes Mittel dar, um Kommunikationsteilnehmer gegen Manipulationen von Kommunikation zu schützen. Analog zu den Konfigurationen von Firewall-Systemen ist es wegen der physischen Zugriffsmöglichkeit des Angreifers auch bei kryptografischen Kommunikationsprotokollen nicht ausreichend, ein Protokoll auf einem VPN-Gateway zu implementieren. Ein Angreifer kann in diesem Fall Nachrichten manipulieren, bevor sie in den geschützten Bereich zwischen den Gateways eintreten oder nachdem sie diesen wieder verlassen haben. Abbildung 5.8 zeigt die physische Ankopplung eines Angreifers an ein lokales Netz.

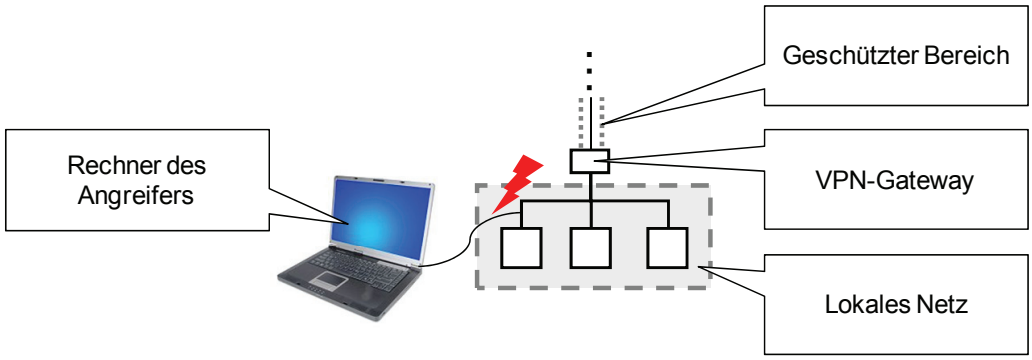

Abbildung 5.8: Physische Ankopplung eines Angreifers an ein lokales Netz

Die zweite Konfiguration, bei der die Schutzfunktionalität auf den einzelnen Rechnern realisiert wird, schließt die ungeschützte Lücke, in der ein Angreifer ansetzen kann. Aufgrund ihrer Zielsetzung der Absicherung von Internet-Kommunikation schützen kryptografische Kommunikationsprotokolle nur unzureichend vor zeitlichen Manipulationen, wie der Verzögerung von Nachrichten. Dies ist dadurch begründet, dass die Internet-Protokolle selbst nicht echtzeitfähig sind. Schließlich sind zum Betrieb von kryptografischen Kommunikationsprotokollen bestimmte Voraussetzungen erforderlich, welche bei Internet-Kommunikation gegeben sind, bei Feldbuskommunikation aber nicht. Hier sind z. B. ein verbindungsorientiertes Transportprotokoll und eine erforderliche Minimallänge einzelner Nachrichten zu nennen. Somit sind die verbreiteten kryptografischen Kommunikationsprotokolle von den Eigenschaften der Internet-Kommunikationsprotokolle abhängig. Des Weiteren weisen die Implementierungen von kryptografischen Kommunikationsprotokollen einen großen Ressourcenbedarf auf, der von dem ausführenden Rechner befriedigt werden muss. Beispielsweise erfordert die OpenSSL-Implementierung[3] des SSL-Protokolls 6,5 MB Speicherplatz. Dies geht weit über die Ressourcenkapazität typischer Feldgeräte hinaus.

5.2.7 Anomalieerkennungsprogramme

Beschreibung

Die in den vorhergehenden Abschnitten beschriebenen Ansätze ermöglichen die Absicherung von Kommunikation. Verfügt ein Angreifer allerdings über die Möglichkeit, Softwareprogramme auf einen Rechner aufzuspielen oder die Konfiguration bestehender Softwareprogramme zu ändern, sind die beschriebenen Ansätze wirkungslos. Aus diesem Grund existieren Programme zur Anomalieerkennung, welche Veränderungen der auf einem Rechner gespeicherten Software erkennen können.

Anomalieerkennungsprogramme lassen sich in Malware-Scanner [Karr02] und Intrusion-Detection-Systeme (IDS) [Buen02] unterteilen. Malware-Scanner sind Programme, welche unerwünschte, gegebenenfalls schädliche Softwareapplikationen identifizieren und beseitigen können. Derartige unerwünschte Applikationen sind z. B. sich selbst reproduzierende und sich selbst verbreitende Schadsoftwareprogramme (Computerviren, Computerwürmer), legitim erscheinende Programme mit versteckten Schadfunktionen (sogenannte trojanische Pferde), Programme, die Informationen über Nutzer sammeln (Spyware), und Programme, die unerwünscht Werbung auf den Bildschirm einblenden (Adware). Malware-Scanner identifizieren ihre Ziele anhand invarianter Signaturen bekannter Malware und heuristischen Untersuchungen von Programmen, um verdächtige Funktionen zu erkennen. Einen ähnlichen Ansatz verfolgen Intrusion-Detection-Systeme. Intrusion-Detection-Systeme versuchen Ereignisse zu erkennen, welche auf

[3] Version 0.9.8

Angriffe, Missbrauchsversuche oder Sicherheitsverletzungen hindeuten [BSI02], wie z. B. charakteristische Veränderungen von Passwortdateien.

Beispiele für Malware-Scanner sind *Client Security* von F-Secure [Fsec07], *Norton AntiVirus* von Symantec [Syma07b] und *Endpoint Security and Control* der Firma Sophos [Soph07]. Verbreitete Intrusion-Detection-Systeme basieren z B. auf den Programmen *Snort* [Snor07] und *Tripwire* [Trip07].

Bewertung

Anomalieerkennungsprogramme ermöglichen die Erkennung von Schadsoftware beispielsweise in E-Mail-Anhängen oder Dateien, die aus dem Internet bezogen wurden. Ebenfalls können bereits auf einem Rechner installierte Schadsoftwareprogramme erkannt werden, was die Einleitung weiterer Maßnahmen ermöglicht, wie die automatische Entfernung oder die Benachrichtigung eines Administrators. Anomalieerkennungsprogramme sind von dem Betriebssystem abhängig, unter dem sie eingesetzt werden. Zudem sind die in Anomalieerkennungsprogrammen eingesetzten Heuristiken oftmals nicht zeitlich deterministisch, wodurch keine Echtzeitfähigkeit besteht[4]. Aufgrund der komplexen Heuristiken und der großen Menge an Schadsoftwaresignaturen erfordern Anomalieerkennungsprogramme eine große Ressourcenkapazität, beispielsweise benötigt Tripwire[5] 4,9 MB Speicherplatz. Eine solche Speicherplatzkapazität ist in den meisten Feldgeräten nicht verfügbar.

5.2.8 Beispiel PROFIsafe

Dieser Abschnitt soll anhand eines konkreten Kommunikationsprotokolls der Feldebene veranschaulichen, wie einige der oben dargestellten Ansätze zum Schutz der Kommunikation auf der Feldebene verwendet werden.

Beschreibung

PROFIsafe [IEC61784-3] ist ein Kommunikationsprotokoll für die Feldbussysteme PROFIBUS DP und PA sowie PROFInet IO mit dem Ziel, nichtintentionale Manipulationen zu detektieren. PROFIsafe unterscheidet sich von dem oben dargestellten Schutz insofern, dass es nicht nur einen, sondern mehrere Ansätze beinhaltet. Die einzelnen Ansätze des PROFIsafe-Protokolls sind Fehlererkennung durch CRC und zeitliche Überwachung mit expliziten sowie impliziten Zeitmarkierungen. Zudem schreibt das PROFIsafe-Protokoll eine eindeutige Identifikation des Absenders von Nachrichten vor. Die einzelnen Ansätze entsprechen dabei den oben dargestellten und werden daher hier nicht weiter erörtert.

---

[4] Sogenannte Real-Time-Anomalieerkennungsprogramme überprüfen Programme, während diese zur Ausführung gebracht werden, machen aber keine Aussage über Zeitgrenzen.

[5] Version 2.3.1.2.0-10

Bewertung

PROFIsafe verfolgt den sinnvollen Gedanken, nicht nur einzelne, sondern mehrere Bedrohungen zu berücksichtigen, welche auf die Feldbuskommunikation einwirken können. Die dabei definierten Ansätze sind effizient und echtzeitfähig umsetzbar. PROFIsafe ist somit ohne wirtschaftliche Nachteile für PROFIBUS und PROFInet verwendbar. Es ist allerdings aufgrund der Ausrichtung auf die PROFIBUS-Familie nicht technologieunabhängig. Aufgrund der Fokussierung auf nichtintentionale Manipulationen sind die einzelnen, in PROFIsafe verwendeten Ansätze nicht sicher in Bezug auf Manipulationen durch einen Angreifer, der über physischen Zugriff verfügt. Somit ist PROFIsafe gegen die im Rahmen dieser Arbeit betrachteten Manipulationen nicht wirksam.

5.3 Zusammenfassung und Auswahl geeigneter Ansätze

Die folgende Tabelle 5.1 fasst die Bewertungen der betrachteten Ansätze zusammen.

Tabelle 5.1: Übersicht über die Bewertungen[6] der Ansätze zum Schutz der Feldebene

	Wirksamkeit (Schutz der Funktionalität)	**Wirksamkeit (Schutz der Kommunikation)**	**Wirtschaftliche Umsetzbarkeit**	**Erhaltung Echtzeitfähigkeit**	**Unabhängigkeit von spezifischen Standards**
Lineare Fehlererkennung	-	-	+	+	+
Zeitliche Überwachung	-	-	+	+	+
Redundanz	-	-	-	+	+
Physischer Zugangsschutz	-	-	-	+	+
Firewall-Systeme (netzbasiert)	-	-	+	-	-
Firewall-Systeme (Personal-Firewall)	-	(+[7])	-	-	-
Kommunikationsschutz-protokolle (Rechner-Rechner)	-	(+[8])	-	-	-
Kommunikationsschutz-protokolle (VPN)	-	-	+	-	-
Anomalieerkennungs-programme	+	-	-	-	-
PROFIsafe	-	-	+	+	-

[6] Legende: + = Kriterium erfüllt, - = Kriterium nicht erfüllt
[7] Falls manipulierte Nachricht erkennbar.
[8] Kryptografische Kommunikationsprotokolle beinhalten die Detektion von manipulierten Kommunikationsvorgängen, aber keine Reaktion.

Aus der in diesem Kapitel dargestellten Untersuchung wird ersichtlich, dass bislang kein Ansatz existiert, der alle Anforderungen erfüllt, welche an den Schutz der IT-Sicherheit auf der Feldebene angelegt werden müssen. Somit wird ein neuartiges Konzept benötigt, welches alle Anforderungen erfüllt. Als Erfolg versprechende Lösung erscheint ein Konzept, welches die Funktionsprinzipien der Ansätze, die eine Wirksamkeit gegen die betrachteten Angriffe erbringen, so kombiniert, dass die Anforderungen der Feldebene erfüllt werden, und welches so gestaltet ist, dass es einen geringen Kostenaufwand verursacht.

Um zu verhindern, dass Angreifer durch manipulierte Nachrichten unerwünschte Aktionen des Automatisierungssystems auslösen, bietet sich das Funktionsprinzip von Firewalls an. Dieses verhindert, dass manipulierte Nachrichten an ein Applikationsprogramm weitergeleitet werden. Um eine solche Filterung durchführen zu können, ist jedoch die Erkennung aller Kommunikationsmanipulationen erforderlich, welche von Angreifern mit physischem Zugriff auf Feldbusse durchgeführt werden können. Dazu sind Firewalls nicht in der Lage, wohl aber kryptografische Kommunikationsprotokolle.

Zur Absicherung vor Veränderungen der Feldgerätefunktionalität ist das Funktionsprinzip von Anomalieerkennungsprogrammen gut geeignet. Es ist jedoch sicherzustellen, dass die Möglichkeiten von Angreifern mit physischem Zugriff auf die Rechnerhardware berücksichtigt werden, wie z. B. den Zugriff auf den Speicher von Feldgeräten über die Programmierschnittstelle. Somit ergeben sich die Funktionsprinzipien der folgenden Ansätze als Ausgangspunkt für ein Konzept zum Schutz der Feldebene:

- Funktionsprinzip von Firewall-Systemen
- Funktionsprinzip von kryptografischen Kommunikationsprotokollen
- Funktionsprinzip von Anomalieerkennungsprogrammen

In diesem Kapitel wurde dargestellt, welchen Anforderungen der Schutz der IT-Sicherheit auf der Feldebene genügen muss. Der Schutz der Feldebene muss gegen Manipulationen zielgerichtet agierender Angreifer wirksam sein. Dabei muss der Schutz berücksichtigen, dass es Angreifern möglich ist, sich physisch an die Feldebene anzukoppeln. Aufgrund des Wettbewerbs, dem die Hersteller von Automatisierungssystemen ausgesetzt sind, muss der Schutz zudem wirtschaftlich umsetzbar sein. Der Schutz soll außerdem unabhängig von spezifischen Standards sein und den (oftmals harten) Echtzeitbedingungen genügen, die von technischen Prozessen vorgegeben werden. Es wurden unterschiedliche Ansätze zum Schutz der IT-Sicherheit vorgestellt und anhand der Anforderungen bewertet. Da kein Ansatz alle Anforderungen erfüllt, wurde der Bedarf nach einem neuartigen Konzept festgestellt. Als Basis für dieses Konzept wurden die Funktionsprinzipien von drei Ansätzen ausgewählt, welche zusammengenommen einen wirksamen Schutz erbringen können, selbst wenn der Angreifer an beliebiger Stelle auf die Feldebene zugreifen kann: die Funktionsprinzipien von Firewall-Systemen, kryptographi-

schen Kommunikationsprotokollen und Anomalieerkennungsprogrammen Im folgenden Kapitel wird nun der Grundgedanke des Schutzes hergeleitet, welcher auf den genannten Funktionsprinzipien basiert.

6 Grundgedanke des Schutzes der IT-Sicherheit auf der Feldebene

In dem vorangegangenen Kapitel wurden bestehende Ansätze zum Schutz der IT-Sicherheit betrachtet. Die Bewertung dieser Ansätze ergab, dass kein bestehender Ansatz alle Anforderungen an den Schutz der IT-Sicherheit auf der Feldebene erfüllt und daher ein neues Konzept zum Schutz der IT-Sicherheit auf der Feldebene erforderlich ist. Daher wird in diesem Kapitel zunächst der Grundgedanke des Schutzes der IT-Sicherheit auf der Feldebene entwickelt, welcher die Wirkungsweise, die örtliche Struktur und die Ausprägung des Schutzes bestimmt. Aufgrund dieser Festlegungen ergeben sich Implikationen für die Anforderungen an das Schutzkonzept für die IT-Sicherheit auf der Feldebene. Daher werden diese Anforderungen so präzisiert, dass das Konzept in dem nachfolgenden Kapitel 7 hergeleitet werden kann.

6.1 Wirkungsweise des Schutzes

Die zur Absicherung der IT-Sicherheit auf der Feldebene gewählten Funktionsprinzipien erbringen ihre Schutzwirkung durch Eingriffe in die Informationsverarbeitung des zu schützenden Systems. Ein solcher *informationstechnischer Schutz* ist nicht in der Lage, Angreifer daran zu hindern, physisch auf die Feldebene zuzugreifen. Es ist Angreifern somit möglich, sich an Systemelemente der Feldebene anzukoppeln und Manipulationen von Informationen durchzuführen, beispielsweise durch Ankopplung an ein Feldbuskabel und die Veränderung von übertragenen Nachrichten oder durch die Ankopplung an ein Feldgerät und das Überschreiben von Programmcode.

Aus dieser Tatsache ergibt sich die Wirkungsweise des Schutzes der Feldebene: Ein präventiver Schutz, welcher Angriffe verhindert oder abfängt, bevor sie die Feldebene erreichen, ist nicht möglich. Stattdessen müssen unerwünschte Auswirkungen, die durch die Angriffe entstehen können, unterbunden werden. Um die Auswirkungen von Angriffen zu unterbinden, müssen Angriffe, welche auf die Feldebene einwirken, zunächst als solche erkannt werden. Im Fall eines Angriffs sind dann entsprechende Reaktionen einzuleiten, welche bewirken, dass der Übergang des Automatisierungssystems in einen unerwünschten Fehlerzustand unterbleibt. Das Automatisierungssystem muss stattdessen in einem bestimmungsgemäßen Zustand verbleiben oder in einen tolerierbaren Fehlerzustand übergehen. Welche Zustände unerwünscht und welche Zustände tolerierbar sind, muss für jedes Automatisierungssystem individuell entschieden werden, beispielsweise müssen gefährliche Fehlerzustände als unerwünscht eingestuft werden. Die folgende Abbildung 6.1 stellt die Wirkungsweise des Schutzes der Feldebene dar.

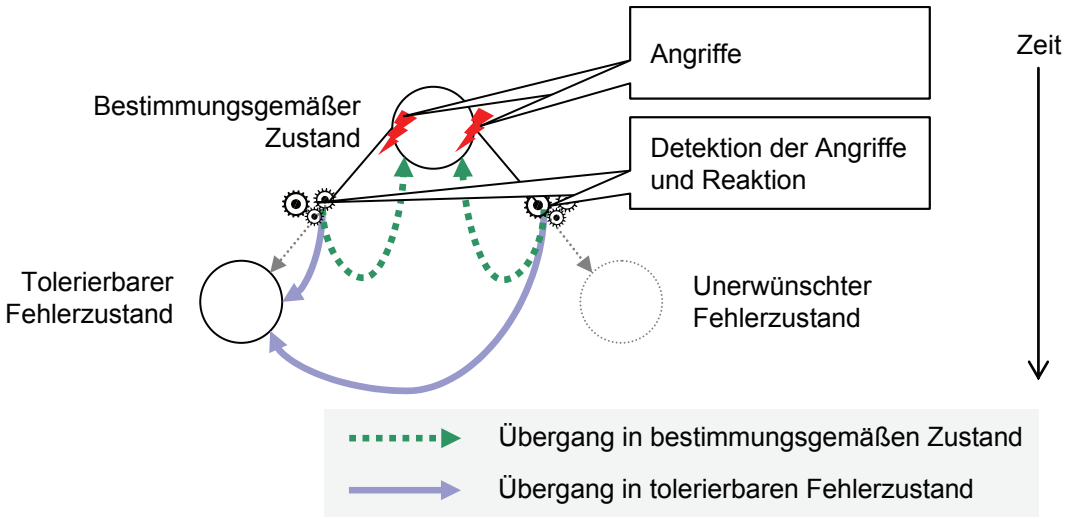

Abbildung 6.1: Wirkungsweise des Schutzes der Feldebene

Um Angriffe erkennen zu können, werden Funktionalitäten benötigt, welche legitime Informationen von manipulierten Informationen unterscheiden können. Zur Unterscheidung von legitimer und manipulierter Feldbuskommunikation wird das Funktionsprinzip von Kommunikationsschutzprotokollen herangezogen. Dieses Funktionsprinzip besteht darin, Kommunikationsvorgänge so zu verändern, dass Manipulationen zuverlässig erkannt werden können. Diese Unterscheidung zwischen legitimen Kommunikationsvorgängen und Angriffen erfolgt mithilfe von Unterscheidungsmerkmalen, welche an geeigneten Stellen erzeugt und überprüft werden müssen.

Zur Detektion von manipulierter Feldgerätefunktionalität wird das Funktionsprinzip von Anomaliedetektionsprogrammen verwendet. Anomaliedetektionsprogramme überwachen den Programmcode der zu schützenden Software, bewerten Veränderungen anhand logischer oder heuristischer Regeln und schließen daraus auf die Art der Veränderung.

Sind Manipulationen der Kommunikation und der Funktionalität erkennbar, können im Fall eines Angriffs Handlungen eingeleitet werden, welche den Übergang des Automatisierungssystems in einen unerwünschten Fehlerzustand verhindern. Dazu wird das Funktionsprinzip von Firewall-Systemen eingesetzt. Firewall-Systeme können unerwünschte Kommunikationsnachrichten blockieren sowie weitere Maßnahmen zur Reaktion auf Angriffe einleiten.

6.2 Örtliche Struktur des Schutzes

Bezüglich der örtlichen Struktur des Schutzes, also der Orte, an denen die Schutzfunktionalitäten ausgeführt werden, sind zahlreiche Varianten zwischen örtlich zentraler Anordnung und örtlich dezentraler Anordnung denkbar. Ein wesentlicher Aspekt des Schutzes ist jedoch, dass dieser wirksam gegen Manipulationen bei physischem Zugriff auf die Systemelemente der Feldebene sein muss. Der Ort, an dem der physische Zugriff des Angreifers auf Systemelemente der Feldebene stattfindet, ist nicht vorhersehbar. Betrachtet man nun die Wirkungsweise des

Schutzes der Kommunikation, also die Detektion von Angriffen anhand geeigneter Unterscheidungsmerkmale, müssen die Orte der Erzeugung dieser Merkmale sowie deren Überprüfung so gewählt werden, dass die Detektierbarkeit von Manipulationen bei einem beliebigen Ankopplungsort des Angreifers gewährleistet ist. Abbildung 6.2 stellt diese Problematik anhand der prinzipiellen örtlichen Verteilung der Ausführungsorte der Schutzfunktionalitäten dar.

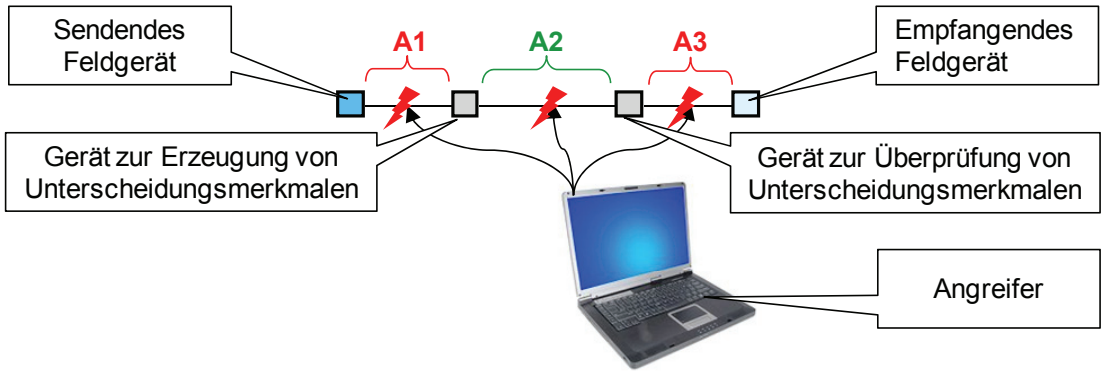

Abbildung 6.2: Abhängigkeit der Schutzwirkung vom Ankopplungsort des Angreifers

Falls am Ankopplungsort des Angreifers die Unterscheidungsmerkmale noch nicht erzeugt wurden (Bereich A1), ist eine Erkennung von Angriffen nicht möglich, da ein später erzeugtes Unterscheidungsmerkmal eine Nachricht absichern würde, die bereits manipuliert ist. Ebenso verhält es sich, wenn sich der Ankopplungsort des Angreifers zwischen dem Ort der Überprüfung der Unterscheidungsmerkmale und dem Ort der Nutzung der übertragenen Informationen befindet (Bereich A3), da die Überprüfung auf eine nicht manipulierte Nachricht schließen lassen würde. Nur wenn die Manipulation durch den Angreifer zwischen dem Ort der Erzeugung und dem Ort der Überprüfung der Unterscheidungsmerkmale stattfindet (Bereich A2), ist eine Entdeckung der Manipulation möglich.

Da der Angreifer sich an jedem Ort an die Feldebene ankoppeln kann, müssen die Orte der Erzeugung bzw. der Überprüfung der Unterscheidungsmerkmale so gewählt werden, dass die unsicheren Bereiche A1 und A3 verschwinden. Dadurch ist eine Detektion von Angriffen über die gesamte Transportstrecke der Nachricht möglich. Eine solche Konfiguration wird erreicht, wenn die Integration der Unterscheidungsmerkmale innerhalb des sendenden Feldgeräts und die Überprüfung der Unterscheidungsmerkmale innerhalb des empfangenden Feldgeräts durchgeführt werden (Abbildung 6.3).

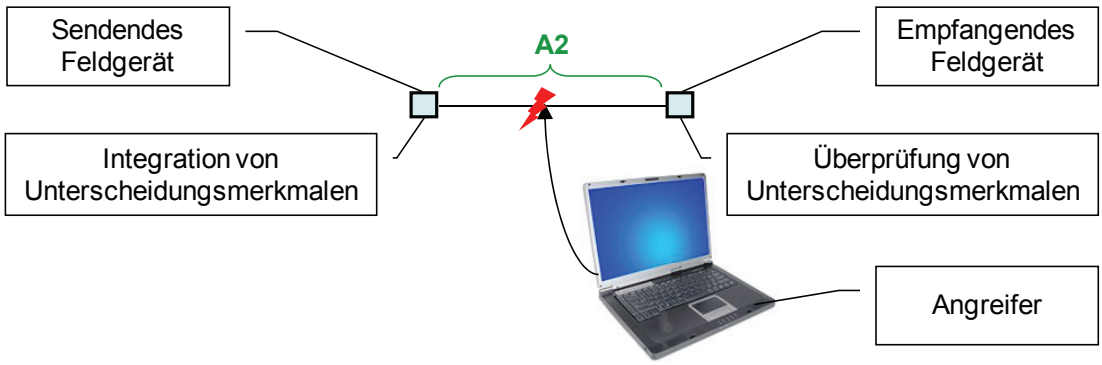

Abbildung 6.3: Ausführung der Schutzfunktionalitäten in den Feldgeräten

6.3 Ausprägung des Schutzes

Die Ausprägung von informationstechnischem Schutz kann entweder in Software, Hardware oder Hardware-Software-Systemen bestehen. Um in die Informationsverarbeitung der Feldebene eingreifen zu können, muss der Schutz, gleich welcher Ausprägung, geeignet in die Feldebene integriert werden.

Da Software ohne Hardware nicht lauffähig ist, müssen bei einer reinen Software-Ausprägung des Schutzes bestehende Rechner der Feldebene zur Ausführung der Schutz-Software verwendet werden. Bei einer Hardware- und einer Hardware-Software-Ausprägung sind zusätzliche Geräte in die Feldebene einzubringen. Im Fall einer reinen Hardware-Ausprägung werden Geräte mit einer „festverdrahteten“ Funktionalität verwendet (z. B. ASICs oder FPGAs), im Fall der Hardware-Software-Ausprägung werden programmierbare Geräte herangezogen (z. B. PCs oder Mikrocontroller). Um die für die gegebene Problemstellung am besten geeignete Ausprägung auszuwählen, werden wiederum die Anforderungen an den Schutz der IT-Sicherheit auf der Feldebene herangezogen. Tabelle 6.1 gleicht die Eigenschaften der unterschiedlichen Ausprägungen mit den Anforderungen ab.

Tabelle 6.1: Bewertung der möglichen Ausprägungen des Schutzes

Anforderung	**Bewertung der Ausprägungen**
Wirksamkeit des Schutzes (Kommunikation und Funktionalität)	Die gewählten Funktionsprinzipien lassen sich mit allen Ausprägungen umsetzen, die Wirksamkeit des Schutzes ist also nicht von der gewählten Ausprägung abhängig.
Erhaltung der Echtzeitfähigkeit	Echtzeitfähige Systeme können in allen Ausprägungen realisiert werden, somit kann diese Anforderung nicht zur Entscheidung für eine Ausprägung herangezogen werden.
Unabhängigkeit von spezifischen Standards	Die unterschiedlichen Feldgeräte- und Feldbusstandards unterscheiden sich auf Hardware-Ebene voneinander. Eine Anpassung zusätzlicher Hardware-Elemente ist aufwändiger als die Anpassung von Software, die immateriell und nicht durch physikalische Gesetze begrenzt ist [Göhn07].
Wirtschaftliche Umsetzbarkeit	Software kann mit minimalen Kosten vervielfältigt werden [Göhn07], wogegen bei der Verwendung zusätzlicher Geräte Kosten für deren Anschaffung auftreten.

Aus dieser Bewertung folgt, dass eine reine Softwarelösung einer Hardware- oder Hardware-Software-Ausprägung aufgrund ihrer einfacheren Anpassbarkeit in Bezug auf die Systemunabhängigkeit überlegen ist. Da die örtliche Struktur des Schutzes so ausgelegt ist, dass der Schutz in den einzelnen Feldgeräten angesiedelt ist, treten keine Kosten für zusätzliche Geräte auf, sodass eine Softwareausprägung auch in dieser Kategorie den anderen Lösungen überlegen ist. Daher wird für den Schutz der Feldebene die Software-Ausprägung gewählt. Bei einer reinen Softwareausprägung müssen die für den Schutz erforderlichen Ressourcen bereitgestellt werden können. Bei der Entwicklung neuer Automatisierungssysteme können diese Ressourcen im Rahmen der Entwicklung vorgesehen und eingeplant werden. Eine Absicherung bestehender Automatisierungssysteme ist ohne zusätzliche Gerätekosten möglich, falls noch ausreichend freie Ressourcen zur Verfügung stehen. Der Grundgedanke des Schutzes der Feldebene besteht somit aus einer Wirkungsweise, welche Schutz durch Detektion von Angriffen und Durchführung von Reaktionen erbringt. Dieser Schutz wird in Form von Software realisiert, welche in die einzelnen Feldgeräte integriert wird.

6.4 Integration des Schutzes in die Feldgerätesoftware

Die Feldgerätesoftware enthält die Softwarebausteine, welche die Funktionalitäten des Feldgeräts realisieren, d. h. die Applikationen des Feldgeräts. Diese setzen auf einer Laufzeitumgebung oder einem Betriebssystem auf. Zur Feldbuskommunikation wird ein Kommunikationsstack verwendet. In Abschnitt 6.3 wurde festgelegt, den Schutz der IT-Sicherheit auf der Feldebene als Software in die Feldgeräte zu integrieren. Daher werden Schutzfunktionalitäten in den Aufbau der Feldgerätesoftware eingefügt. Diese Schutzfunktionalitäten lassen sich, entsprechend der Wirkungsweise des Schutzes, in Funktionalitäten zur *Detektion von Angriffen* und Funktionalitäten zur *Reaktion auf Angriffe* einteilen. Abbildung 6.4 stellt den Aufbau der Feldgerätesoftware, bestehend aus den Feldgerätefunktionalitäten und den Schutzfunktionalitäten, dar.

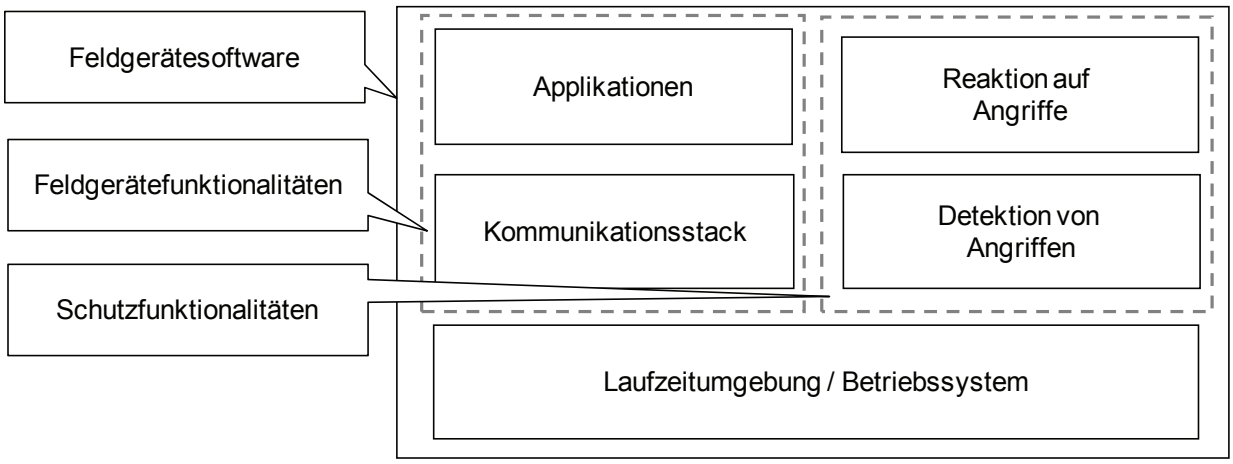

Abbildung 6.4: Aufbau der Feldgerätesoftware

6.5 Präzisierung der Anforderungen an den Schutz der IT-Sicherheit auf der Feldebene

Mit den in den obigen Abschnitten getroffenen Entscheidungen für die Wirkungsweise, die örtliche Struktur und die Ausprägung des Schutzes ergeben sich Implikationen für die Anforderungen an das Konzept zum Schutz der IT-Sicherheit auf der Feldebene. In den folgenden Abschnitten werden die Anforderungen daher so präzisiert, dass das Konzept aus ihnen abgeleitet werden kann.

6.5.1 Absicherung des Schutzes

Das Ziel dieser Arbeit ist es, die Feldebene vor Angriffen auf Kommunikation und Funktionalität zu schützen. Die Wirkungsweise des Schutzes besteht nun darin, dass durch die Integration und Überprüfung von Unterscheidungsmerkmalen Manipulationen von Informationen auf der Feldebene detektiert werden können. Bei einer Detektion von Manipulationen können dann

Reaktionen ergriffen werden, welche Übergänge des Automatisierungssystems in unerwünschte Fehlerzustände unterbinden.

Die Schutzfunktionalitäten und die Unterscheidungsmerkmale sind selbst ebenfalls Informationen, die in Form von Programmcode und Kommunikationsnachrichten auf der Feldebene existieren. Somit sind auch die Schutzfunktionalitäten und die Unterscheidungsmerkmale von einer Manipulation durch einen intelligenten, zielgerichtet agierenden Angreifer bedroht. Aus diesem Grund muss sichergestellt sein, dass die Schutzfunktionalität und die Unterscheidungsmerkmale ebenfalls so ausgelegt sind, dass sie vor Manipulationen geschützt sind.

6.5.2 Anpassbarkeit des Schutzes

Aufgrund der Vielzahl unterschiedlicher, auf der Feldebene verwendeter Technologien soll der Schutz der IT-Sicherheit der Feldebene nicht für eine spezifische Technologie ausgerichtet sein, wie z. B. eine bestimmte Mikrocontrollerarchitektur oder ein bestimmtes Feldbusprotokoll. Ansonsten wäre ein solcher Schutz zwar zur Absicherung dieser Technologie geeignet, für andere Technologien aber nicht anwendbar. Da die Schutzfunktionalitäten in Software ausgeführt werden, ist zunächst erforderlich, dass diese Software auf unterschiedlichen Hardware-Plattformen ausgeführt werden kann. Zudem muss eine Anpassbarkeit der Software gegeben sein, die es ermöglicht, Feldgerätefunktionen in unterschiedlichen Feldgerätesoftwarearchitekturen zu überwachen und in die Kommunikation mit unterschiedlichen Feldbussen integriert zu werden. Da die Schutzfunktionalitäten als Reaktion auf Angriffe Aktionen in den Feldgeräten auslösen müssen, ist eine Möglichkeit vorzusehen, die Schutzfunktionalitäten an unterschiedliche Feldgerätefunktionalitäten anzubinden.

6.5.3 Zeitlicher Determinismus des Schutzes

Die Feldebene von Automatisierungssystemen muss die Echtzeitanforderungen, die von dem geführten technischen Prozess gestellt werden, erfüllen können (Echtzeitfähigkeit). Dies bedeutet, dass Rechen- und Kommunikationsvorgänge auf der Feldebene nicht nur richtig, sondern auch zum korrekten Zeitpunkt erfolgen müssen. Dies muss auch von Systemelementen der Feldebene geleistet werden, welche informationstechnische Schutzfunktionalitäten beinhalten.

Um Echtzeitfähigkeit zu erlangen, werden zwei Forderungen an die Datenverarbeitung der Feldebene gestellt: die Forderung nach Gleichzeitigkeit und die Forderung nach Rechtzeitigkeit. Die Forderung nach Gleichzeitigkeit, d. h. die Reaktion der Feldebene auf gleichzeitig stattfindende Ereignisse in der „Umwelt“, wird entweder durch die Verwendung eines eigenen Rechners für jedes gleichzeitige Ereignis erfüllt oder durch die Verarbeitung gleichzeitiger Ereignisse durch Operationen, welche mit kurzen Zeitabständen nacheinander ausgeführt werden. Zur Erbringung von Rechtzeitigkeit muss gewährleistet sein, dass Tätigkeiten innerhalb festgelegter Zeitschranken abgeschlossen werden (nicht zu früh und nicht zu spät). In der Praxis ist nur die

Einhaltung der späteren Zeitschranke schwierig zu erreichen, da ein Warten bei verfrühtem Abschluss einer Operation einfach möglich ist.

Beide Prinzipien werden heutzutage so umgesetzt, dass die Tätigkeiten in einzelne Rechenprozesse aufgeteilt werden, deren zeitlicher Ablauf entweder vor der Ausführung geplant (synchrone Programmierung) oder während der Ausführung organisiert wird (asynchrone Programmierung) [LaGö99a]. Bei der synchronen Programmierung muss a priori bekannt sein, welche Ausführungsdauern die einzelnen Prozesse aufweisen, um eine Planung vornehmen zu können. Soll beim Einsatz von asynchroner Programmierung sichergestellt sein, dass die definierten Zeitschranken eingehalten werden können, müssen Schedulability-Tests erfüllt werden, welche wiederum Kenntnis über die Ausführungsdauern der einzelnen Prozesse erfordern [LiLa73]. Aus diesem Grund müssen auch von den Schutzfunktionalitäten die Ausführungsdauern bekannt sein. Dazu ist es erforderlich, dass sich die Schutzfunktionalitäten stets zeitlich vorhersagbar (zeitlich deterministisch) verhalten.

6.5.4 Effizienz des Schutzes

Die Schutzfunktionalitäten werden als Software ausgeführt. Die Vervielfältigung der Schutzfunktionalitäten wirkt aufgrund ihrer geringen Kosten der wirtschaftlichen Umsetzung des Schutzes nicht entgegen. Aufgrund der Ausführung in den bestehenden Feldgeräten fallen auch keine Kosten für zusätzliche Rechner an; es ist jedoch zu berücksichtigen, dass sich die Eigenschaften der Schutzfunktionalitäten mit denen der Feldgeräte vereinbaren lassen müssen.

Wie in Abschnitt 2.3.1 dargestellt, besitzen Feldgeräte im Vergleich zu Standard-IT-Rechnern (PCs) zumeist nur einen Bruchteil der Ressourcen Speicherplatz und Rechenkapazität. Auch Feldbussysteme sind oftmals eher für zeitlich deterministischen Datentransfer mit geringen Latenzzeiten ausgelegt als für die Bereitstellung einer großen Kommunikationsbandbreite (vgl. Abschnitt 2.3.2). Zudem dienen die Schutzfunktionalitäten keinem Selbstzweck, sondern müssen zusätzlich zur bereits vorhandenen Funktionalität in den Feldgeräten ausgeführt werden. Dies schränkt die für die Schutzfunktionalitäten zur Verfügung stehenden Ressourcen ein.

Zur Lösung dieses Dilemmas existieren zwei Möglichkeiten: Die erste Möglichkeit besteht in der Verwendung von Rechnern und Kommunikationssystemen, welche über ausreichend Ressourcen für die Ausführung der Schutzfunktionalitäten verfügen. Die zweite Möglichkeit besteht in der Reduktion des Ressourcenbedarfs, d. h. der Erreichung einer so hohen Effizienz [ISO9126-1], dass die Schutzfunktionalitäten in Feldgeräten zusätzlich zur bereits vorhandenen Funktionalität ausgeführt werden können.

Bei der ersten Möglichkeit entstehen aufgrund des Zwangs zur Verwendung von leistungsfähigeren und somit teureren Systemelementen erhebliche Mehrkosten bei der Realisierung der Feldebene von Automatisierungssystemen. Bei einer Erhöhung der Effizienz der Schutzfunktio-

nalitäten, sodass die Schutzfunktionalitäten von üblichen Feldgeräten ausgeführt werden können, entstehen dagegen keine zusätzlichen Gerätekosten.

Geht man davon aus, dass jedes ernst zu nehmende Konzept zur Absicherung der IT-Sicherheit auf der Feldebene einen wirksamen Schutz vor Angreifern bietet, so ergibt sich aufgrund der Wettbewerbssituation unter den Herstellern von Automatisierungssystemen die Wirtschaftlichkeit eines solchen Konzepts als wesentliches Kriterium für dessen Auswahl. Aus diesem Grund wird im Folgenden die Effizienz des Konzepts als zentrales Ziel betrachtet.

6.5.5 Korrektheit der Schutzimplementierung

Treten bei der Ausführung der Schutzfunktionalitäten Fehlerzustände auf, besteht die Möglichkeit, dass dadurch die IT-Sicherheit auf der Feldebene kompromittiert wird. Es existieren zahlreiche Fälle aus der Praxis, bei denen Implementierungsfehler in den Schutzfunktionalitäten von Computersystemen eben dazu ausgenutzt wurden, was durch die Schutzfunktionalitäten eigentlich hätte verhindert werden sollen. Ein Beispiel hierfür ist das Hervorrufen und Ausnutzen von *Buffer Overflows* [Müll07]. Dies kann dazu führen, dass beliebige, vom Angreifer ausgewählte Funktionen ausgeführt werden. Derartige Manipulationen waren beispielsweise bei dem Checkpoint-Produkt *Firewall-1* möglich [VáCa07].

Die Korrektheit der Implementierung der Schutzfunktionalitäten ist daher von großer Bedeutung für die Wirksamkeit des Schutzes. Da im Rahmen dieser Arbeit ein neuartiger Ansatz verfolgt wird, dessen Implementierung nicht aus existierenden und entsprechend reifen Produkten per „Copy & Paste“ übernommen werden kann, muss auch die Qualität der Implementierung des neuen Ansatzes explizit berücksichtigt werden. Deshalb ist die Implementierung des Schutzes auf eine Art und Weise durchzuführen, welche eine möglichst hohe Korrektheit der Schutzfunktionalitäten ergibt.

In diesem Kapitel wurde zunächst der Grundgedanke des neuartigen Konzepts zum Schutz der Feldebene hergeleitet. Aufgrund der Eigenschaften und Randbedingungen der Feldebene muss dieses Konzept Schutzfunktionalitäten verwenden, die informationstechnischer Art sind und als Software in den Feldgeräten ausgeführt werden. Da es somit nicht möglich ist, Angriffe auf die Feldebene im Voraus zu verhindern, besteht die Wirkungsweise des Schutzes in der Detektion von Angriffen und Durchführung von Reaktionen auf diese Angriffe. Dadurch wird der Übergang des angegriffenen Automatisierungssystems in unerwünschte Fehlerzustände unterbunden. Die Anforderungen, welche an den Schutz der Feldebene anzulegen sind, wurden daraufhin unter Berücksichtigung dieser Festlegungen so präzisiert, dass der Feinentwurf des Konzepts zum Schutz der IT-Sicherheit auf der Feldebene, dessen Realisierung und eine Validierung möglich werden. Als wesentliches Ziel wurde dabei die Effizienz des Schutzes identifiziert. Das Konzept dieses effizienten Schutzes wird im folgenden Kapitel dargestellt.

7 Konzept des Schutzes der IT-Sicherheit auf der Feldebene

In dem vorangegangenen Kapitel wurde der Grundgedanke des Schutzes dargestellt, der von den Feldgeräten in Software ausgeführt wird. Als zentrale Herausforderung wurde dabei die erforderliche Effizienz identifiziert. In diesem Kapitel wird daher das Konzept hergeleitet, welches wirksamen Schutz unter optimaler Ausnutzung der vorhandenen Ressourcen erbringt und somit eine hohe Effizienz erzielt. Dazu ist es zunächst erforderlich, zu analysieren, welche Schutzfunktionalitäten benötigt werden, um alle Angriffe detektieren zu können, die mit physischem Zugriff auf die Feldebene durchführbar sind. Ebenso ist zu betrachten, welche Reaktionen existieren müssen, um einen Übergang des Automatisierungssystems in unerwünschte Fehlerzustände zu unterbinden. Da eine optimale Ausnutzung der verfügbaren Ressourcen erreicht werden soll, ist die Umsetzung der erforderlichen Funktionalitäten zum Schutz der IT-Sicherheit auf der Feldebene zu diskutieren. Dazu werden für die Feldebene geeignete Klassen von Schutzmechanismen ausgewählt. Da wirksame Schutzmechanismen einen für verbreitete Feldgeräte hohen Ressourcenbedarf aufweisen, wird untersucht, auf welche Weise dieser Ressourcenbedarf zu reduzieren ist, um die geforderte Effizienz zu erreichen. Damit die Strukturen, die aus dieser Untersuchung folgen, auf unterschiedliche Automatisierungssysteme übertragen werden können, wird eine Softwarearchitektur definiert, welche die erforderlichen Softwarebausteine sowie deren Verbindungen beschreibt.

7.1 Detektion von Angriffen

In Abschnitt 4.2 wurde dargestellt, welche Auswirkungen durch unterschiedliche Arten von Angriffen auf die IT-Sicherheit der Feldebene auftreten können. In den folgenden Abschnitten werden diese Angriffsarten so weit verfeinert, dass Schutzfunktionalitäten identifiziert werden können, welche eine Detektion der jeweiligen Angriffe ermöglichen. Die untersuchten Angriffsarten ergeben sich aus der Übertragung der in Abschnitt 3.2.1 vorgestellten Grundbedrohungen *unautorisierte Informationsgewinnung*, *unautorisierte Manipulation von Kommunikation* und *unautorisierte Manipulation von Funktionalität* auf die Feldebene.

7.1.1 Unautorisierte Informationsgewinnung

Durch die physische Ankopplung an Feldbusse oder Feldgeräte ist es für Angreifer möglich, die dort anliegenden physikalischen Größen zu erfassen. Abbildung 7.1 stellt dies beispielhaft anhand der Ankopplung eines Angreifers an einen Feldbus dar.

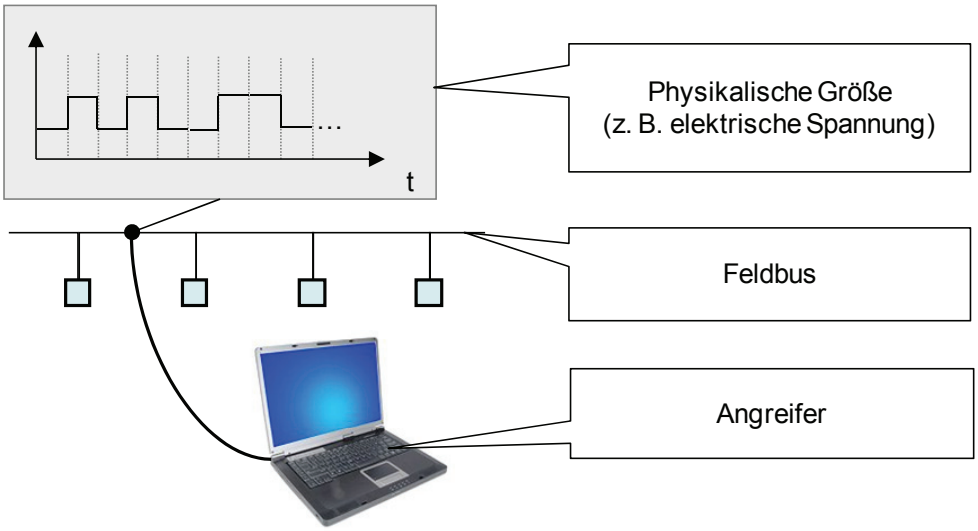

Abbildung 7.1: Erfassung der auf einem Feldbus anliegenden physikalischen Größe

Die Spezifikationen der meisten Feldbusprotokolle sind öffentlich zugänglich. Mit Kenntnis über den Aufbau des verwendeten Kommunikationsprotokolls ist es einfach, die in den abgehörten Daten beinhalteten Informationen zu extrahieren. Aber auch ohne Kenntnis über den Protokollaufbau ist es möglich, durch Beobachtung eines Automatisierungssystems Rückschlüsse auf die in den Daten abgelegten Informationen zu ziehen [Plew06] [Pech07].

Die Geheimhaltung der Spezifikationen der Systemelemente auf der Feldebene kann daher ein Abhören oder andere Angriffe auf die Feldebene nicht wesentlich erschweren oder gar verhindern. Es ist zudem ein anerkannter Grundsatz der IT-Sicherheit, dass die Sicherheit (im Sinne von Security) eines Systems oder eines Verfahrens nicht von der Geheimhaltung der Spezifikation des Systems bzw. des Verfahrens abhängen darf (Security through Obscurity) [Schn06].

Eine Informationsgewinnung durch Abhören führt keine Zustandsänderung des Automatisierungssystems herbei. Daher wird ein Automatisierungssystem durch Abhören zwar nicht unmittelbar gefährdet, die so gewonnenen Informationen können aber dazu verwendet werden, weitere Angriffe vorzubereiten, welche das Automatisierungssystem in einen unerwünschten Fehlerzustand bringen würden. Außerdem können abgehörte Informationen Geschäftsgeheimnisse beinhalten, z. B. Rezepturen für die Arzneimittelproduktion.

Aus diesen Gründen muss das Schutzkonzept eine unautorisierte Informationsgewinnung unmöglich machen. Da aufgrund des physischen Zugriffs des Angreifers nicht verhindert werden kann, dass die physikalischen Größen auf dem Kommunikationsmedium erfasst werden, muss die Extraktion von Informationen aus diesen Daten mithilfe einer Schutzfunktionalität zum *Schutz der Vertraulichkeit* unterbunden werden.

7.1.2 Unautorisierte Manipulation von Kommunikation

Die physische Ankopplung durch Angreifer an Feldbusse ermöglicht Angreifern außer der Erfassung der physikalischen Größen, die auf dem Übertragungsmedium anliegen, auch deren Beeinflussung. Auf der Feldebene wird für die Kommunikation gefordert, dass Informationen korrekt übertragen werden und dass die Übertragung der Informationen innerhalb vorgegebener zeitlicher Grenzen geschieht. Aus diesen Forderungen ergeben sich Ansatzpunkte für Angreifer.

7.1.2.1 Erzeugung eigener Nachrichten

Ein Ansatz, die Übermittlung korrekter Informationen anzugreifen, ist die Erzeugung eigener, unkorrekter Informationen durch den Angreifer. Durch physischen Zugriff auf Feldbusse sind Angreifer in der Lage, Signale in das Feldbusmedium zu induzieren. Diese Signale müssen dabei nicht dem Feldbusprotokoll entsprechen. Solche Manipulationen entsprechen nichtintentionalen Störungen, wie sie auch durch starke elektromagnetische Strahlung hervorgerufen werden. Feldbusse sind für den Einsatz in Bereichen ausgelegt, in denen mit derartigen Beeinflussungen zu rechnen ist. Daher werden solche Manipulationen durch Feldbussysteme mit hoher Wahrscheinlichkeit erkannt.

Um eine Erkennung von Manipulationen zu umgehen, ist es für Angreifer aber möglich, spezifikationskonforme Nachrichten auf den Feldbus aufzubringen. Eine solche Erzeugung eigener Nachrichten kann durch das Ersetzen eines Feldgeräts durch den Angreifer erfolgen. Aufgrund der Ausprägung von Feldbussen als serielle Kommunikationsmedien, an die mehrere Feldgeräte angebunden werden können, ist eine Ankopplung durch einen Angreifer oftmals auch möglich, ohne zuvor ein Feldgerät abzutrennen. Abbildung 7.2 stellt beide Ankopplungsarten dar.

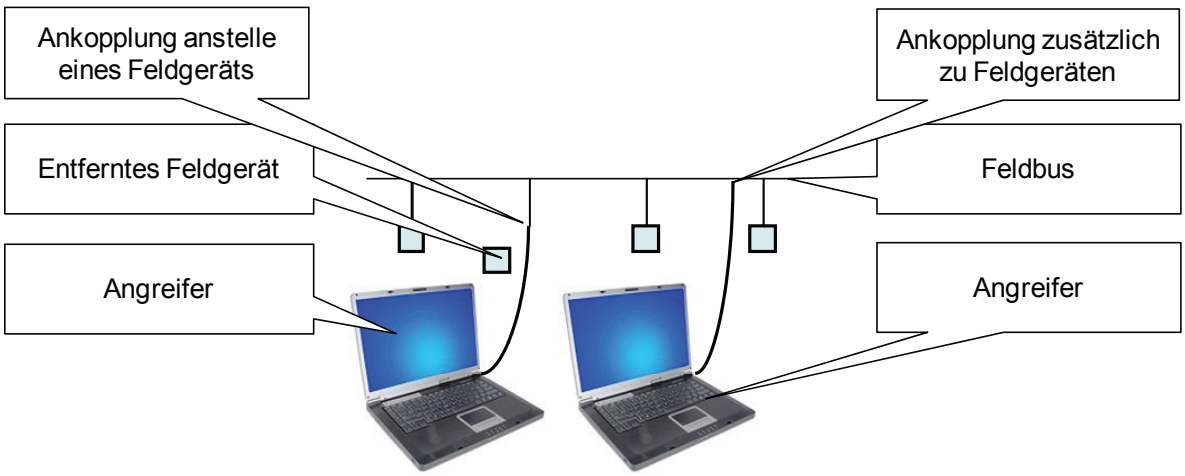

Abbildung 7.2: Ankopplung von Angreifern an einen Feldbus

Das Versenden eigener, spezifikationskonformer Nachrichten durch Angreifer kann den Zustand des Automatisierungssystems so verändern, dass ein Übergang des Automatisierungssystems in einen unerwünschten Fehlerzustand bewirkt wird. Um solche Manipulationen erkennen zu kön-

nen, wird daher eine Funktionalität zum *Schutz der Autorisierung* benötigt, welche sicherstellt, dass jede von einem Empfänger akzeptierte Kommunikationsnachricht von einem Absender stammt, der zu der Versendung der jeweiligen, in der Nachricht enthaltenen Information berechtigt ist.

7.1.2.2 Veränderung des Nachrichteninhalts

Ein weiterer Ansatz, die Feldbuskommunikation zu manipulieren, besteht in der Veränderung des Inhalts von legitimen Feldbusnachrichten. Dies kann durch die Auftrennung des Feldbusmediums und die Integration eines Rechners, welcher die aufgetrennten Bussegmente miteinander verbindet, erreicht werden (serielle Ankopplung). Ein derartig in die Feldebene integrierter Rechner ist in der Lage, den Inhalt von weitergeleiteten Nachrichten zu verändern. Moderne Standard-IT-Rechner sind leistungsfähig genug, um solche Manipulationen ohne für das Feldbussystem wahrnehmbare Verzögerungen durchzuführen. Eine derartige Manipulation des Nachrichteninhalts zeigt Abbildung 7.3.

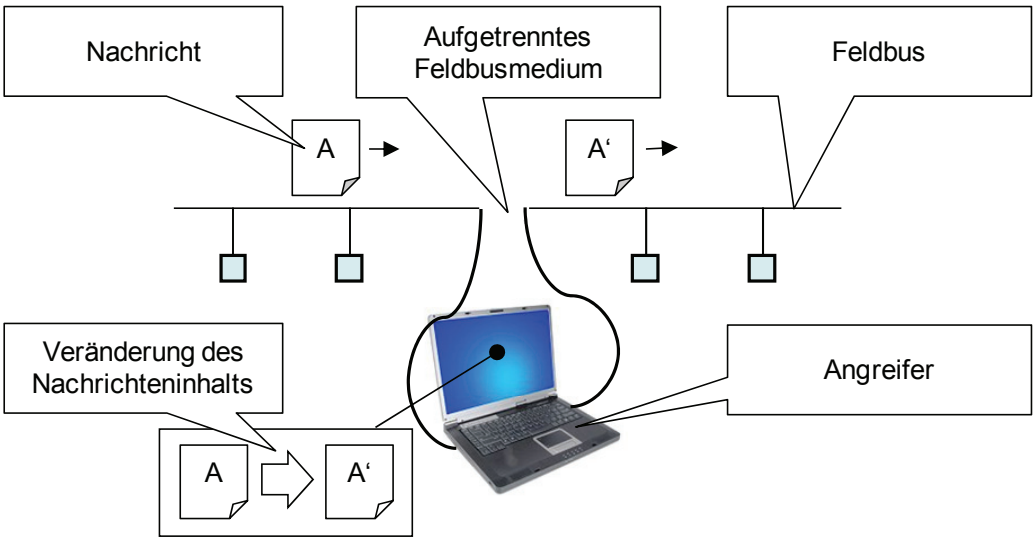

Abbildung 7.3: Serielle Ankopplung eines Angreifers zur Manipulation des Nachrichteninhalts

Derartige Veränderungen bewirken, dass dem Empfänger der veränderten Nachrichten ein Zustand des Automatisierungssystems vorgetäuscht wird, welcher nicht der tatsächlichen Situation entspricht. Trifft der Empfänger einer veränderten Nachricht bei der Führung des technischen Prozesses Entscheidungen, die auf dem vorgetäuschten Zustand beruhen, kann das Automatisierungssystem in einen unerwünschten Fehlerzustand übergehen. Daher ist ein *Schutz der inhaltlichen Integrität* von Feldbusnachrichten erforderlich.

7.1.2.3 Veränderung der zeitlichen Eigenschaften von Nachrichten

Aus den Echtzeitanforderungen des technischen Prozesses ergeben sich für Angreifer weitere Ansatzpunkte zur Manipulation der Feldbuskommunikation. Nachrichten können verzögert,

gelöscht[9] oder wiederholt (Replay-Angriff) werden, d. h., die zeitlichen Eigenschaften der Nachrichten werden verändert. Der Inhalt von Nachrichten wird bei Manipulationen der zeitlichen Eigenschaften nicht verändert. Solche Angriffe können ebenfalls durch die serielle Ankopplung eines Rechners an den Feldbus durchgeführt werden, wobei der Rechner Nachrichten verzögert weiterleiten, mehrfach weiterleiten oder löschen kann. Abbildung 7.4 stellt beispielhaft die Wiederholung von Nachrichten durch einen Angreifer dar.

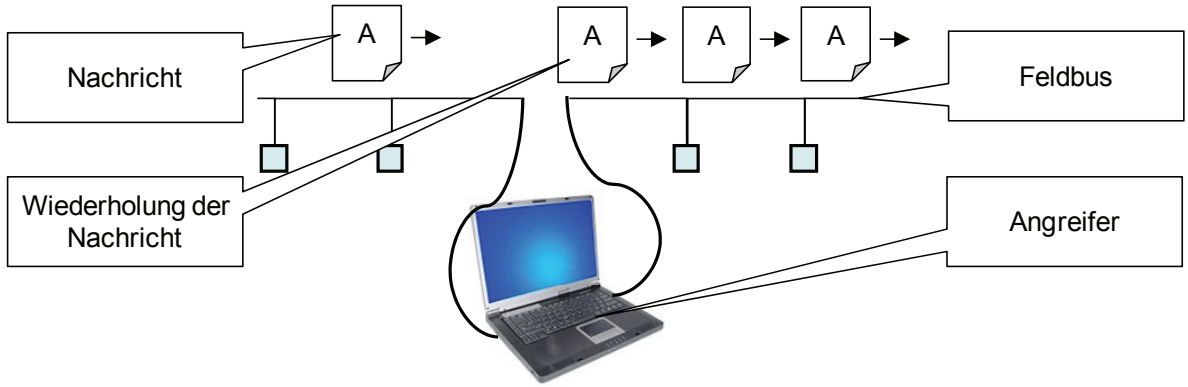

Abbildung 7.4: Wiederholung von Nachrichten durch Angreifer

Derartige Manipulationen können Funktionen des Automatisierungssystems zu „falschen“ Zeitpunkten auslösen bzw. zu „richtigen“ Zeitpunkten unterdrücken, wodurch ein Übergang des Automatisierungssystems in einen unerwünschten Fehlerzustand verursacht werden kann. Deshalb ist eine Funktionalität zum *Schutz der zeitlichen Integrität* von Nachrichten erforderlich.

7.1.3 Unautorisierte Manipulation von Funktionalität

Die Funktionalitäten von Feldgeräten werden entweder „festverdrahtet“ oder in Form von Software realisiert (vgl. Abschnitt 2.3.1). Unautorisierte funktionale Veränderungen einer korrekten Software bzw. einer korrekten „festverdrahteten“ Hardware führen dazu, dass das Feldgerät seine Anforderungsspezifikation nicht mehr erfüllt. Manipulationen der Funktionalität können daher neben einer Veränderung der Hardware auch mittels Veränderungen der Software erreicht werden. Veränderungen von Hardware erfolgen stets durch das Einwirken physischer Kräfte auf die Hardware, Veränderungen von Software erfordern dagegen eine informationstechnische Interaktion mit dem Feldgerät, das die Software ausführt.

Die Feldgerätesoftware ist in einem Speicher abgelegt, der diese auch ohne ständige Spannungsversorgung persistent speichert (ROM) und aus dem sie zur Ausführung durch den Prozessorkern geladen wird. Bei modernen Mikrocontrollern ist der Programmspeicher oftmals in Form mehrfach beschreibbarer Flash-ROM-Bausteine ausgeführt [Soff03]. Diese Eigenschaft kann von Angreifern ausgenützt werden, um die Feldgerätefunktionalität durch Überschreiben der vorhandenen Software (oder Teilen davon) zu manipulieren.

[9] entspricht unendlich langer Verzögerung

Legitime Veränderungen von Feldgerätesoftware werden üblicherweise im Rahmen von Wartungstätigkeiten durchgeführt, beispielsweise bei der Nachbesserung von Funktionalitäten oder der Anpassung von Funktionalitäten an geänderte Anforderungen oder Randbedingungen [LaGö99a]. Dazu werden zwei Vorgehensweisen herangezogen. Zum einen sind Software-Updates in modernen Feldgeräten oftmals vorgesehen, weshalb in solchen Feldgeräten Funktionen integriert sind, die Updates über Feldbusse empfangen und in den Programmspeicher ablegen („flashen“) können [ASAM08].

Zum anderen kann der Programmspeicher über die Programmierschnittstelle des Mikrocontrollers beschrieben werden. Da die Programmspeicherung über die Programmierschnittstelle in Hardware realisiert ist („festverdrahtet“), kann dieses Verfahren auch dann zum Einsatz kommen, wenn keine Update-Funktionalität vorhanden ist oder Programmteile aktualisiert werden sollen, die durch die Update-Funktionen nicht verändert werden können. Dazu ist ein physischer Zugriff auf das Feldgerät erforderlich. Beide Vorgehensweisen können auch durch Angreifer, welche über physischen Zugriff auf die Feldebene verfügen, zur Manipulation der Feldgerätefunktionalität herangezogen werden (Abbildung 7.5).

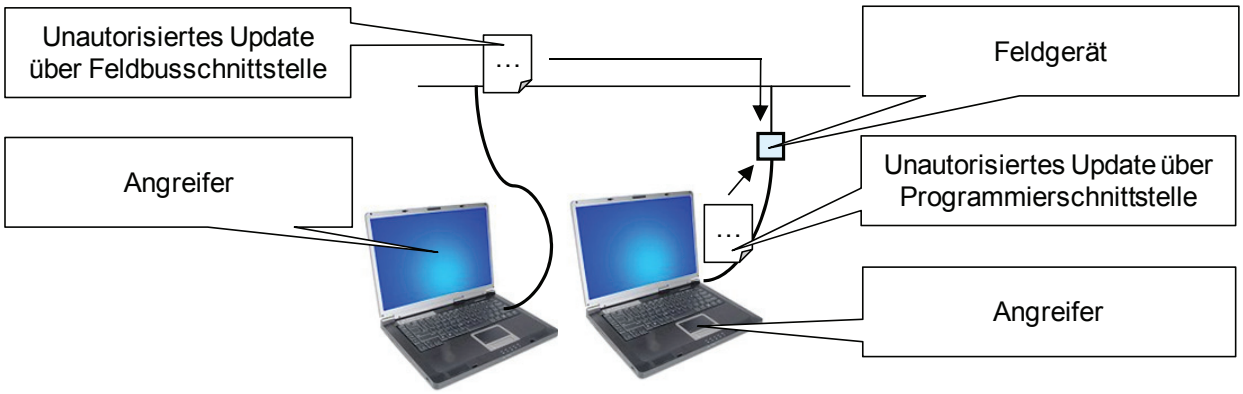

Abbildung 7.5: Unautorisierte Veränderung der Funktionalität über Feldbusschnittstelle (links) und über Programmierschnittstelle (rechts)

Da jede unautorisierte Veränderung Fehler verursachen kann, ist es unerheblich, in welchen Softwaremodulen der Feldgerätesoftware Manipulationen durchgeführt werden. Daher wird im Folgenden nicht unterschieden, welcher Teil der Software eines Feldgeräts manipuliert wurde, sondern es wird nur betrachtet, ob die Software eines Feldgeräts manipuliert wurde oder nicht. Um dies erkennen zu können, wird eine Funktionalität zum *Schutz der Feldgerätesoftware* benötigt.

7.2 Reaktion auf Angriffe

Der Schutz der IT-Sicherheit auf der Feldebene soll unerwünschte Auswirkungen von Angriffen verhindern. Daher müssen im Falle eines Angriffs geeignete Reaktionen eingeleitet werden. Die Erkennung des Angriffsfalls wird durch die in Abschnitt 7.1 beschriebenen Schutzfunktionalitäten durchgeführt.

Manipulierte Nachrichten sollen grundsätzlich nicht von der Feldgerätefunktionalität verarbeitet werden, da dies entsprechende Fehlerzustände nach sich ziehen kann. Daher wird eine Funktionalität benötigt, welche eine Filterung von manipulierten Nachrichten durchführt (Funktionsprinzip von Firewall-Systemen). Als Filterkriterium wird das Überprüfungsergebnis herangezogen, das durch die Funktionalitäten zur Erkennung von Manipulationen von Nachrichten ermittelt wurde. Ist eine Nachricht nicht manipuliert, so wird sie an die Feldgerätefunktionalität weitergeleitet. Manipulierte Nachrichten werden nicht weitergeleitet.

Über diese Filterung hinaus ist es gegebenenfalls erforderlich, weitere Gegenmaßnahmen einzuleiten. Gegenmaßnahmen sind ebenfalls einzuleiten, wenn erkannt wird, dass eine Feldgerätefunktionalität selbst manipuliert wurde. Abbildung 7.6 stellt die Daten- und Kontrollflüsse dar, welche bei der Reaktion auf Angriffe relevant sind.

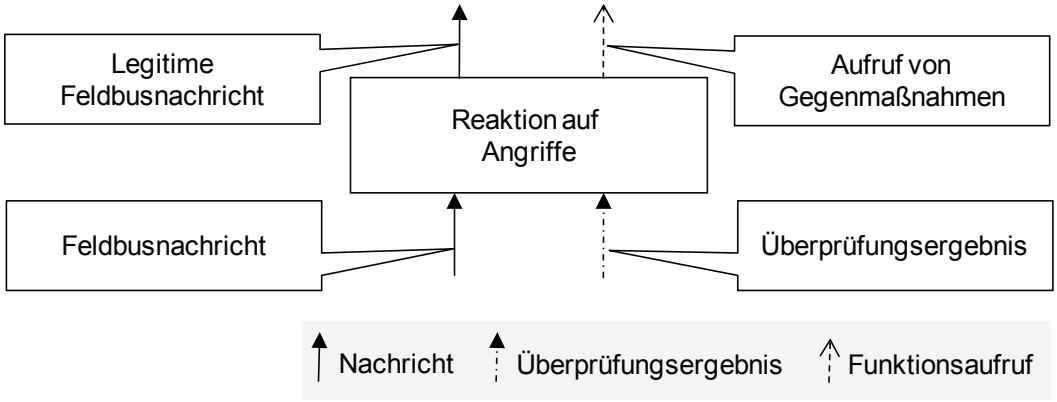

Abbildung 7.6: Zur Reaktion auf Angriffe erforderliche Daten- und Kontrollflüsse

Die Gegenmaßnahmen zielen auf die Vermeidung von unerwünschten Fehlerzuständen. Wie dies zu erreichen ist, hängt von dem konkreten Automatisierungssystem ab, beispielsweise durch die Benachrichtigung von Bedienpersonal, die Abschaltung eines Teilsystems oder die Aktivierung von Back-up-Systemen. Dazu muss die Reaktionsfunktionalität Funktionen der Feldgerätefunktionalität aufzurufen, welche je nach Automatisierungssystem unterschiedlich ausgeführt sind. Abbildung 7.7 stellt die Anbindung der Schutzfunktionalitäten an die Feldgerätesoftware dar.

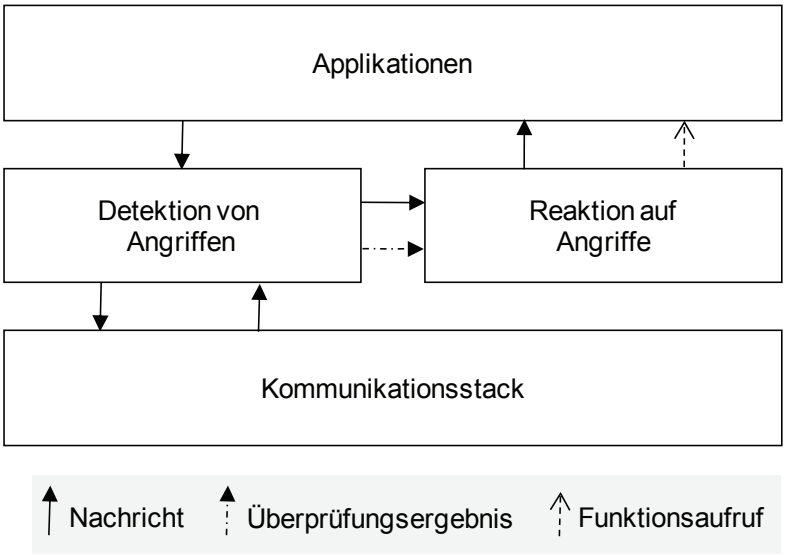

Abbildung 7.7: Bausteine und Beziehungen der Feldgerätesoftware mit integriertem Schutz

7.3 Erbringung der Schutzfunktionalitäten mit Schutzmechanismen

In den vorhergehenden Abschnitten wurde dargestellt, welche Schutzfunktionalitäten zur Absicherung der IT-Sicherheit auf der Feldebene erforderlich sind. Schutzfunktionalitäten werden grundsätzlich mit Schutzmechanismen erbracht (vgl. Abschnitt 3.3.1). Da jede Schutzfunktionalität im Allgemeinen mit unterschiedlichen Schutzmechanismen umgesetzt werden kann, wird im Rahmen dieses Abschnitts untersucht, welche Schutzmechanismen zur Umsetzung auf der Feldebene am besten geeignet sind. Dazu werden die Schutzmechanismen anhand ihrer Wirkungsweisen in Klassen eingeteilt, die Eigenschaften dieser Klassen diskutiert und schließlich jeweils eine Schutzmechanismusklasse zur Umsetzung jeder Schutzfunktionalität auf der Feldebene ausgewählt.

Die im Rahmen dieser Arbeit relevanten Eigenschaften von Schutzmechanismen betreffen die Stärke sowie den Ressourcenverbrauch der Schutzmechanismen. Da die Schutzmechanismen in Software realisiert werden, sind die Ressourcen, welche verbraucht werden, Speicherplatz und Rechenzeit von Feldgeräten sowie Bandbreite bei der Feldbuskommunikation. Da Automatisierungssysteme im Allgemeinen den Echtzeitanforderungen des geführten technischen Prozesses gerecht werden müssen, ist ein deterministischer Verbrauch der Ressource Zeit durch die Schutzmechanismen zu beachten.

7.3.1 Vertraulichkeit

Die informationstechnische Erbringung der Schutzfunktionalität *Vertraulichkeit* wird stets durch Schutzmechanismen erbracht, welche kryptografische Ver- bzw. Entschlüsselungen durchführen [BSW01]. Eine Verschlüsselung ist eine bijektive Transformation eines Klartexts k in einen

Schlüsseltext *s* mithilfe eines mit einem Schlüssel *v* zu parametrierenden Algorithmus *V*. Kryptografische Verschlüsselungen zeichnen sich dadurch aus, dass eine Entschlüsselung, d. h. die Transformation *E* des Schlüsseltexts *s* in den Klartext *k*, ohne Kenntnis eines Schlüssels *e* nicht möglich oder zumindest „rechnerisch nicht machbar“ (computationally infeasible) ist [MvOV96]. Abbildung 7.8 stellt das Prinzip der Ver- und Entschlüsselung zur Wahrung der Vertraulichkeit von Kommunikationsvorgängen dar.

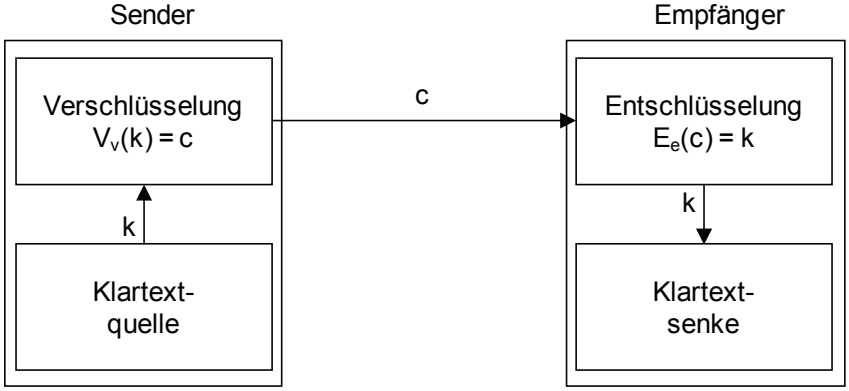

Abbildung 7.8: Prinzip der verschlüsselten Kommunikation

Es lassen sich zwei Klassen von Schutzmechanismen zur Erbringung des Vertraulichkeitsschutzes unterscheiden, welche in der Funktionsweise differieren: symmetrische sowie asymmetrische Verschlüsselungsverfahren. Symmetrische Verschlüsselungsverfahren gelten als effizienter als asymmetrische, erfordern allerdings die Kenntnis eines einzigen geheimen Schlüssels sowohl durch Sender als auch Empfänger einer verschlüsselten Nachricht. Daher muss dieser Schlüssel vor der initialen Aufnahme der verschlüsselten Kommunikation über einen sicheren Kanal zwischen den Kommunikationspartnern übermittelt werden. Symmetrische Verschlüsselungsverfahren lassen sich weiter in blockorientierte und stromorientierte Verfahren unterteilen. Stromorientierte Verfahren sind zumeist etwas effizienter als blockorientierte, weisen aber Nachteile bei der mehrfachen Verwendung eines Schlüssels auf.

Asymmetrische Verschlüsselungsverfahren definieren dagegen ein Schlüsselpaar, bestehend aus einem privaten und einem öffentlichen Schlüssel. Mithilfe des öffentlichen Schlüssels kann nur verschlüsselt, nicht aber entschlüsselt werden, sodass der öffentliche Schlüssel über unsichere Kanäle übermittelt werden kann.

Aufgrund der engen Kopplung der Systemelemente der Feldebene sind die Kommunikationsbeziehungen der Feldgeräte bereits zur Entwicklungszeit des Automatisierungssystems bekannt. Somit ist es einfach möglich, Schlüssel auf sicherem Weg, z. B. im Rahmen der Inbetriebnahme, auf die einzelnen Feldgeräte zu verteilen.

Die Schutzstärke von Verschlüsselungsalgorithmen hängt von dem konkreten Algorithmus sowie von der Länge des verwendeten Schlüssels ab. Da symmetrische Verschlüsselungsalgorithmen bei gleicher Schutzstärke wesentlich effizienter sind als asymmetrische Verschlüsselungs-

algorithmen und die Verteilung der Schlüssel auf der Feldebene keinen Nachteil darstellt, bieten sich symmetrische Algorithmen zum Schutz der Vertraulichkeit auf der Feldebene an. Da die mehrfache Nutzung eines Schlüssels bei blockorientierten Algorithmen einfacher als bei stromorientierten Algorithmen möglich ist, sind blockorientierte Algorithmen besser zum Einsatz auf der Feldebene geeignet.

7.3.2 Autorisierung

Zum Schutz der Autorisierung ist zunächst eine eindeutige Identifizierung (Authentifizierung) des Absenders einer Nachricht erforderlich. Daraufhin ist zu prüfen, ob der Absender auch berechtigt ist, die durch die Nachricht ausgelöste Operation durchzuführen. Die Authentifizierung erfordert die Angabe und den Nachweis der Identität eines Nachrichtensenders, damit es Angreifern nicht möglich ist, Manipulationen mit selbst erzeugten Nachrichten durchzuführen, welche dem Empfänger einen legitimen Absender vortäuschen. Zum Nachweis der Identität wird daher ein Merkmal benötigt, anhand dessen Feldgeräte ihre Kommunikationspartner eindeutig identifizieren können. Zudem muss es möglich sein, manipulationssicher festzustellen, dass es sich bei dem durch das Identitätsmerkmal angegebenen Kommunikationspartner tatsächlich um diesen handelt und nicht um einen Angreifer, der dessen Identität nur vortäuscht. Schutzmechanismen zum Schutz der Autorisierung definieren daher Verfahren, wie Identitäten überprüft werden können.

Dabei lassen sich prinzipiell drei Klassen solcher Schutzmechanismen unterscheiden. Eine Klasse dieser Schutzmechanismen beweist die Identität durch Übergabe des Identifikationsmerkmals (z. B. eines Passworts bei der Benutzeranmeldung an einen PC), wodurch die prüfende Partei die Identität des Kommunikationspartners durch Vergleich feststellen kann. Die Verwendung derartiger Schutzmechanismen ist dann möglich, wenn davon ausgegangen werden kann, dass es Angreifern nicht möglich ist, durch Abhören in den Besitz des Identifikationsmerkmals zu kommen. Daher sind solche Schutzmechanismen für die Feldebene nicht geeignet.

Bei der zweiten Klasse wird das Identifizierungsmerkmal selbst nicht übergeben, sondern nur seine Kenntnis durch ein Frage-Antwort-Verfahren nachgewiesen (Challenge-Response-Verfahren). Dafür sind mehrere Nachrichten erforderlich, sodass sich diese Verfahren zum Schutz von verbindungsorientierter Kommunikation eignen, nicht aber für die verbindungslose, nachrichtenorientierte Kommunikation der Feldebene.

Die dritte Schutzmechanismusklasse verwendet digitale Signaturen, welche auf kryptografischen Verfahren beruhen. Mit digitalen Signaturen kann die Identität eines Nachrichtensenders auch dann ohne Kompromittierung des Identifikationsmerkmals nachgewiesen werden, wenn Angreifer in der Lage sind, die Kommunikation abzuhören. Daher wird die Funktionsweise digitaler Signaturen für die Erbringung der Autorisierung auf der Feldebene ausgewählt, es muss

jedoch berücksichtigt werden, dass Algorithmen zur Berechnung von digitalen Signaturen komplex sind und somit einen hohen Ressourcenverbrauch aufweisen.

7.3.3 Inhaltliche Integrität

Die Erbringung eines informationstechnischen Schutzes zur Gewährleistung inhaltlicher Integrität muss in der Lage sein, unautorisierte Veränderungen von Nachrichten zu erkennen. Dazu existieren zahlreiche unterschiedliche Schutzmechanismen [BMB+05]. Diese Mechanismen führen eine Abbildung des Nachrichteninhalts auf ein Datum mit zumeist fester Länge durch. Dieses Datum wird ebenfalls an den Empfänger einer geschützten Nachricht übermittelt. Dieser ist somit in der Lage, das empfangene Datum mit einem Datum zu vergleichen, die er selbst aus dem empfangenen Nachrichteninhalt berechnen kann. Stimmen beide überein, wird daraus auf eine unveränderte Nachricht geschlossen.

Derartige Schutzmechanismen lassen sich in drei Klassen unterteilen. Lineare Fehlererkennungsverfahren, wie Prüfsummen, wurden bereits in Abschnitt 5.2.2 diskutiert und aufgrund ihrer einfachen Manipulierbarkeit als nicht geeignet für die betrachtete Problemstellung befunden.

Die zweite Klasse von Schutzmechanismen sind kryptografische Hashfunktionen. Kryptografische Hashfunktionen sind mit Prüfsummen vergleichbar, zeichnen sich aber durch zwei Eigenschaften aus, welche bei Prüfsummen nicht gegeben sind: Aufgrund ihrer *Urbildresistenz* ist die Konstruktion von Nachrichten schwierig, welche einem gegebenen Hashwert entsprechen, durch ihre *Kollisionsresistenz* ergeben unterschiedliche Nachrichten mit hoher Wahrscheinlichkeit unterschiedliche Hashwerte.

Diese Schutzmechanismusklasse wird vorrangig in der Standard-IT eingesetzt, wenn die zu schützende Information und der Hashwert von unterschiedlichen Absendern stammen und über unterschiedliche Wege transportiert werden (z. B. beim Download von Software über das Internet). Die Sicherheit dieses Vorgehens basiert auf der Annahme, dass ein Angreifer nicht gleichzeitig die herunterzuladende Software als auch den von einer anderen Quelle stammenden Hashwert manipulieren kann. Auf der Feldebene existieren zumeist keine unterschiedlichen Routen für Nachrichten und es muss angenommen werden, dass ein Angreifer aufgrund des physischen Zugriffs in der Lage ist, die Nachricht und den Hashwert zu manipulieren. Zudem ist der Ressourcenbedarf kryptografischer Hashfunktionen wesentlich höher als der von linearen Fehlererkennungsverfahren.

Als dritte Klasse von Schutzmechanismen zum Schutz der inhaltlichen Integrität verwenden Keyed-Hash-Funktionen Algorithmen, welche Konzepte von kryptografischen Hashfunktionen und Verschlüsselung kombinieren, um Manipulationen des Hashwerts erkennen zu können. Keyed-Hash-Funktionen verschlüsseln dazu den Hashwert, wodurch ein Angreifer ohne Kenntnis des verwendeten Schlüssels nicht in der Lage ist, einen validen Hashwert für eine manipu-

lierte Nachricht zu erzeugen. Daher ist diese Klasse von Schutzmechanismen prinzipiell zum Schutz der inhaltlichen Integrität von Informationen auf der Feldebene geeignet. Aufgrund der Kombination von Verschlüsselung und kryptografischen Hashfunktionen ist der Ressourcenbedarf von Keyed-Hash-Funktionen jedoch der höchste aller Schutzmechanismusklassen zum Schutz der inhaltlichen Integrität.

7.3.4 Zeitliche Integrität

Zur Erbringung eines Schutzes der zeitlichen Eigenschaften von Nachrichten müssen Manipulationen erkannt werden können, bei denen Nachrichten selbst nicht verändert werden, sondern verzögert weitergeleitet, wiederholt oder gelöscht werden. Es existieren dabei zwei Klassen von Schutzmechanismen zur Erbringung eines Schutzes der zeitlichen Integrität [BMB+05]. Zeitstempelbasierte Schutzmechanismen versehen Nachrichten mit fortlaufenden Nummern bzw. Angaben über Sendezeit und Gültigkeit und ermöglichen so bei Empfang der Nachricht die Detektion von Vertauschungen, Wiederholungen und Verzögerungen. Wie bei der Verwendung von Identifikationsmerkmalen sind diese Mechanismen der Manipulation durch Angreifer mit physischem Zugriff auf das Übertragungsmedium ausgesetzt.

Um Verzögerungen bzw. Löschungen von Nachrichten detektieren zu können, ohne dass die entsprechende Nachricht empfangen werden konnte, existieren Heartbeat-Mechanismen, welche kontinuierlich vom Sender zum Empfänger einer Nachricht gesendet werden, die vor Verzögerung geschützt werden muss. Diese Nachrichten informieren den Empfänger, dass die zu schützende Nachricht selbst noch nicht gesendet wurde. Bleiben die Heartbeat-Nachrichten und die dadurch abzusichernde Nachricht aus, kann der Empfänger folgern, dass gesendete Nachrichten während der Übertragung gelöscht wurden.

Aufgrund der Bedrohung des Wiederholens, Verzögerns bzw. Löschens von Nachrichten auf der Feldebene sind die beschriebenen Schutzmechanismen zur Absicherung der zeitlichen Integrität erforderlich. Zeitstempelbasierte Mechanismen eignen sich für Anwendungen, bei denen die Ausführung einer Operation, die von einer nicht rechtzeitig eintreffenden Nachricht ausgelöst wird, schädliche Auswirkungen hervorruft. Heartbeat-Mechanismen sind dagegen erforderlich, wenn das Ausbleiben einer Nachricht bis zu einem bestimmten Zeitpunkt schädliche Auswirkungen hat. Sowohl Heartbeat-Informationen als auch Zeitstempel müssen gegen eine Manipulation durch Angreifer abgesichert sein, was einen zusätzlichen Ressourcenaufwand erfordert.

7.3.5 Schutz der Feldgerätesoftware

Zur Veränderung der Feldgerätesoftware müssen Informationen in den Programmspeicher des Feldgeräts geschrieben werden. Feldgeräte besitzen drei Kanäle, über die sie Informationen aufnehmen können. Sie messen physikalische Größen aus dem technischen Prozess und empfangen

Daten über die Feldbus- sowie die Programmierschnittstelle. Abbildung 7.9 stellt die Kanäle dar, über die Informationen in den Rechner des Feldgeräts gelangen.

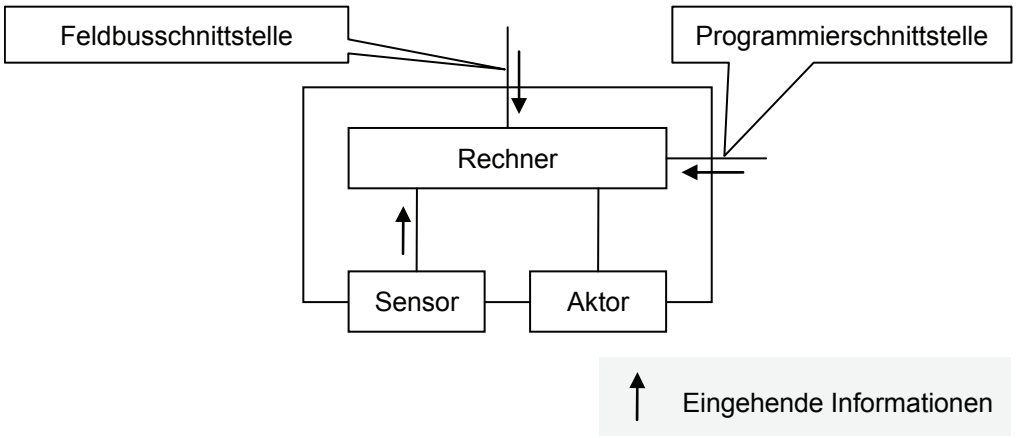

Abbildung 7.9: Mögliche Angriffsvektoren zur Manipulation des Programmspeichers

Informationen, welche von Sensoren gemessen werden, werden im Allgemeinen in Form einer elektrischen Größe vom Rechner des Feldgeräts aufgenommen und dort weiterverarbeitet. Im Speicher des Rechners stellen diese Informationen Daten dar, welche keine Veränderung des Programmcodes bewirken können. Es ist aber möglich, über die verbleibenden Schnittstellen Manipulationen der Software vorzunehmen.

Über einen Feldbus durchgeführte Veränderungen an der Funktionalität von Feldgeräten, wie z. B. das Update eines Kfz-Steuergeräts über den CAN-Bus, erfordern prinzipbedingt eine Update- oder Flashfunktion, welche die über den Feldbus übertragenen Programmänderungen in den Programmspeicher des Feldgeräts schreibt. Ein Angreifer, der die Gerätefunktionalität über einen Feldbus verändern möchte, muss entsprechende Nachrichten an das angegriffene Feldgerät senden. Gegen derartige Manipulationen kann der in Abschnitt 7.3.2 beschriebene Schutz der Autorisierung eingesetzt werden, da ein Angreifer per Definition nicht zur Nutzung der Updatefunktionalität berechtigt ist. Dadurch wird eine Verarbeitung der Nachrichten durch die Update-/Flashsoftware unterbunden.

Es muss beim Schutz der Feldgerätefunktionalität jedoch berücksichtigt werden, dass Angreifer auch über physischen Zugriff auf die Feldgeräte verfügen können. Angreifer sind daher in der Lage, mittels Ankopplung an die Programmierschnittstelle von Feldgeräten Softwareprogramme zu ändern, indem auf Hardwareebene auf den Programmspeicher zugegriffen und mit manipulierten Programmen überschrieben wird. Zur Erkennung derartiger Veränderungen sollen die Funktionsprinzipien von Anomalieerkennungsprogrammen herangezogen werden. Dabei lassen sich zwei Klassen von Schutzmechanismen unterscheiden.

Die Klasse der regelbasierten Schutzmechanismen identifiziert unerwünschte Software bzw. Veränderungen an Software anhand fester Regeln (z. B. Erkennung eines Virus anhand des invarianten Virus-Programmcodes). Heuristische Schutzmechanismen verwenden stattdessen

Vorgehensweisen, die auf Hypothesen und Vermutungen aufbauen [MiFo04], um auch solche Veränderungen aufspüren zu können, für die keine feste Regel festgelegt ist (z. B. Erkennung eines Virus anhand bestimmter Verhaltensweisen).

Im Gegensatz zu Standard-IT-Rechnern, bei denen sich die Zusammensetzung der installierten Software häufig ändert, wird die Software von Feldgeräten im Allgemeinen nicht im regulären Betrieb, sondern nur im Fall der Wartung oder bei Modifikationen an dem Automatisierungssystem verändert. Im Rahmen solcher Tätigkeiten ist es einfach möglich, auch einen eingesetzten Schutzmechanismus über das Update zu informieren. Somit kann grundsätzlich jede Änderung der Software, über die der Schutzmechanismus nicht informiert ist, als Angriff interpretiert werden.

Dadurch ist es bei der Absicherung der Funktionalität von Feldgeräten möglich, Schutzmechanismen zu verwenden, die anhand einer einzigen Regel entscheiden, ob eine Manipulation vorliegt: Jede Abweichung von dem bekannten Softwarestand ist als Angriff zu betrachten. Zur Erkennung von Änderungen der Software wird als Unterscheidungsmerkmal ein „Fingerabdruck“ der Software erzeugt, der bei legitimen Änderungen entsprechend aktualisiert werden muss.

Bei Ankopplung des Angreifers an die Programmierschnittstelle kann jedoch nicht nur die Implementierung der Feldgerätefunktionalität, sondern auch die Implementierung der Schutzfunktionalität zum Schutz der Feldgerätefunktionalität überschrieben und somit unwirksam gemacht werden. Ein Angriff auf diese Schutzfunktionalität ist in Abbildung 7.10 dargestellt.

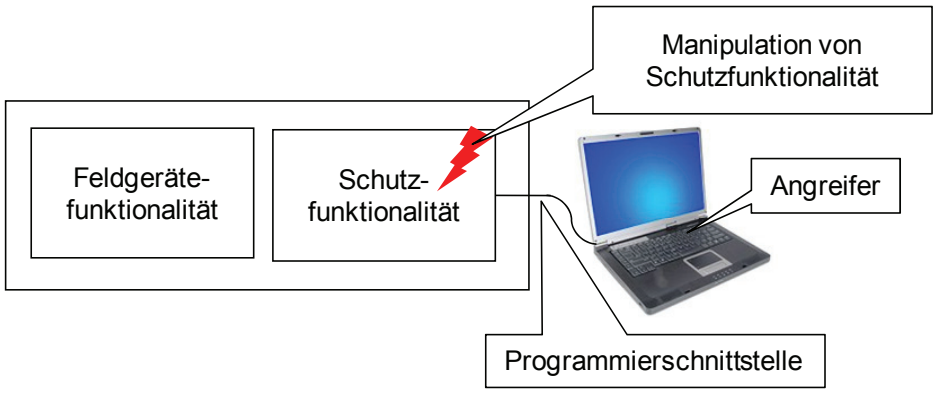

Abbildung 7.10: Manipulation von Schutzfunktionalität auf einem Feldgerät

Aus diesem Grund muss die Funktionalität zur Absicherung der Feldgerätesoftware so erbracht werden, dass trotz Manipulationen am Programmcode eine Entdeckung und eine Reaktion möglich sind. Befinden sich der Schutzmechanismus, der den Schutz der Funktionalität erbringt, und die zu schützende Software auf demselben Gerät, kann dies nicht gewährleistet werden, da ein Angreifer auch den Schutzmechanismus selbst manipulieren kann. Daher ist es erforderlich, den Schutz der Gerätefunktionalität über Gerätegrenzen hinweg zu verteilen. Schutzmechanismen

zur Absicherung der Funktionalität von Feldgeräten müssen daher eine verteilte Überwachung erbringen. Das Prinzip dieses Schutzmechanismus ist in Abbildung 7.11 dargestellt.

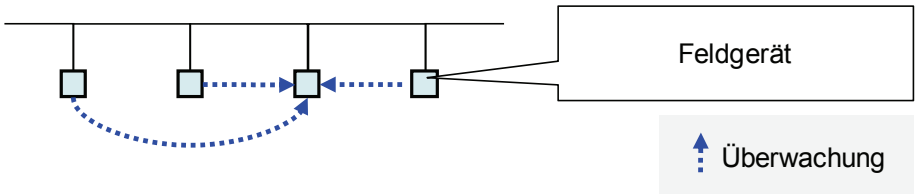

Abbildung 7.11: Prinzip der verteilten Überwachung der Feldgerätesoftware

Daraus folgt, dass sich der Schutzmechanismus zur Überwachung der Feldgerätesoftware in zwei Teile aufspaltet. Der eine Teil wird auf dem Feldgerät ausgeführt, welches ein oder mehrere andere Feldgeräte überwacht. Dieser Teil des Schutzmechanismus verfügt über Unterscheidungsmerkmale („Fingerabdrücke“) der unveränderten, zu überwachenden Software. Der andere Teil des Schutzmechanismus wird auf dem überwachten Feldgerät ausgeführt und erzeugt auf Anfrage des überwachenden Feldgeräts einen „Fingerabdruck“ der momentan auf dem Feldgerät vorhandenen Software und sendet diesen an das anfragende Feldgerät. Dieses kann durch Vergleich des gespeicherten Unterscheidungsmerkmals mit dem übermittelten Unterscheidungsmerkmal Veränderungen der Software erkennen.

Um Manipulationen zu erschweren, bei denen der auf dem überwachten Feldgerät ausgeführte Teil des Schutzmechanismus so verändert wird, dass er nicht das aus dem aktuellen Stand der Software erzeugte Unterscheidungsmerkmal übermittelt, sondern ein zu einem früheren Zeitpunkt erzeugtes, wird bei jeder Anfrage ein Zahlenwert übermittelt, der in die Erzeugung des Unterscheidungsmerkmals miteinbezogen wird (Challenge-Response-Verfahren).

7.4 Reduktion des Ressourcenbedarfs des Schutzes

In den vorangegangenen Kapiteln wurden Klassen von Schutzmechanismen identifiziert, welche aufgrund ihrer funktionalen Eigenschaften zur Erbringung eines wirksamen Schutzes der IT-Sicherheit auf der Feldebene geeignet sind. Dabei macht es die Bedrohung der Feldebene durch intentionale Angriffe erforderlich, Schutzmechanismen zur Detektion von Angriffen auszuwählen, welche eine hohe Schutzstärke und daher einen hohen Ressourcenbedarf aufweisen. Die Berechnung einer digitalen Signatur benötigt auf einem Renesas-M16C-Mikrocontroller eine Dauer von ca. 2,5 min [Clau03] [Hell04]. Derartige Schutzmechanismen sind in der vorliegenden Ausführung für die Randbedingungen der Feldebene nicht geeignet.

Um einen wirksamen Schutz der IT-Sicherheit auf der Feldebene von neu zu entwickelnden Automatisierungssystemen ohne oder nur mit minimalen Zusatzkosten bzw. bei bestehenden Automatisierungssystemen überhaupt[10] einsetzen zu können, muss der Ressourcenbedarf der

[10] D. h. ohne die Integration leistungsfähigerer Systemelemente

ausgewählten Schutzmechanismen reduziert werden. Dazu werden zunächst unterschiedliche Möglichkeiten dargestellt, wie der erforderliche Speicherplatz, die benötigte Rechenzeit und die erforderliche Kommunikationsbandbreite verringert werden können. Diese Möglichkeiten werden dann unter den gegebenen Randbedingungen der Feldebene und der IT-Sicherheit bewertet.

7.4.1 Ansätze zur Reduktion des Ressourcenbedarfs

Grundsätzlich existieren drei Ansätze zur Einsparung von Ressourcen in verteilten Systemen, welche entweder isoliert oder in Kombination angewendet werden können [Trau07]. Bei einer *örtlichen Verteilung* von Funktionalität wird die erforderliche Gesamtfunktionalität in Teilfunktionalitäten zerlegt und auf unterschiedliche Rechner verteilt. In diesem Fall trägt jeder Einzelrechner nur einen Bruchteil der Speicher- und Rechenleistungslast. Voraussetzung für eine derartige Verteilung sind eine Zerlegbarkeit der Funktionalität und eine Kommunikationsverbindung zwischen den Einzelrechnern. Zudem muss der zusätzliche Bedarf an Kommunikationsbandbreite berücksichtigt werden, der aufgrund der Kommunikationsvorgänge zwischen den Teilfunktionalitäten entsteht.

Bei *temporärer Funktionsintegration* werden Funktionalitäten nicht statisch auf einem Rechner gespeichert, sondern durch Kommunikationsverbindungen übermittelt und nur für den Zeitraum, in dem sie benötigt werden, auf dem Zielrechner abgelegt und ausgeführt. Wird die Funktion nicht mehr benötigt, kann sie gelöscht oder durch andere Funktionen ersetzt werden. Dadurch lassen sich beliebig viele unterschiedliche Funktionalitäten nacheinander auf einem Rechner ausführen, wenn die zur Verfügung stehenden Ressourcen für die einzelnen Funktionalitäten ausreichend sind. Da die Dauern für die Übermittlung und Integration der zu integrierenden Funktionalitäten ebenfalls berücksichtigt werden müssen, können unterschiedliche temporär integrierte Funktionen nicht unmittelbar nacheinander ausgeführt werden.

Im Rahmen einer *ressourcenschonenden Entwicklung* werden unterschiedliche Maßnahmen angewendet, um Software so zu entwerfen und zu implementieren, dass die geforderte Funktionalität auf Systemen mit geringen Ressourcen ausgeführt werden kann. Im Rahmen des Entwurfs sind die folgenden Maßnahmen zu nennen [Cool03]:

- Der Aufbau von Programmen ausschließlich aus den minimal erforderlichen Programmelementen
- Die Auslegung des Programmcodes auf Mehrfachverwendung innerhalb der Applikation
- Die Definition schmaler Schnittstellen
- Die Optimierung von Algorithmen auf geringen Speicherplatz- oder Rechenzeitbedarf

Bei der Implementierung bietet sich die Verwendung effizienter Technologien an, z. B. hardwarenahe Programmiersprachen mit geringem Overhead. Die in einem konkreten Fall zu wäh-

lenden Maßnahmen sind dabei von den Eigenschaften des gegebenen Systems abhängig, sodass keine allgemeingültige Menge von Maßnahmen zur ressourcenschonenden Entwicklung existiert, welche in jedem Fall verwendet werden kann [Trau07].

7.4.2 Eignung der Reduktionsansätze für den Schutz der Feldebene

Wie in Abschnitt 6.2 dargestellt, müssen die Schutzfunktionalitäten auf den Feldgeräten ausgeführt werden, deren Kommunikation oder Funktionalität geschützt werden soll. Dabei werden Unterscheidungsmerkmale in die Feldbusnachrichten integriert (durch den Sender einer Feldbusnachricht) bzw. eine Überprüfung dieser Unterscheidungsmerkmale durchgeführt (durch den Empfänger der Feldbusnachricht) sowie die Software der Feldgeräte über Gerätegrenzen hinweg überwacht. Die Schutzfunktionalitäten sind also bereits a priori örtlich verteilt. Eine Zerlegung dieser Funktionalitäten, beispielsweise der Erzeugung von Unterscheidungsmerkmalen, würde Kommunikationsvorgänge zum Zusammenwirken der Teilfunktionen erfordern. Diese Kommunikationsvorgänge wären dann wiederum nicht gegen Angriffe auf die IT-Sicherheit geschützt, wodurch eine weitere Zerlegung und Verteilung des Schutzes, die über die durch den Lösungsansatz bedingte hinausgeht, nicht zur Reduktion des Ressourcenverbrauchs herangezogen werden kann.

Temporär in ein Automatisierungssystem integrierte Funktionalitäten müssen über Kommunikationsverbindungen übertragen werden. Da die Aufgabe von Kommunikationsschutzfunktionalitäten die Absicherung ebendieser Kommunikationsverbindungen ist, eignen sich diese Schutzfunktionalitäten nicht zur temporären Integration. Ein Angreifer wäre in diesem Fall in der Lage, die Kommunikationsschutzfunktionalitäten bei ihrer Übertragung zu manipulieren, was den Zweck ihrer Übertragung ad absurdum führen würde.

Der Schutz zur Absicherung der Feldgerätefunktionalität kann prinzipiell temporär integriert werden, wenn eine geschützte Übertragung gewährleistet ist. Eine temporäre Integration von Funktionalität kann jedoch nur dann verwendet werden, wenn die temporär integrierten Funktionalitäten nicht gleichzeitig benötigt werden. Zur Überwachung der Funktionalität muss jedoch sowohl der Schutzmechanismus als auch der überwachte Programmcode im Gerät vorhanden sein. Bei einer temporären Integration des Schutzmechanismus können nur die dauerhaft vorhandenen Teile der Feldgerätefunktionalität überwacht werden, nicht aber die temporär integrierten Teile, da diese niemals gleichzeitig mit dem Schutzmechanismus auf dem Feldgerät vorhanden sind.

Ein großes Potenzial zur Ressourceneinsparung besteht im Rahmen einer ressourcenschonenden Entwicklung, indem das zu entwickelnde Softwaresystem so entworfen wird, dass es die für ein spezifisches Automatisierungssystem geltenden Randbedingungen optimal erfüllt. Im Kontext eines ressourcenschonenden Entwurfs muss daher die Frage gestellt werden, ob zur Absicherung der Feldebene stets alle beschriebenen Schutzfunktionalitäten erforderlich sind. Ist dies nicht der

Fall, können die einzelnen Schutzfunktionalitäten so konzipiert werden, dass sie abhängig von der gegebenen Gefährdungslage ausgewählt und kombiniert werden können. Zur effizienten Implementierung des Schutzes erscheint die Verwendung einer hardwarenahen Programmiersprache sinnvoll, mit welcher die Ressourcen der Feldgeräte optimal genutzt werden können.

7.4.3 Ressourcenschonender Entwurf der Schutzfunktionalitäten

7.4.3.1 Variabilität der Schutzfunktionalität

Die Lage der Gefährdung durch Angriffe auf die IT-Sicherheit ist nicht bei jedem Automatisierungssystem identisch. Eine Schutzfunktionalität zur Absicherung eines Systemelements der Feldebene gegen eine bestimmte Art von Angriff ist immer dann erforderlich, wenn anzunehmen ist, dass eine Bedrohung dieses Systemelements durch diese Angriffsart besteht und Schwachstellen des Systemelements existieren, die ein Angreifer ausnutzen kann. Ist eine solche Gefährdung nicht gegeben, ist die betreffende Schutzfunktionalität nicht erforderlich. Zusätzlich zu diesem binären Kriterium (Gefährdung existiert / Gefährdung existiert nicht) können auch Risikobetrachtungen durchgeführt werden, in welche die Auswirkungen eines erfolgreichen Angriffs einfließen. Somit muss die Entscheidung, welche Schutzfunktionalitäten erforderlich sind, für jedes Systemelement der Feldebene individuell getroffen werden. Kapitel 8 stellt eine Vorgehensweise zur Bestimmung der jeweils erforderlichen Schutzfunktionalitäten vor, an dieser Stelle sei festgehalten, dass nicht immer alle Schutzfunktionalitäten auf jedem Feldgerät benötigt werden.

7.4.3.2 Modularisierung der Schutzfunktionalitäten

Werden nicht alle Schutzfunktionalitäten auf jedem Feldgerät benötigt, dürfen die Schutzfunktionalitäten nicht in einer monolithischen Form vorliegen. Stattdessen muss der Schutz der Feldebene so aufgebaut werden, dass beliebige Kombinationen von Schutzfunktionalitäten möglich sind. Dadurch kann der im jeweiligen Fall erforderliche Schutz erbracht werden, die Ressourcen zur Realisierung der nicht benötigten Schutzfunktionalitäten werden eingespart.

Um eine solche Kombinierbarkeit mit geringem Aufwand zu ermöglichen, muss der Schutz *modular* aufgebaut werden, d. h. aus funktional abgeschlossenen Bausteinen konstruiert, welche die Operationen und Daten zur Erfüllung einer abgeschlossenen Aufgabe zusammenfassen. Um unerwünschte Seiteneffekte durch inkonsistente Veränderungen von modulinternen Daten zu vermeiden, kommunizieren Module ausschließlich über eindeutig definierte Schnittstellen mit der Außenwelt [Göhn07]. Abbildung 7.12 stellt die Bestandteile von Modulen dar, bestehend aus Operationen und Daten sowie den Schnittstellen, die das Modul anderen Modulen zur Verfügung stellt (Exportschnittstelle) bzw. die das Modul von anderen Modulen nutzt (Importschnittstelle).

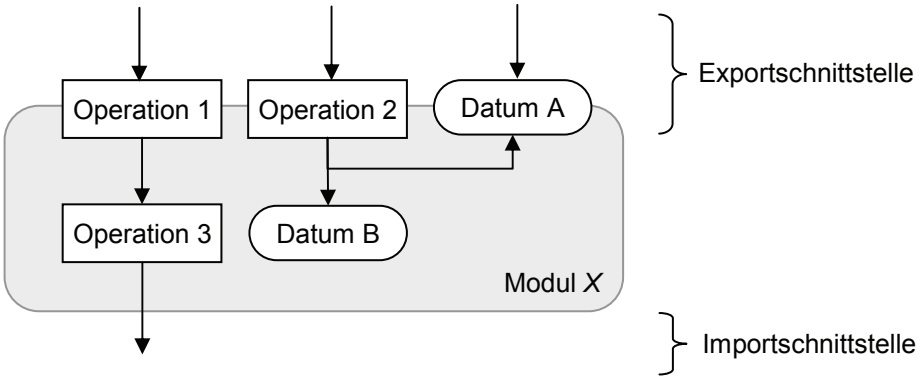

Abbildung 7.12: Bestandteile von Modulen

Damit die unterschiedlichen Schutzfunktionalitäten einzeln ausgewählt und zum Schutz der IT-Sicherheit auf der Feldebene eingesetzt werden können, wird jede Schutzfunktionalität in Form eines Moduls konstruiert. Diese Module können dann unabhängig voneinander eingesetzt und verknüpft werden. Die Schnittstellen der Module müssen einerseits die abzusichernden Artefakte (Nachrichten oder Programmcode) entgegennehmen können, um Unterscheidungsmerkmale zu integrieren bzw. zu überprüfen. Zudem müssen auch die Ergebnisse der Überprüfung über die Modulschnittstellen übermittelt werden. Die Erbringung der Schutzfunktionalität erfolgt durch einen Schutzmechanismus, welcher durch die modulinternen Daten und Operationen realisiert wird. Abbildung 7.13 zeigt beispielhaft ein Modul zum Schutz der Vertraulichkeit.

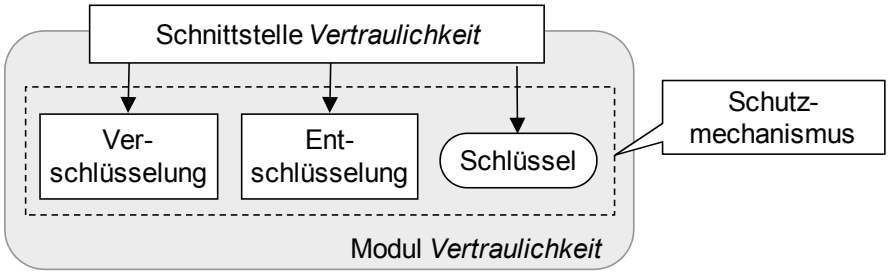

Abbildung 7.13: Modul zum Schutz der Vertraulichkeit

7.4.4 Ressourcenschonender Entwurf der Schutzmechanismen

7.4.4.1 Entkopplung von Schutzfunktionalitäten und Schutzmechanismen

Schutzfunktionalitäten werden durch Schutzmechanismen erbracht, d. h. konkrete Vorschriften, wie z. B. kryptografische Algorithmen. In Abschnitt 7.3 wurde für jede Schutzfunktionalität eine Klasse von Schutzmechanismen ausgewählt, die für die Erbringung von Schutz auf der Feldebene geeignet ist. Innerhalb dieser Klassen existieren im Allgemeinen unterschiedliche Schutzmechanismen, welche sich in ihrer Schutzstärke, ihrem Ressourcenverbrauch und ihrem zeitlichen Determinismus unterscheiden.

Im Allgemeinen korreliert die Schutzstärke eines Schutzmechanismus mit seinem Ressourcenkonsum, sodass stärkere Schutzmechanismen im Allgemeinen einen höheren Ressourcenbedarf aufweisen als schwächere. Daher ist es nicht möglich, einen ausreichend starken Schutzmechanismus für jede Schutzfunktionalität festzulegen, da nicht gewährleistet werden kann, dass die unterschiedlichen Typen von Systemelementen der Feldebene über ausreichende Ressourcen für seine Ausführung verfügen.

Da unterschiedliche Stärken der Bedrohung auf der Feldebene existieren, unterschiedliche Auswirkungen von Angriffen auftreten können und die Feldgeräte und Feldbusse unterschiedliche Ressourcenkapazitäten aufweisen, muss es stattdessen möglich sein, die Erbringung der Schutzfunktionalitäten an die konkrete Situation anzupassen. Ist die Stärke der Bedrohung gering oder sind die zu erwartenden Auswirkungen einer Gefährdung unwesentlich, ist es nicht erforderlich, starke und somit ressourcenintensive Schutzmechanismen zu verwenden. Stattdessen können schwächere Schutzmechanismen eingesetzt werden, welche entsprechend weniger Ressourcen benötigen. Im gegenteiligen Fall sind starke Schutzmechanismen erforderlich.

Somit ergibt sich zusätzlich zum Erfordernis einer variablen Kombination von Schutzfunktionalitäten auch das Erfordernis einer variablen Erbringung der einzelnen Schutzfunktionalitäten durch unterschiedliche Schutzmechanismen. Um im Rahmen des Entwurfs einen minimalen Ressourcenverbrauch unter Berücksichtigung der erforderlichen Schutzstärke erzielen zu können, müssen die Schutzmechanismen so ausgelegt werden, dass unterschiedliche, zur Erbringung einer Schutzfunktionalität geeignete Schutzmechanismen beliebig gewählt werden können.

Um Schutzfunktionalitäten mit beliebigen geeigneten Schutzmechanismen erbringen zu können, müssen die Schutzfunktionalitäten von ihrer Erbringung, d. h. den Schutzmechanismen, getrennt werden. Daher werden auch die Schutzmechanismen als Module konstruiert, welche die für den jeweiligen Schutzmechanismus spezifischen Operationen implementieren. In den Schutzfunktionalitätsmodulen verbleiben die Operationen, welche von den Schutzmechanismen unabhängig sind. Abbildung 7.14 zeigt beispielhaft die Verknüpfung des Schutzfunktionalitätsmoduls, welches den Schutz der Vertraulichkeit bereitstellt, mit einem Modul, das einen spezifischen Schutzmechanismus (Verschlüsselungsalgorithmus AES) realisiert.

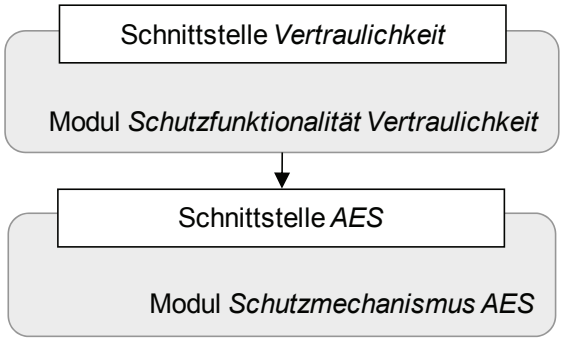

Abbildung 7.14: Verknüpfung eines Schutzfunktionalitätsmoduls mit einem Schutzmechanismusmodul

Die Forderung nach der Erbringung der Schutzfunktionalitäten durch jeweils unterschiedliche Schutzmechanismen setzt voraus, dass Schutzfunktionalitätsmodule mit verschiedenartigen Schutzmechanismusmodulen verknüpft werden können. Schnittstellen zwischen Modulen sind jedoch stets schutzmechanismusspezifisch. So ermöglicht beispielsweise die Importschnittstelle des Moduls *Schutzfunktionalität Vertraulichkeit* in Abbildung 7.14 eine Verbindung mit dem Modul *Schutzmechanismus AES*, welche über dessen Exportschnittstelle hergestellt wird. Es ist mit der Importschnittstelle des Moduls *Schutz der Vertraulichkeit* allerdings nicht möglich, Verbindungen mit anderen Modulen, welche andere geeignete Schutzmechanismen beinhalten, einzugehen.

Um beliebige, geeignete Schutzmechanismen mit Schutzfunktionalitäten verknüpfen zu können, müssen die Schnittstellen zwischen Schutzfunktionalitäten und Schutzmechanismus entkoppelt werden. Dafür ist eine zusätzliche Schnittstelle erforderlich, welche von den konkreten Modulschnittstellen abstrahiert. Dadurch können beliebige Schutzmechanismen, welche eine solche abstrakte Schnittstelle implementieren, verwendet werden. Abbildung 7.15 stellt die Verknüpfung eines Moduls *Schutzfunktionalität A* mit dem Modul *Schutzmechanismus X* über die abstrakte Schnittstelle *S* (links) dar sowie die Verknüpfung des gleichen Moduls *Schutzfunktionalität A* mit einem anderen Modul *Schutzmechanismus Y* (rechts).

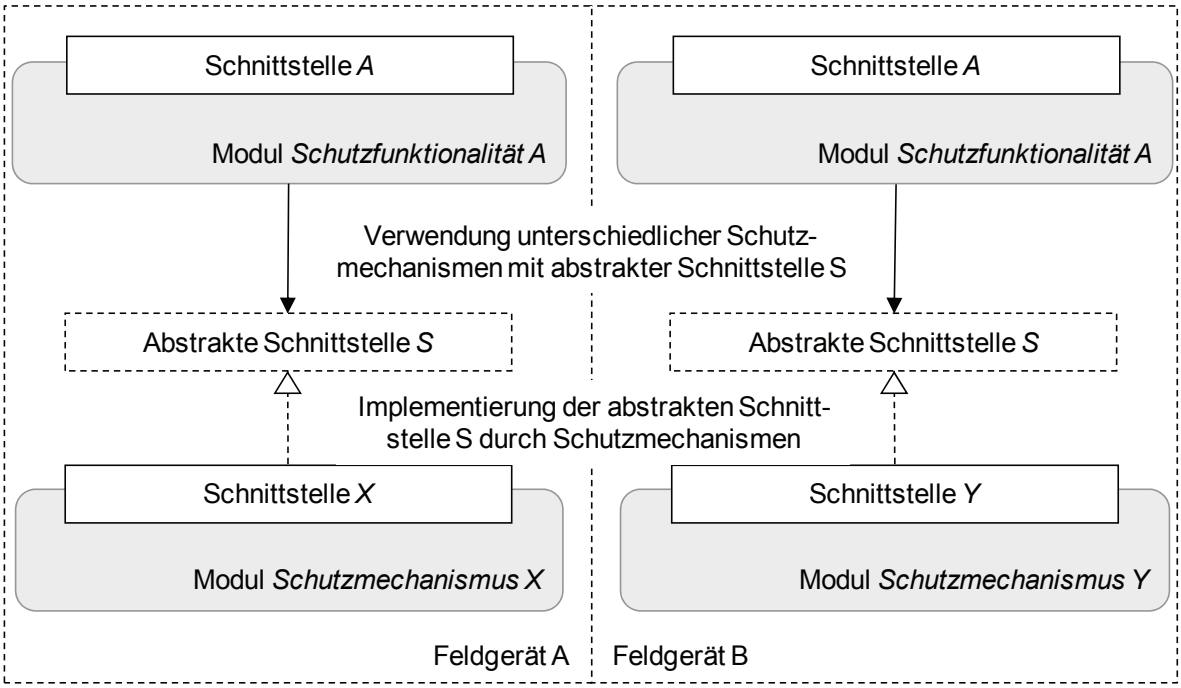

Abbildung 7.15: Verwendung unterschiedlicher Schutzmechanismen mittels abstrakter Schnittstelle

Mit dieser Modularisierung von Schutzmechanismen sowie der Entkopplung der Schnittstellen zwischen Schutzfunktionalitäten und Schutzmechanismen ist es möglich, die verfügbaren Ressourcen optimal zu nutzen, indem nur die minimal erforderliche Stärke realisiert wird.

7.4.4.2 Mehrfachverwendung von Schutzmechanismen

Sind mehrere Schutzfunktionalitäten zur Absicherung der IT-Sicherheit auf der Feldebene erforderlich, werden auch mehrere Schutzmechanismen zur Erbringung dieser Schutzfunktionalitäten benötigt. Werden beispielsweise die Schutzfunktionalitäten *Vertraulichkeit* und *inhaltliche Integrität* benötigt, würden ein symmetrisches Verschlüsselungsverfahren (vgl. Abschnitt 7.3.1) zur Erbringung des Vertraulichkeitsschutzes und eine Keyed-Hash-Funktion (vgl. Abschnitt 7.3.3) zur Erbringung des Schutzes der inhaltlichen Integrität eingesetzt werden. Der Keyed-Hash-Schutzmechanismus führt dabei zwei unterschiedliche Arten kryptografischer Operationen aus: die Berechnung eines kryptografischen Hashwerts sowie die Verschlüsselung desselben, um eine unbemerkte Veränderung des Hashwerts durch Angreifer zu erschweren. Es fällt auf, dass beide Schutzmechanismen mit der kryptografischen Verschlüsselung eine gleichartige Operation durchführen. Abbildung 7.16 stellt die von den unterschiedlichen Schutzmechanismen ausgeführten Operationen dar.

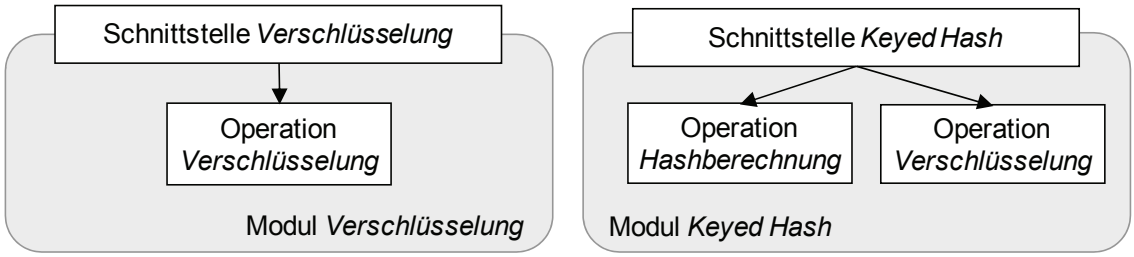

Abbildung 7.16: Ausführung gleichartiger Operationen durch unterschiedliche Schutzmechanismen

Da der Ressourcenverbrauch minimiert werden soll, um eine hohe Effizienz zu erreichen, ist es widersinnig, gleichartige Operationen mehrmals zu implementieren. Stattdessen können Operationen, welche von verschiedenen Schutzmechanismen benötigt werden, einmalig implementiert und mehrfach durch unterschiedliche Schutzmechanismen verwendet werden. Zusätzlich zu der Ressourcenersparnis erfüllt ein solcher Entwurf auch den Software-Engineering-Grundsatz „Don't Repeat Yourself" (DRY) [HuTh03], was eine Verbesserung der Softwarequalität (insbesondere der Wartbarkeit) mit sich bringt, da Änderungen an gleichartigen Operationen nur noch an einer Stelle durchgeführt werden müssen [Beck07].

In dem oben beschriebenen Beispiel würde der Schutzmechanismus *Keyed Hash* die in dem Schutzmechanismus *Verschlüsselung* enthaltene Verschlüsselungsoperation verwenden und diese nicht selbst implementieren. Dazu ist eine Verknüpfung zwischen den Schutzmechanismen erforderlich. Abbildung 7.17 stellt den Aufbau der Schutzmechanismen *Verschlüsselung* und *Keyed Hash* dar, wenn die Operation *Verschlüsselung* nur einmalig implementiert wird.

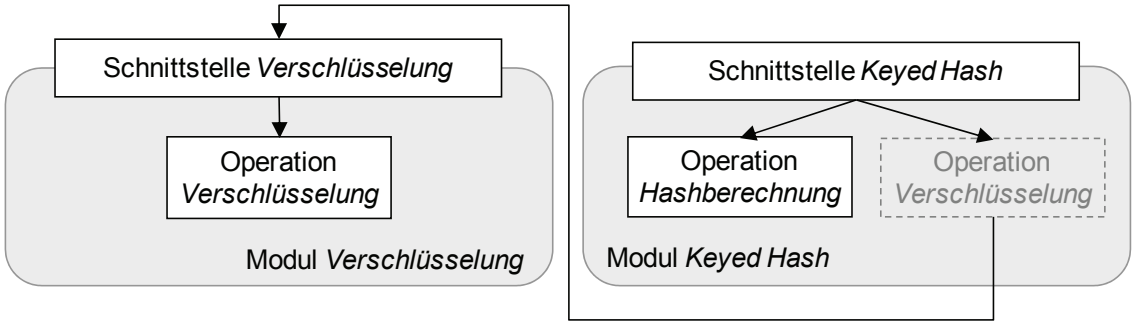

Abbildung 7.17: Mehrfachverwendung der Verschlüsselung

7.5 Softwarearchitektur des effizienten Schutzes

In den vorhergehenden Abschnitten wurden Schutzfunktionalitäten und Klassen von Schutzmechanismen zur Absicherung der IT-Sicherheit auf der Feldebene identifiziert. Es wurde festgelegt, dass die Schutzfunktionalitäten und die Schutzmechanismen in Form von Softwaremodulen entworfen werden. Um einen minimalen Ressourcenverbrauch zu erreichen, sollen die Module so verknüpft werden, dass eine Mehrfachverwendung von gleichartigen Operationen durch die Schutzmechanismen erreicht wird. Darüber hinaus wurde aber noch keine Ordnung festgelegt, welche die Softwaremodule systematisch miteinander in Beziehung setzt und somit die Struktur des gesamten Schutzes beschreibt. Eine derartige Struktur, bestehend aus Softwareelementen und deren Verbindungen untereinander, wird als Softwarearchitektur bezeichnet [SEI03]. Im Folgenden wird die Softwarearchitektur des effizienten Schutzes hergeleitet.

7.5.1 Ermittlung der Softwarearchitektur

Die in den vorangegangenen Abschnitten beschriebenen Schutzmechanismen zur Erkennung von Manipulationen der Kommunikation sind so beschaffen, dass sie einerseits ein Unterscheidungsmerkmal erzeugen, andererseits dieses Unterscheidungsmerkmal gegen Manipulationen durch Angreifer absichern. Die Unterscheidungsmerkmale sind ebenfalls Informationen, welche von Angreifern manipuliert werden können. Daher müssen die Unterscheidungsmerkmale so beschaffen sein, dass sie gegen Manipulationen geschützt sind, d. h., Veränderungen der Unterscheidungsmerkmale durch Angreifer müssen ebenso detektiert werden können wie Veränderungen der Nutzinformationen.

Existiert ein Schutzmechanismus SM_A, welcher ein Unterscheidungsmerkmal UM_A erzeugt, das durch einen anderen Schutzmechanismus SM_B abgesichert werden kann, benötigt der Schutzmechanismus SM_A keine eigenen Operationen zur Absicherung des Unterscheidungsmerkmals UM_A. Dazu kann stattdessen der Schutzmechanismus SM_B verwendet werden, sodass SM_A keine Ressourcen für die Absicherung seines Unterscheidungsmerkmals aufbringen muss. Abbildung 7.18 stellt den beschriebenen Übergang von den unabhängigen Schutzmechanismen SM_A und SM_B zu einer Verwendung von SM_B durch SM_A dar.

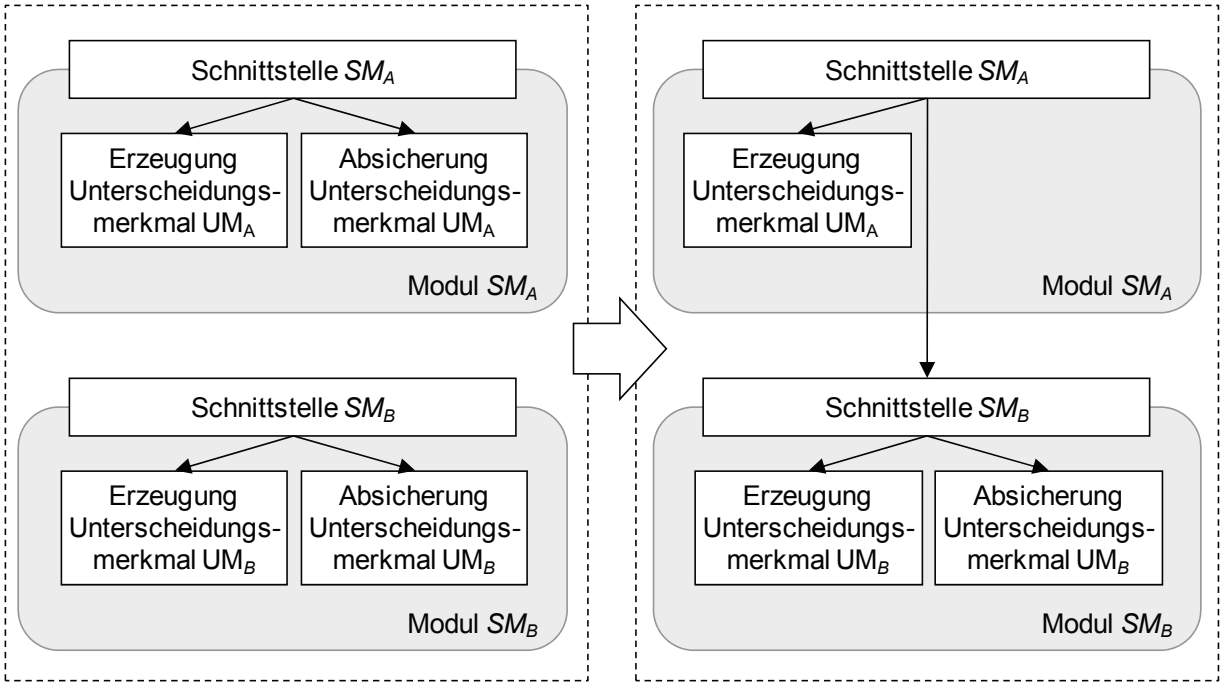

Abbildung 7.18: Übergang von Mehrfachimplementierung zu Mehrfachverwendung

Dadurch entsteht eine Hierarchie von Schutzmechanismen, wobei die Schutzmechanismen, deren Unterscheidungsmerkmale durch andere Schutzmechanismen abgesichert werden können, einer höheren Hierarchiestufe angehören als die Schutzmechanismen, welche zur Absicherung mit verwendet werden. Zur Strukturierung von Softwaresystemen, deren Bestandteile einer derartigen Hierarchie unterliegen, bieten sich sogenannte Schichtenarchitekturen an [Spin06]. Schichtenarchitekturen stellen Dienste zur Verfügung, welche von den Softwareelementen, die in den Schichten enthalten sind, implementiert werden. Zwischen den Schichten gelten Zugriffsregeln, welche durch die Ordnung der Schichtenarchitektur bestimmt werden, z. B. lineare Ordnung, strikte Ordnung oder baumartige Ordnung [Göhn07].

7.5.2 Definition von Schutzschichten

Die erforderlichen Schichten zur ressourcenschonenden Realisierung von Schutzfunktionalitäten ergeben sich aus den benötigten Klassen von Schutzmechanismen und den im vorigen Abschnitt beschriebenen Abhängigkeiten zwischen Schutzmechanismen. Aus diesem Grund werden die Schichten im Folgenden als *Schutzschichten* bezeichnet. In den folgenden Abschnitten wird hergeleitet, welche Klasse von Schutzmechanismen welcher Schutzschicht zuzuordnen ist und wie die einzelnen Schutzschichten miteinander in Beziehung stehen.

7.5.2.1 Schutzschicht 1: Verschlüsselungsverfahren

Die beschriebene Mehrfachverwendung von Schutzmechanismen durch Zugriff auf Schutzmechanismen einer tieferen Schutzschicht erfordert eine Schutzschicht, welche die unterste Hierarchieebene einnimmt und selbst keine weiteren Schichten nutzen muss, d. h. für sich allein be-

stehen kann, und dennoch keine Mehrfachimplementierung bewirkt. Von den in Abschnitt 7.3 beschriebenen Schutzmechanismusklassen erfüllt nur eine diese Forderung: die Klasse der Verschlüsselungsverfahren, welche zur Erbringung von Vertraulichkeit herangezogen wird. Daher wird der von dieser Klasse erbrachte Dienst der Ver-/Entschlüsselung der untersten Schutzschicht 1 zugeordnet. Konkrete Ver- und Entschlüsselungen können durch beliebig wählbare Verschlüsselungsalgorithmen durchgeführt werden.

7.5.2.2 Schutzschicht 2: kryptografische Hashfunktionen

Für die Erbringung der Schutzfunktionalität zur Absicherung der inhaltlichen Integrität wurden Keyed-Hash-Funktionen ausgewählt, welche eine kryptografische Hashfunktion ausführen und das Ergebnis zur Absicherung gegen Manipulationen verschlüsseln. Ein Angreifer ist somit zwar in der Lage, die Hashfunktion über eine inhaltlich veränderte Nachricht auszuführen, nicht aber, diese zu verschlüsseln, da er nicht über den Schlüssel verfügt, der ihm die korrekte Verschlüsselung des berechneten Hashwerts ermöglichen würde. Wird der vom Angreifer errechnete Hashwert nicht korrekt verschlüsselt, ergibt die Überprüfung des vom Nachrichtenempfänger entschlüsselten Hashwerts die Manipulation der Nachricht. Da die kryptografische Verschlüsselung von Schutzschicht 1 realisiert wird, kann der Schutz der inhaltlichen Integrität einer auf der Schutzschicht 1 aufbauenden Schutzschicht 2 zugeordnet werden. Die Schutzschicht 2 definiert somit einen Dienst, welcher kryptografische Hashfunktionen berechnet und die Schutzschicht 1 zur Absicherung der Hashwerte nutzt.

7.5.2.3 Schutzschicht 3: Identifikationsmerkmal

Zum Schutz der Autorisierung wird eine eindeutige Identifizierung von Nachrichtenabsendern benötigt. Zur Identifikation des Absenders ist ein Identifikationsmerkmal erforderlich, das mit der Nachricht übertragen wird. Dieses Identifikationsmerkmal muss so abgesichert werden, dass es nicht während der Nachrichtenübertragung unbemerkt verändert werden kann; ansonsten sind Angreifer in der Lage, Nachrichten so zu manipulieren, dass ein anderer als der tatsächliche Absender vorgegeben wird. Diese Funktionalität wird durch die Schutzschicht 2 zur Verfügung gestellt. Somit eignet sich die Erzeugung und Überprüfung von Identifikationsmerkmalen als Dienst, der einer dritten Schutzschicht zugeordnet wird und auf die Schutzschicht 2 zugreift.

Zusätzlich zum Schutz vor Veränderung des Identifikationsmerkmals muss auch sichergestellt werden, dass es Angreifern nicht möglich ist, eigene Nachrichten mit einem gefälschten, beispielsweise durch Abhören angeeigneten, Identifikationsmerkmal zu erzeugen. Dazu bietet sich die Verschlüsselung des Identifikationsmerkmals an. Eine solche Verschlüsselung des Identifikationsmerkmals würde aber stets denselben Schlüsseltext ergeben. Dadurch wäre es für Angreifer, welche in der Lage sind, Identifikationsmerkmale von Nachrichtenabsendern abzuhören, einfach möglich, auch das verschlüsselte Identifikationsmerkmal abzuhören. In diesem Fall können von Angreifern Nachrichten beliebigen Inhalts erzeugt werden, welche anhand des ver-

schlüsselten Identifikationsmerkmals einen falschen Absender vortäuschen, ohne dass eine solche Manipulation entdeckt werden kann.

Aus diesem Grund muss das Identifikationsmerkmal mit anderen (nichtkonstanten) Informationen verknüpft werden, welche ebenfalls in die Verschlüsselung eingehen und somit bei jeder Nachricht ein unterschiedliches verschlüsseltes Identifikationsmerkmal ergeben (diese Informationen werden daher auch als *Salt*, engl. Salz, bezeichnet, da sie den „Geschmack" des Identifikationsmerkmals variieren). Diese Informationen müssen eine ausreichend hohe Zufälligkeit aufweisen, damit sie von Angreifern nicht erraten werden können, und mit der zu schützenden Nachricht übertragen werden, damit der Empfänger das damit verknüpfte Identifikationsmerkmal wieder extrahieren kann.

Die Übertragung einer zusätzlichen Information in der Nachricht widerspricht der angestrebten Ressourceneinsparung. Eine Eigenschaft der von dieser Schutzschicht 3 benötigten kryptografischen Hashfunktion der Schutzschicht 2 ist die Abbildung der Nutzinformation in einen großen Wertebereich auf eine Weise, dass Änderungen der Nutzinformation keine Voraussagen auf die Änderungen des Transformationsergebnisses zulassen und dadurch ausreichend zufällig erscheinen. Daher bietet sich der von der Schutzschicht 2 erzeugte Hashwert zur Verknüpfung mit dem Identitätsmerkmal an.

Es existieren jedoch Feldbusstandards, die bereits (im Sinne der IT-Sicherheit) ungeschützte Identitätsmerkmale für Nachrichtenabsender enthalten. Würden diese verschlüsselt, wären bestehende Kommunikationsstacks nicht mehr in der Lage, derartig veränderte Nachrichten zu verarbeiten. Aus diesem Grund empfiehlt es sich, nicht das Identifikationsmerkmal mit dem Hashwert zu „salzen" und danach das Identifikationsmerkmal zu verschlüsseln, sondern stattdessen das Identifikationsmerkmal in die Hashwertberechnung einfließen zu lassen und diese danach zu verschlüsseln. Somit können bestehende Identifikationsmerkmale unverändert genutzt werden, wodurch sowohl die Kompatibilität zu dem Feldbusstandard erhalten bleibt als auch die Bandbreite nicht erhöht wird.

7.5.2.4 Schutzschicht 4: Zeitmarkierungen und Heartbeat

Durch die Absicherung mit den Schutzschichten 1–3 ist es für einen Angreifer nicht mehr möglich, eigene Nachrichten beliebigen Inhalts zu erzeugen. Allerdings können Nachrichten, die mit legitimen Nachrichten identisch sind und von einem Angreifer gegebenenfalls mehrfach gesendet werden, durch diesen Schutz nicht detektiert werden. Es ist ebenfalls nicht möglich, zu erkennen, dass Nachrichten verzögert bzw. gelöscht wurden.

Zum Schutz vor diesen Manipulationen können Zeitmarkierungen in Nachrichten eingefügt werden oder ein Heartbeat-Mechanismus herangezogen werden. Dazu muss bekannt sein, zu welchem Zeitpunkt eine Nachricht abgesendet wurde und wie lange die enthaltenen Informationen Gültigkeit besitzen. Dazu sind Zeitstempel in die abzusichernde Nachricht einzufügen, wel-

che diese Informationen beinhalten. Somit können bei Empfang von Nachrichten zeitliche Manipulationen erkannt werden. Zur Detektion von zeitlichen Manipulationen im Fall des Ausbleibens von nicht periodisch gesendeten Nachrichten muss zusätzlich ein Heartbeat-Mechanismus eingesetzt werden, der den Empfänger kontinuierlich unterrichtet, dass die abzusichernde Nachricht nicht gesendet wurde. Trifft weder eine Heartbeat- noch die abzusichernde Nachricht ein, kann der Empfänger davon ausgehen, dass Nachrichten von einem Angreifer gelöscht wurden.

Um Zeitstempel in einem verteilten System erzeugen und verifizieren zu können, ist eine zeitliche Synchronisation der einzelnen Systemelemente erforderlich. Eine derartige Synchronisation erfordert eine Zeitquelle, welche eine Zeitangabe an die Systemelemente übermittelt. Dabei ist zu verhindern, dass ein Angreifer die Identität der Zeitquelle vortäuschen oder die übermittelte Zeitangabe verändern kann, um die zeitliche Synchronisation zu stören. Ein derartiger Schutz wird durch die Schutzschichten 1–3 realisiert. Somit lassen sich die Zeitstempelerzeugung und -absicherung sowie die zeitliche Synchronisierung in eine vierte Schutzschicht einordnen, welche auf die Schutzschicht 3 zurückgreifen muss.

7.5.2.5 Schutzschicht 5: verteilte Überwachung der Feldgerätesoftware

Aufgrund der geringen Veränderlichkeit der Feldgerätesoftware kann bei der Konstruktion der Überwachung der Feldgerätesoftware davon ausgegangen werden, dass jede Veränderung der Software im Rahmen eines legitimen Updatevorgangs der Überwachungsfunktion kenntlich gemacht wird. Daher kann bei der Konstruktion der Überwachung auf Schadsoftwaresignaturen und komplexe Heuristiken, die in der Standard-IT erforderlich sind, verzichtet werden. Die Überwachung der Feldgerätesoftware muss ausschließlich in der Lage sein, Veränderungen an der Feldgerätesoftware zu detektieren.

Aufgrund der Verteilung der Überwachung von Feldgerätesoftware auf mehrere Feldgeräte wird eine Kommunikation zwischen den beteiligten Feldgeräten benötigt. Dabei sendet ein überprüfendes Feldgerät Prüfanfragen an das zu prüfende Feldgerät, das den gegenwärtigen Zustand der Software an das überprüfende Feldgerät zurücksendet. Angreifer, welche diese Kommunikation unbemerkt manipulieren können, sind daher in der Lage, unautorisierte Veränderungen von Feldgerätesoftware zu verbergen oder Manipulationen der Feldgerätesoftware vorzutäuschen. Beide Fälle würden „falsche“ Reaktionen der überwachenden Feldgeräte hervorrufen.

Daher ist eine vollständige Absicherung der Kommunikation zur verteilten Überwachung von Feldgerätesoftware erforderlich. Somit kann die verteilte Überwachung der Feldgerätesoftware einer fünften Schutzschicht zugeordnet werden, welche auf die Schutzschicht 4 zugreift und dadurch alle Schutzmechanismen der darunterliegenden Schichten zur Absicherung der Kommunikation nutzt.

7.5.3 Zusammenhang zwischen Schutzfunktionalitäten und Schutzschichten

Alle beschriebenen Schutzschichten außer Schutzschicht 1 erfordern Zugriff auf die jeweils darunterliegende Schutzschicht. Daraus ergibt sich eine strikte, lineare Schichtenarchitektur (Zugriff nur von höheren Hierarchieebenen auf tiefere Hierarchieebenen und Zugriff nur auf benachbarte Hierarchieebenen), welche die einzelnen Softwaremodule zur Detektion von Angriffen auf die IT-Sicherheit der Feldebene beinhaltet. Dabei werden die einzelnen Schutzfunktionalitäten jeweils durch Schutzmechanismen einer bestimmten Schutzschicht erbracht, die wiederum auf die darunterliegende Schutzschicht zugreifen müssen.

Die folgende Abbildung 7.19 ermöglicht die Ermittlung der erforderlichen Schutzschichten zur Erbringung von vorgegebenen Schutzfunktionalitäten bzw. die Ermittlung der möglichen Schutzfunktionalitäten, falls die Anzahl der möglichen Schutzschichten aufgrund der Ressourcenknappheit der Feldebene begrenzt ist.

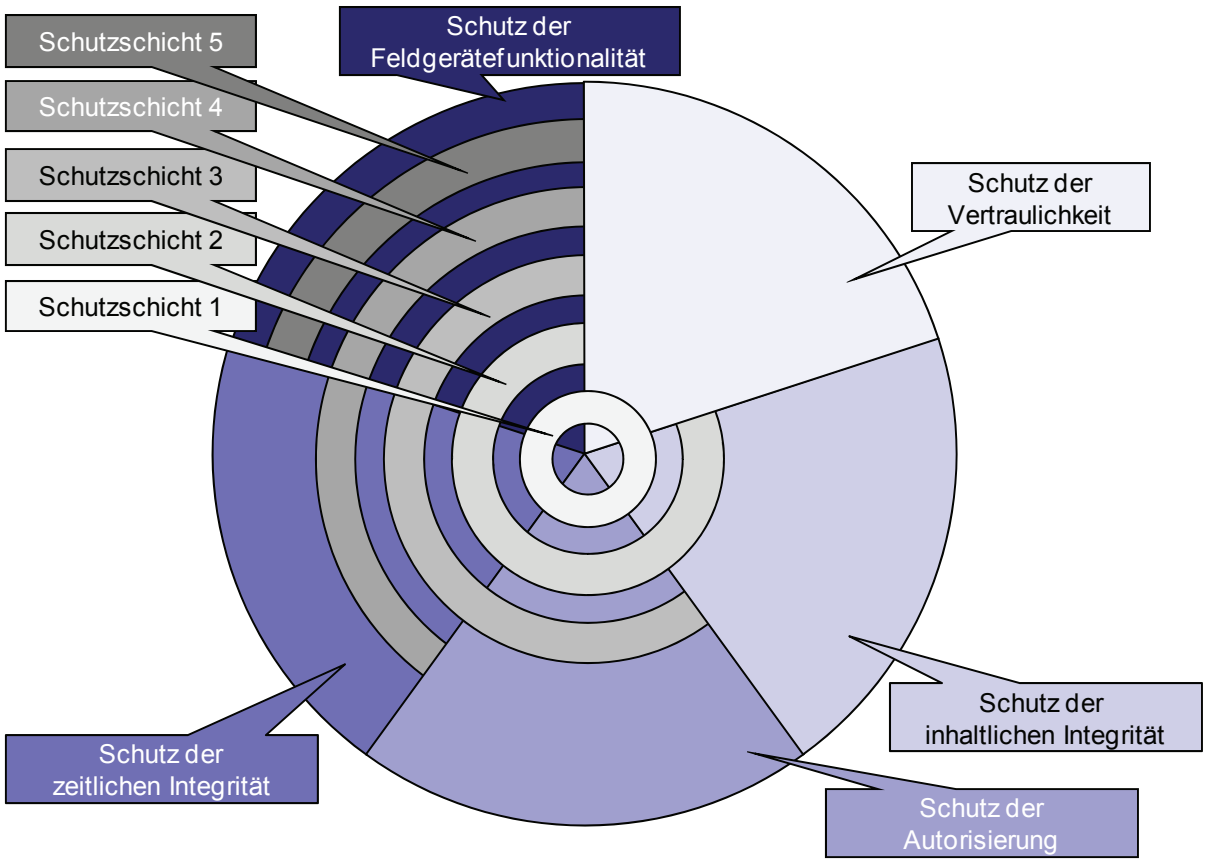

Abbildung 7.19: Zusammenhang zwischen Schutzfunktionalitäten und Schutzschichten

7.5.4 Konstruktion der Schutzschichten

In den Schutzschichten werden unterschiedliche Unterscheidungsmerkmale erzeugt bzw. überprüft. Zur Erzeugung dieser Unterscheidungsmerkmale werden jeweils unterschiedliche Infor-

mationen herangezogen. Die Schutzschichten müssen daher diese Informationen akquirieren und den jeweiligen Schutzmechanismen zur Verfügung stellen. Im Fall der Integration von Unterscheidungsmerkmalen muss das von den Schutzmechanismen berechnete Unterscheidungsmerkmal in die Nachricht integriert werden, im Falle der Überprüfung ist das bereits vorhandene Unterscheidungsmerkmal mit dem berechneten zu vergleichen. Abbildung 7.24 stellt den Ablauf dieser Vorgänge dar.

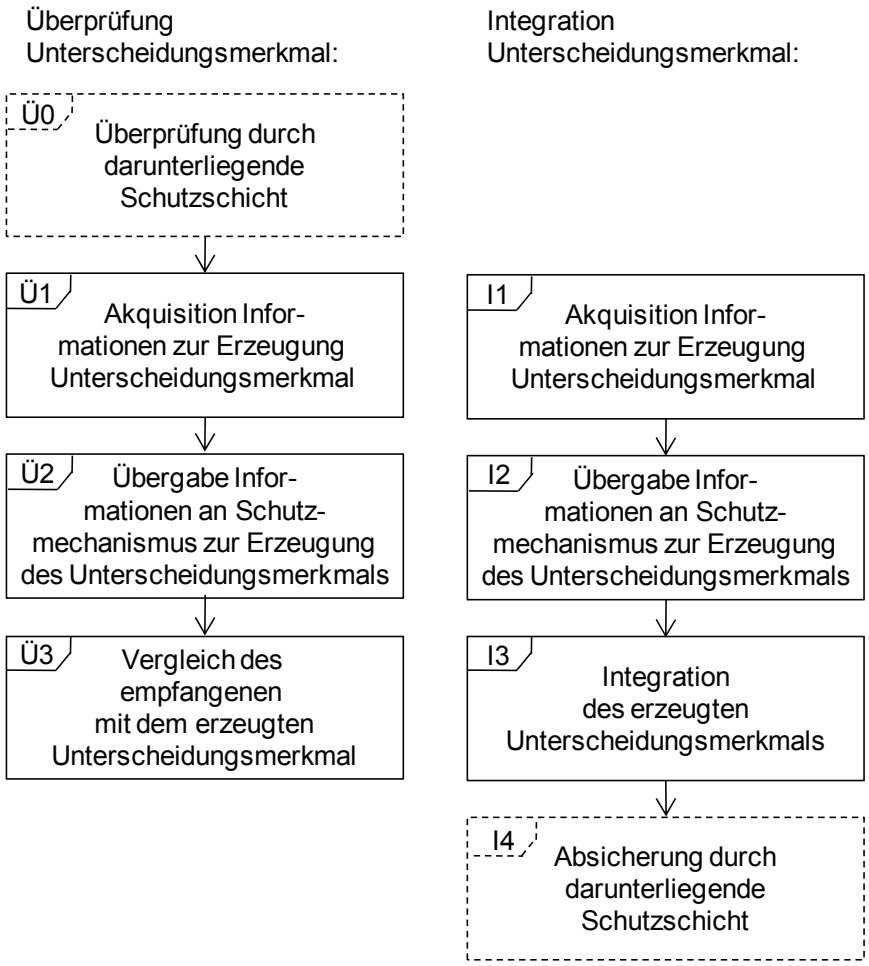

Abbildung 7.20: Ablauf der Überprüfung und der Integration von Unterscheidungsmerkmalen

Der in Abbildung 7.20 skizzierte Ablauf ist für jede Schutzschicht spezifisch, jedoch unabhängig von dem in der Schutzschicht verwendeten Schutzmechanismus. Die Operationen, welche von konkreten Schutzmechanismen unabhängig sind, wurden in Abschnitt 7.4.4.1 in Modulen zusammengefasst, die als Schutzfunktionalitätsmodule bezeichnet wurden. In der dargestellten Schichtenarchitektur erbringen die Module die Schutzfunktionalitäten jedoch in Zusammenarbeit, sodass für die einzelnen Schutzfunktionalitäten keine dedizierten Module existieren können. Daher werden die Module, welche die schutzschichtspezifischen Operationen enthalten, als *Schutzschichtmodule* bezeichnet.

Die Schutzschichtmodule rufen einen Schutzmechanismus und das Schutzschichtmodul der darunterliegenden Schutzschicht auf und stellen dadurch eine Schutzfunktionalität zur Verfügung.

Da die in der Abbildung 7.20 dargestellten Schritte Ü1 und I1 sowie Ü2 und I2 identisch sind, können diese jeweils zu einem Funktionsblock zusammengefasst werden, wodurch wiederum Speicherplatz eingespart wird. Abbildung 7.21 stellt den Aufbau einer Schutzschicht dar.

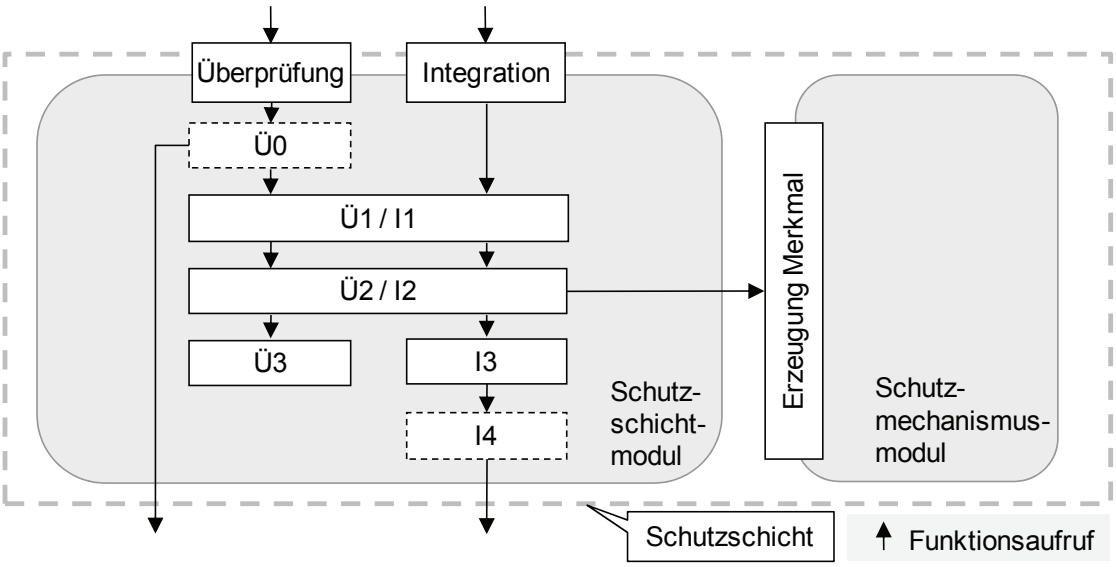

Abbildung 7.21: Prinzipieller Aufbau einer Schutzschicht

Die Informationen, welche zur Erzeugung der Unterscheidungsmerkmale benötigt werden, sind je nach Kontext, d. h. dem verwendeten Feldbus- bzw. Feldgerätestandard sowie den zu übertragenden Nutzdaten bzw. Feldgeräteprogrammen, unterschiedlich. Um das im Rahmen dieser Arbeit entwickelte Konzept unabhängig von spezifischen Standards einsetzen zu können, existieren zwei Möglichkeiten der Anpassung. Einerseits kann je nach Kontext für jede Schutzschicht ein spezifisches Schutzschichtmodul entwickelt werden. Andererseits ist es auch möglich, die Schutzschichtmodule entsprechend parametrierbar zu gestalten, wodurch ihnen die erforderlichen Kenntnisse über den Kontext vermittelt werden.

Da Schutzschichtkomponenten, welche für jeden Kontext individuell zu entwickeln sind, einen großen Entwicklungsaufwand und somit hohe Entwicklungskosten bedeuten, wird für das Konzept zum effizienten Schutz der Feldebene die zweite Möglichkeit gewählt: Die Schutzschichtkomponenten werden um die Möglichkeit der Parametrierung zum Zweck der Anpassung an unterschiedliche Feldbus- und Feldgerätestandards erweitert. Da mehrfach verwendbare Komponenten im Allgemeinen mehr Ressourcen verbrauchen, als solche, die gezielt für einen bestimmten Kontext entwickelt wurden, werden die Parameter auch dazu verwendet, nur die für einen konkreten Fall tatsächlich benötigten Ressourcen auszuwählen. Dies kann beispielsweise mittels *bedingter Compilierung* erreicht werden, bei der nur genau der Quellcode übersetzt wird, der für die in einem konkreten Fall zu bewältigende Aufgabe erforderlich ist. Daraus folgt die in Abbildung 7.22 dargestellte Exportschnittstelle einer Schutzschicht.

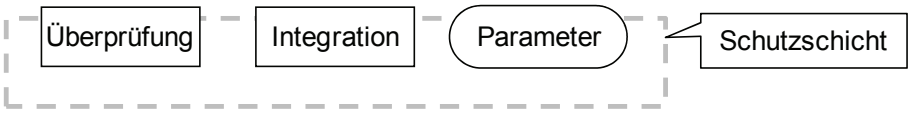

Abbildung 7.22: Exportschnittstelle einer Schutzschicht

7.5.5 Aufbau geschützter Feldbusnachrichten

Damit das im Rahmen dieser Arbeit entwickelte Konzept an unterschiedliche Feldbusse angepasst werden kann, muss es in der Lage sein, die Unterscheidungsmerkmale zur Erkennung von unautorisierten Informationsveränderungen in die Nachrichten unterschiedlicher Feldbusstandards zu integrieren. Damit die Konformität der Nachrichten zu der Feldbusspezifikation erhalten bleibt, muss die Integration der Unterscheidungsmerkmale so geschehen, dass der Aufbau bestehender Feldbusnachrichten nicht verändert wird.

Da keine Veränderung bestehender Nachrichtenformate durchgeführt werden soll, können zusätzliche Informationen, wie Unterscheidungsmerkmale, nur in die Ebene 7 des ISO/OSI-Referenzmodells (vgl. Abschnitt 2.3.2) integriert werden. Sind in unteren Ebenen jedoch Informationen vorhanden, die als Unterscheidungsmerkmal herangezogen werden können, z. B. Identitätsmerkmale oder Zeitstempel, werden diese genutzt und nicht zusätzlich in die Ebene 7 eingefügt, da dies zusätzliche Kommunikationsbandbreite erfordern würde. Abbildung 7.23 stellt den Aufbau geschützter Feldbusnachrichten dar.

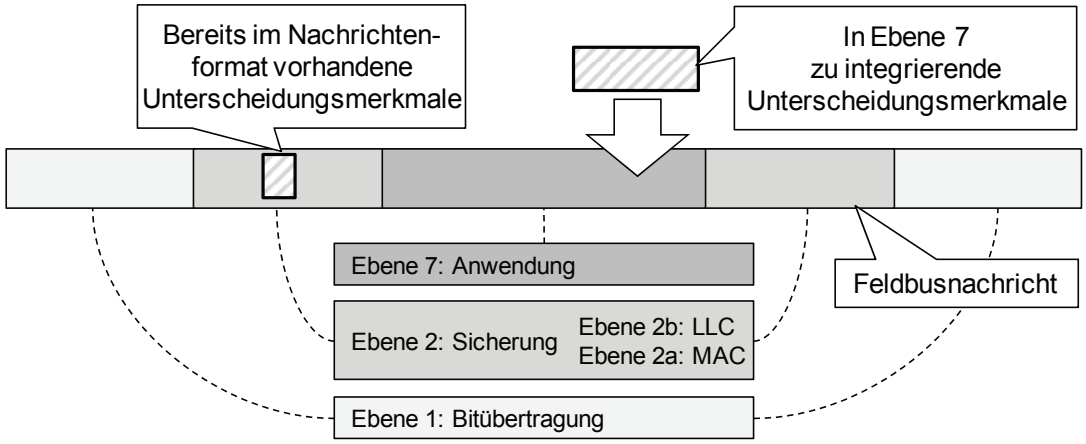

Abbildung 7.23: Aufbau geschützter Feldbusnachrichten

Die Position der in die Ebene 7 zu integrierenden Unterscheidungsmerkmale innerhalb der Ebene 7 ist aus IT-Sicherheits-Gesichtspunkten nicht relevant; es ist jedoch sicherzustellen, dass die Unterscheidungsmerkmale entsprechend der Reihenfolge der in Abschnitt 7.5.2 hergeleiteten Schichtenarchitektur abgesichert werden. Um die Rechenzeit für die Erzeugung der Unterscheidungsmerkmale zu minimieren, bietet es sich jedoch an, die Unterscheidungsmerkmale so anzuordnen, dass die Schutzmechanismen (welche außer den Nutzdaten auch die Unterscheidungsmerkmale der darüberliegenden Schutzschichten einbeziehen) auf einem zusammenhän-

genden Speicherbereich operieren können, was die Anzahl von Sprüngen an die erforderlichen Bereiche reduziert. Abbildung 7.24 stellt den Aufbau der geschützten Feldbusnachrichten dar.

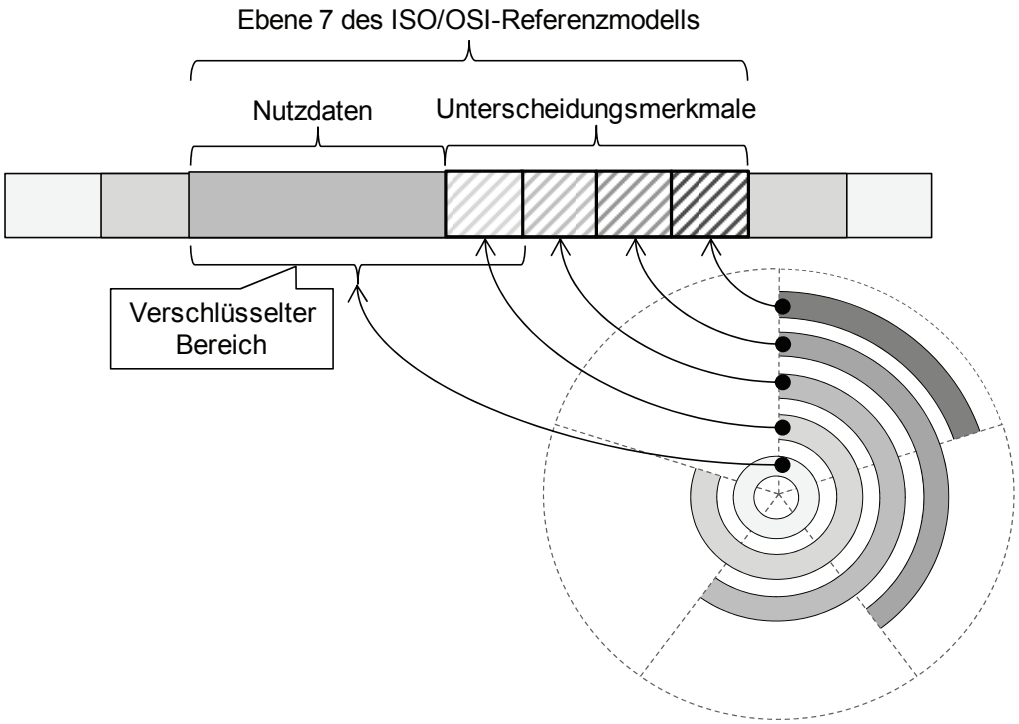

Abbildung 7.24: Abbildung der Schutzschichten in die Struktur von Feldbusnachrichten

In diesem Kapitel wurde das Konzept des effizienten Schutzes der IT-Sicherheit auf der Feldebene hergeleitet. Dazu wurde dargestellt, dass fünf unterschiedliche Schutzfunktionalitäten benötigt werden, um alle Arten von Angriffen gegen die IT-Sicherheit auf der Feldebene detektieren zu können. Ebenso wurde dargestellt, wie die Reaktionen beschaffen sein müssen, um den Übergang des angegriffenen Automatisierungssystems in unerwünschte Fehlerzustände zu unterbinden. Ausgehend von der Festlegung, den Schutz auf informationstechnische Art zu realisieren, wurden die Klassen von Schutzmechanismen ausgewählt, welche am besten zur Erbringung der Schutzfunktionalitäten geeignet sind. Aufgrund des großen Ressourcenbedarfs der geeigneten Schutzmechanismen und der geringen Ressourcen der Feldebene wurde eine ressourcenschonende Konstruktion des Schutzes konzipiert, welche auf der Auswahl und Kombination von Schutzfunktionalitäten und dem synergetischen Zusammenwirken von Schutzmechanismen basiert. Dazu wurde eine aus fünf Schutzschichten bestehende Softwarearchitektur hergeleitet, welche Kombinationsvorschriften definiert, um den Synergieeffekt unter Beibehaltung der Wirksamkeit des Schutzes zu erzielen. Da in diesem Kapitel offengelassen wurde, welche Schutzfunktionalitäten in einem konkreten Fall benötigt werden und mit welchen Schutzmechanismen diese Schutzfunktionalitäten zu erbringen sind, beschreibt das folgende Kapitel eine Vorgehensweise zur Absicherung der Feldebene, welche insbesondere auch diese Fragen thematisiert.

8 Vorgehen zur Absicherung der Feldebene von Automatisierungssystemen

In dem vorangegangenen Kapitel wurde das Konzept des effizienten Schutzes der IT-Sicherheit auf der Feldebene hergeleitet. Ein wesentlicher Aspekt dieses Konzepts besteht darin, nur die in einer konkreten Situation erforderlichen Schutzfunktionalitäten sowie geeignete Schutzmechanismen einzusetzen. Kapitel 7 beschreibt jedoch nicht, wie zu ermitteln ist, welche Schutzfunktionalitäten und Schutzmechanismen jeweils benötigt werden. In diesem Kapitel wird daher eine Vorgehensweise zur Ermittlung der erforderlichen Schutzfunktionalitäten sowie geeigneter Schutzmechanismen beschrieben. Eine solche Ermittlung ist bei jedem abzusichernden Automatisierungssystem durchzuführen.

8.1 Übersicht über das Vorgehen

Da ein möglichst geringer Ressourcenverbrauch erzielt werden soll, sollen nicht alle Systemelemente der Feldebene pauschal vor allen Angriffen abgesichert werden, sondern nur vor solchen, bei denen eine Absicherung als erforderlich erachtet wird. Die Wirkungsweise des im Rahmen dieser Arbeit vorgestellten Konzepts besteht in der Verhinderung von Übergängen des Automatisierungssystems in unerwünschte Zustände. Um diese Verhinderung zu erreichen, ist zunächst zu klären, welche Zustände des Systems durch Angriffe erreicht werden können und welche davon als unerwünscht betrachtet werden (Bedrohungsanalyse).

Um entscheiden zu können, vor welchen Angriffen Schutz realisiert werden muss, ist zu betrachten, welches Risiko aus den einzelnen Angriffen erwächst (Risikobewertung). Dadurch ist es möglich, zu beurteilen, welche Systemelemente der Feldebene vor welchen Arten von Angriffen abzusichern sind, bei denen das Risiko groß genug ist, um den für die Absicherung erforderlichen Ressourcenverbrauch zu rechtfertigen. Aus dieser Entscheidung lässt sich ableiten, welche Schutzfunktionalitäten auf welchen Systemelementen der Feldebene mit welchen Schutzmechanismen ausgeführt werden müssen (Entwurf des Schutzes).

Schließlich ist es erforderlich, den Schutz in Bezug auf Effektivität und Effizienz zu prüfen (Schutzvalidierung). Dabei ist zu prüfen, ob der Schutz gegenüber den identifizierten Angriffen wirksam ist und auf den jeweiligen Feldgeräten ausgeführt werden kann, d. h. ob die Anforderungen bezüglich Speicherplatz- und Bandbreitenbeschränkungen sowie an die zeitliche Determiniertheit und die Rechenzeit erfüllt werden.

8.2 Bedrohungsanalyse

Die im Rahmen dieser Arbeit relevanten Bedrohungen sind die unterschiedlichen Manipulationen der Feldebene mit physischem Zugriff, die in Abschnitt 7.1 beschrieben sind. Da derartige Bedrohungen von Menschen hervorgerufen werden, ist das Auftreten von Bedrohungen von der Motivation bzw. den Fähigkeiten potenzieller Angreifer abhängig. Um die Bedrohungslage abzuschätzen, ist es daher erforderlich, mögliche Motivationen für Angriffe zu identifizieren.

Im Folgenden werden Beispiele für unterschiedliche Motivationen genannt, aufgrund derer Angriffe auf die IT-Sicherheit der Feldebene von Automatisierungssystemen durchgeführt werden können. Im Rahmen einer konkreten Beurteilung der Bedrohungslage können diese Motivationen zur Beurteilung der Bedrohungslage herangezogen werden.

- Finanzielle Motivation (z. B. von Personen mit dem Ziel Erpressung, Spionage oder Diebstahl)
- Abneigung gegen Unternehmen, welche Automatisierungssysteme betreiben (z. B. von enttäuschten ehemaligen Kunden, Partnern, Angestellten, Wettbewerbern)
- Abneigung gegen konkrete Automatisierungssysteme (z. B. von Personen, deren Wohnsitz in unmittelbarer Umgebung eines Automatisierungssystems liegt)
- Ideologische Abneigung gegen bestimmte Branchen oder Arten von Automatisierungssystemen (z. B. von Umweltschützern, Tierschützern)

Lassen sich derartige Motivationen auf das betrachtete Automatisierungssystem abbilden, können Misuse-Cases [SiOp05] gebildet werden, welche jeweils ein Ziel eines Angreifers beschreiben sowie die Interaktionen, die zur Erreichung dieses Ziels durchgeführt werden müssen. Um das Automatisierungssystem vor einem Misuse-Case zu schützen, ist es erforderlich, alle Möglichkeiten zu identifizieren, mit denen das Ziel des Angreifers durch Manipulationen der IT-Sicherheit auf der Feldebene erreicht werden kann.

Dazu ist in einem ersten Schritt zu identifizieren, welche Funktionalitäten für die Zielerreichung des Angreifers ausgeführt bzw. gestört werden müssen und auf welchen Feldgeräten diese angesiedelt sind (z. B. Ausführung der Öffnungsfunktion des Dachsteuergeräts bei einem elektrisch beweglichen Cabriodach zum Zweck des Autodiebstahls). Da Feldgerätefunktionalitäten im Allgemeinen Informationen von außerhalb des Feldgeräts benötigen, ist in einem zweiten Schritt zu ermitteln, welche Informationen von der betreffenden Feldgerätefunktionalität verarbeitet werden und durch welche Kommunikationsvorgänge diese übermittelt werden (z. B. Übermittlung der Stellung des Dach-öffnen-Schalters mittels CAN-Nachrichten). Derartige Informationen werden von Funktionalitäten auf anderen Feldgeräten erzeugt (z. B. einem Armaturensteuergerät), sodass der erste Schritt erneut durchgeführt werden muss, um diese Funktionalitäten sowie die ausführenden Feldgeräte zu ermitteln. Somit ergibt sich der in Abbildung 8.1 dar-

gestellte Zyklus, der so lange zu durchlaufen ist, bis alle beteiligten Feldgerätefunktionalitäten und Feldbuskommunikationsvorgänge ermittelt sind.

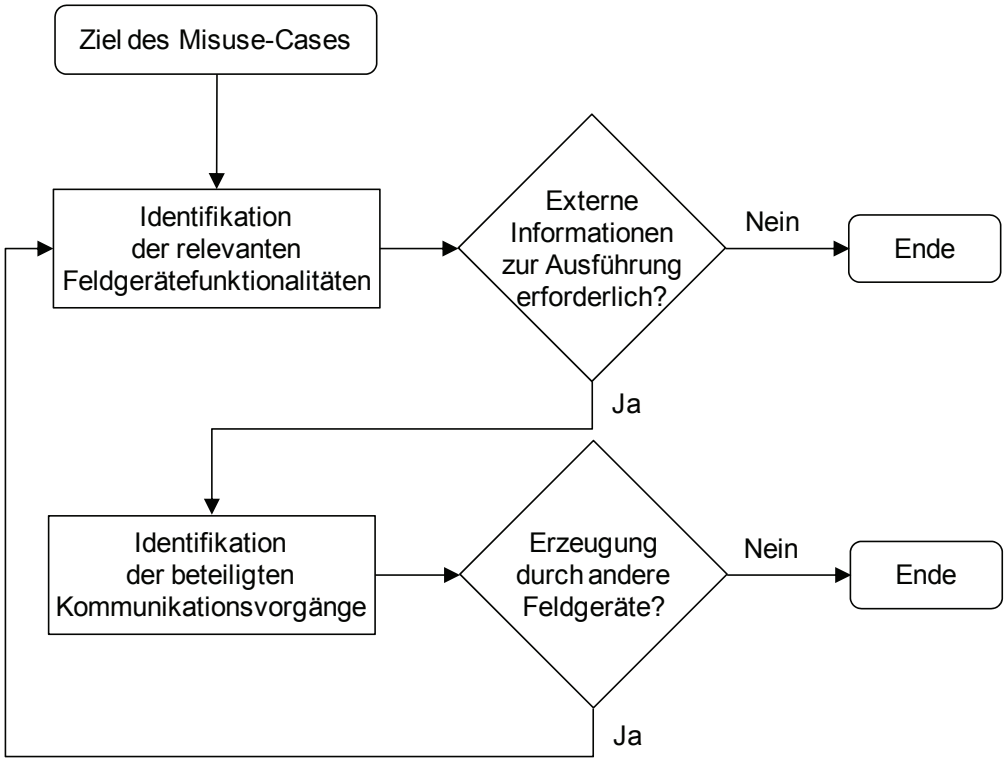

Abbildung 8.1: Ermittlung der bedrohten Feldgerätefunktionalitäten und Feldbusnachrichten

Das Ergebnis dieser Betrachtung ist eine Aufstellung der Feldgerätefunktionalitäten einschließlich der Information, auf welchen Feldgeräten diese ausgeführt werden, sowie der zur Ausführung der Feldgerätefunktionalitäten erforderlichen Informationen einschließlich der Kommunikationsvorgänge, mit denen die Informationen übermittelt werden. In Anhang A.1 ist die Durchführung der beschriebenen Prozedur anhand eines Beispiels dargestellt.

8.3 Risikobewertung

Ein wesentlicher Aspekt des in Kapitel 7 vorgestellten Konzepts besteht in der Anpassung des Schutzes an die geringen Ressourcen, die auf der Feldebene zur Verfügung stehen. Aus diesem Grund werden nur die erforderlichen Schutzfunktionalitäten mit der erforderlichen Schutzstärke eingesetzt. Um entscheiden zu können, welche Schutzfunktionalitäten erforderlich sind, werden Risikobewertungen für die möglichen Angriffe auf die ermittelten Feldgerätefunktionalitäten und Feldbuskommunikationsvorgänge benötigt. Ist das ermittelte Risiko eines Angriffs auf eine Feldgerätefunktionalität bzw. eine Feldbuskommunikation größer als das größte noch vertretbare Risiko (Grenzrisiko), so ist die für den Schutz vor diesem Angriff erforderliche Schutzfunktionalität einzusetzen und somit das Risiko zu reduzieren.

Wie in Abschnitt 3.1.1 dargestellt, beschreibt das Risiko das Produkt aus der Wahrscheinlichkeit für das Auftreten eines Angriffs und der Auswirkung des Angriffs (Schadenspotenzial). Im Gegensatz zu Risikobetrachtungen zur Gewährleistung von *Safety* ist zu berücksichtigen, dass die Wahrscheinlichkeit eines Angriffs von der zu erwartenden Auswirkung nicht unabhängig ist. Wird das Ziel eines Angreifers durch einen möglichen Angriff mit hoher Wahrscheinlichkeit nicht erreicht, so reduziert sich auch die Wahrscheinlichkeit, dass der Angriff überhaupt durchgeführt wird. Auch die Existenz alternativer, weniger aufwändiger Möglichkeiten für den Angreifer, sein Ziel zu erreichen, kann die Wahrscheinlichkeit für das Auftreten eines Angriffs verändern. Bei einer qualitativen Bestimmung der Wahrscheinlichkeit in vier Stufen und der Auswirkung in fünf Stufen ergibt sich das Risiko wie in Tabelle 8.1 dargestellt

Tabelle 8.1: Bestimmung des Risikos aus Wahrscheinlichkeit und Auswirkung

Auswirkung / **Wahrscheinlichkeit**	0 (keine)	1 (gering)	2 (mittel)	3 (schwer)	4 (sehr schwer)
1 (sehr gering)	0	1	2	3	4
2 (gering)	0	2	4	6	8
3 (mittel)	0	3	6	9	12
4 (hoch)	0	4	8	12	16

8.4 Entwurf des Schutzes

8.4.1 Identifikation und Realisierung der erforderlichen Schutzfunktionalitäten

Aus der Risikobewertung ergibt sich, welche Feldgerätefunktionalitäten bzw. Feldbuskommunikationsvorgänge vor welchen Arten von Angriffen zu schützen sind. Daraus lassen sich die erforderlichen Schutzfunktionalitäten sowie deren Realisierung auf den Feldgeräten ableiten. Ist ein Kommunikationsvorgang vor einer bestimmten Angriffsart zu schützen, so ist die dafür erforderliche Schutzfunktionalität so auf die Feldgeräte aufzubringen, dass der Absender der betreffenden Nachricht geeignete Unterscheidungsmerkmale in die Nachricht integrieren kann und alle Empfänger in der Lage sind, diese Unterscheidungsmerkmale zu verifizieren. Analog ist die Funktionalität zum verteilten Schutz der Feldgerätefunktionalität auf die Feldgeräte aufzubringen, deren Funktionalität zu schützen ist, sowie die Feldgeräte, welche die Überwachung durchführen.

Die Unterscheidungsmerkmale, anhand derer manipulierte Nachrichten bzw. Programmcodes detektiert werden können, werden durch Schutzmechanismen realisiert, die in Schutzschichten angeordnet sind. Welche Schutzschichten erforderlich sind, lässt sich mithilfe des in Abbildung 7.19 dargestellten Zusammenhangs zwischen Schutzfunktionalitäten und Schutzschichten ermitteln. So benötigt beispielsweise die Funktionalität *Schutz der Autorisierung* die Schutzschichten 1–3. Abbildung 8.2 stellt beispielhaft die Realisierung des Schutzes der Autorisierung dar.

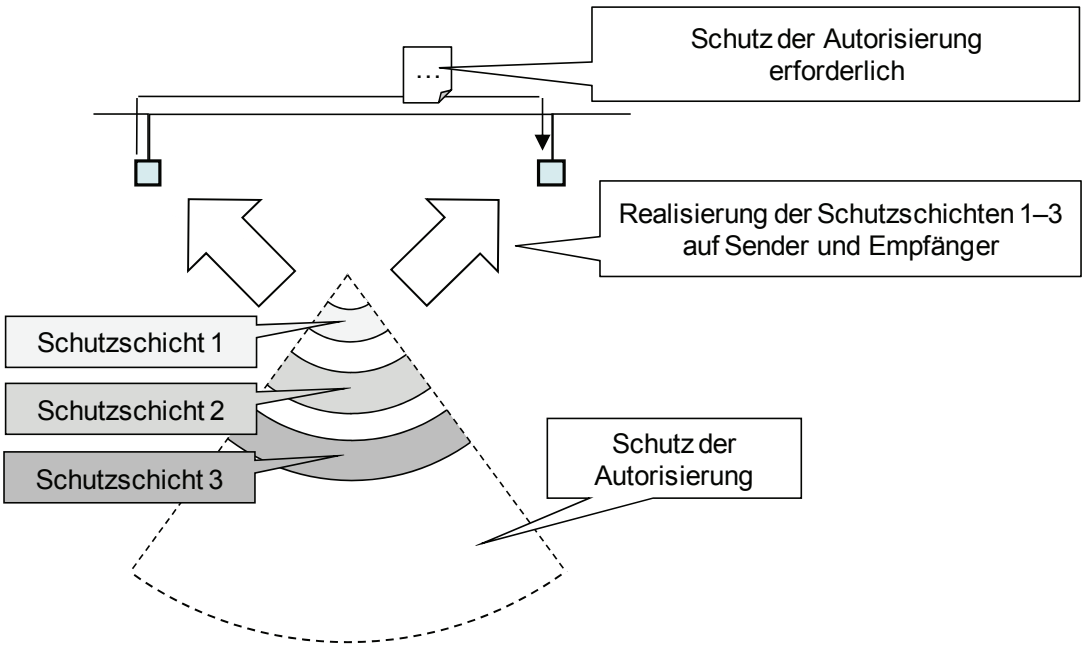

Abbildung 8.2: Realisierung des Schutzes der Autorisierung

Die Schutzschichten stellen definierte Dienste zur Erbringung der Schutzfunktionalitäten zur Verfügung. Diese Dienste müssen durch Schutzmechanismen realisiert werden. Im folgenden Abschnitt wird daher für jede Schutzschicht ein Schutzmechanismus ausgewählt, der aufgrund seiner Eigenschaften zum Einsatz auf der Feldebene geeignet ist.

8.4.2 Auswahl von Schutzmechanismen

Die Kriterien für einen optimalen Schutzmechanismus ergeben sich aus den Anforderungen an das vorgestellte Konzept: Um gegen Angriffe auf die IT-Sicherheit der Feldebene wirksam zu sein, ist ein hoher Grad an Schutzstärke erforderlich. Damit der Schutz eine hohe Effizienz aufweist, soll der Schutzmechanismus nur einen geringen Ressourcenverbrauch aufweisen. Um Echtzeitfähigkeit zu ermöglichen, soll die Ausführung des Schutzmechanismus zeitlich deterministisch ablaufen.

Schutzmechanismen, welche einen hohen Grad an Schutzstärke aufweisen, erfordern jedoch üblicherweise mehr Ressourcen als solche mit einer geringen Schutzstärke. Aufgrund der im Allgemeinen geringen Ressourcen der Feldebene kann daher kein optimaler Schutzmechanis-

mus benannt werden, sondern es ist zwischen Schutzstärke und Ressourcenverbrauch abzuwägen. Dabei ist auch die Verteilung des Bedarfs auf die unterschiedlichen Arten von Ressourcen zu betrachten, die auf der Feldebene existieren: Speicherplatz, Rechenzeit und Bandbreite. Die Auswahl von zeitlich deterministischen Schutzmechanismen ist jedoch unabhängig von den sonstigen Eigenschaften möglich.

Die Schutzstärke von Schutzmechanismen beruht auf dem Aufwand zur Lösung bestimmter mathematischer Problemstellungen. Dabei ist es im Allgemeinen nicht möglich, formal zu beweisen, dass ein bestimmter Minimalaufwand zur Lösung der Problemstellung erforderlich ist. Aus diesem Grund wird die Stärke von Schutzmechanismen üblicherweise in qualitativer Weise angegeben, im Folgenden wird eine Einteilung in geringe, mittlere und hohe Stärke verwendet. Der Ressourcenverbrauch von Schutzmechanismen hängt neben der Funktionsweise des Mechanismus auch von Randbedingungen bei der Implementierung ab (z. B. die Plattform, auf der der Schutzmechanismus ausgeführt wird, oder die Optimierung auf eine bestimmte Ressourcenart).

In der folgenden Tabelle 8.2 wird daher für jede Schutzschicht ein Schutzmechanismus vorgeschlagen, der nach heutigem Stand der Technik einen guten Kompromiss zwischen Schutzstärke und Ressourcenverbrauch darstellt (vgl. [Wiki07] [Wiki08a] [Wiki08b] [Wiki08c] [Wiki08d]). Die ausgewählten Schutzmechanismen verfügen über eine Wirksamkeit von mittlerer Stärke, d. h., sie sind zum Schutz vor Angreifern mittlerer Gefährlichkeit ausgelegt (vgl. Abschnitt 3.2.3). Um Echtzeitfähigkeit zu ermöglichen, sind alle vorgeschlagenen Schutzmechanismen zeitlich deterministisch.

Tabelle 8.2: Empfohlene Schutzmechanismen

Schutzschicht	**Empfohlener Schutzmechanismus**
1	Advanced Encryption Standard (AES), 128 bit Schlüssellänge
2	Secure Hash Algorithm 1 (SHA-1), 160 bit Hashwertlänge
3	Numerischer Identifier, 128 bit Länge
4	Real-Time-Stamp mit variabler zeitlicher Auflösung
5	Verteilte Überwachung

Nach heutigem Stand sind diese Schutzmechanismen gut zur Absicherung der IT-Sicherheit auf der Feldebene geeignet. Es sind jedoch zukünftige Entwicklungen zu berücksichtigen, durch welche die Schutzwirkung der vorgeschlagenen Schutzmechanismen beeinträchtigt werden kann, wie die Entwicklung neuer mathematischer Verfahren, welche die Lösung der Problemstellungen vereinfachen, auf denen die Schutzwirkungen beruhen.

8.5 Schutzvalidierung

Zur Validierung des Schutzes der IT-Sicherheit auf der Feldebene sind zwei unterschiedliche Prüfungen erforderlich: einerseits die Prüfung der Effektivität des Schutzes, andererseits die Prüfung der Effizienz. Um die Effektivität des Schutzes zu prüfen, gelten Penetrationstests, d. h. Angriffe, die zu Zwecken der Schutzverifikation durchgeführt werden, als die beste Vorgehensweise [Ecke03] [BSI03]. Zur Beurteilung der Effizienz des Schutzes ist der Ressourcenverbrauch des Schutzes zu betrachten. Der Verbrauch der Ressourcen Speicherplatz und Bandbreite lässt sich anhand statischer Analysen feststellen. Der Speicherplatzbedarf eines Moduls wird von der Compiler-Linker-Locator-Werkzeugkette ermittelt und ausgegeben. Der Bedarf an Bandbreite folgt unmittelbar aus der Länge des jeweiligen, in Nachrichten zu integrierenden Unterscheidungsmerkmals. Der Rechenzeitbedarf kann durch Ausführung der einzelnen Module auf der Zielplattform gemessen[11] werden.

Um die Wirksamkeit des Schutzes anhand von Penetrationstests bestimmen zu können, wurde ein Penetrationstestsystem für eingebettete Systeme entwickelt [Jin07] [Gutb07a], welches die Durchführung und Auswertung von Penetrationstests automatisiert und somit die Durchführung einer großen Anzahl solcher Tests ermöglicht. Das Penetrationstestsystem besteht aus zwei Teilen. Ein Softwarewerkzeug, das auf einem PC ausgeführt wird, der an das zu evaluierende System angebunden ist, initiiert Angriffe und zeichnet auf, ob der Schutz die Angriffe als solche erkannt hat. Um die Erkennung der Angriffe aufzeichnen zu können, besteht der zweite Teil des Penetrationstestsystems aus einem Modul, welches in die Feldgerätesoftware integriert wird. Dieses protokolliert die Aktionen des Schutzes und sendet diese an den PC-basierten Teil des Penetrationstestsystems. Durch einen Abgleich der aufgezeichneten Aktionen des Schutzes mit den Angriffen kann die Wirksamkeit des Schutzes festgestellt werden. Das in die Feldgerätesoftware integrierte Modul führt auch eine Messung der Rechenzeit der einzelnen Schutzfunktionen durch. Abbildung 8.3 stellt das Penetrationstestsystem für die Feldebene dar.

[11] Der Rechenzeitbedarf kann prinzipiell auch statisch aus Betrachtung der zur Ausführung eines Schutzmechanismus erforderlichen Prozessoroperationen ermittelt werden. Die Komplexität der Schutzmechanismen und die umfangreichen Optimierungen, die moderne Compiler vornehmen, machen dies aber sehr aufwändig.

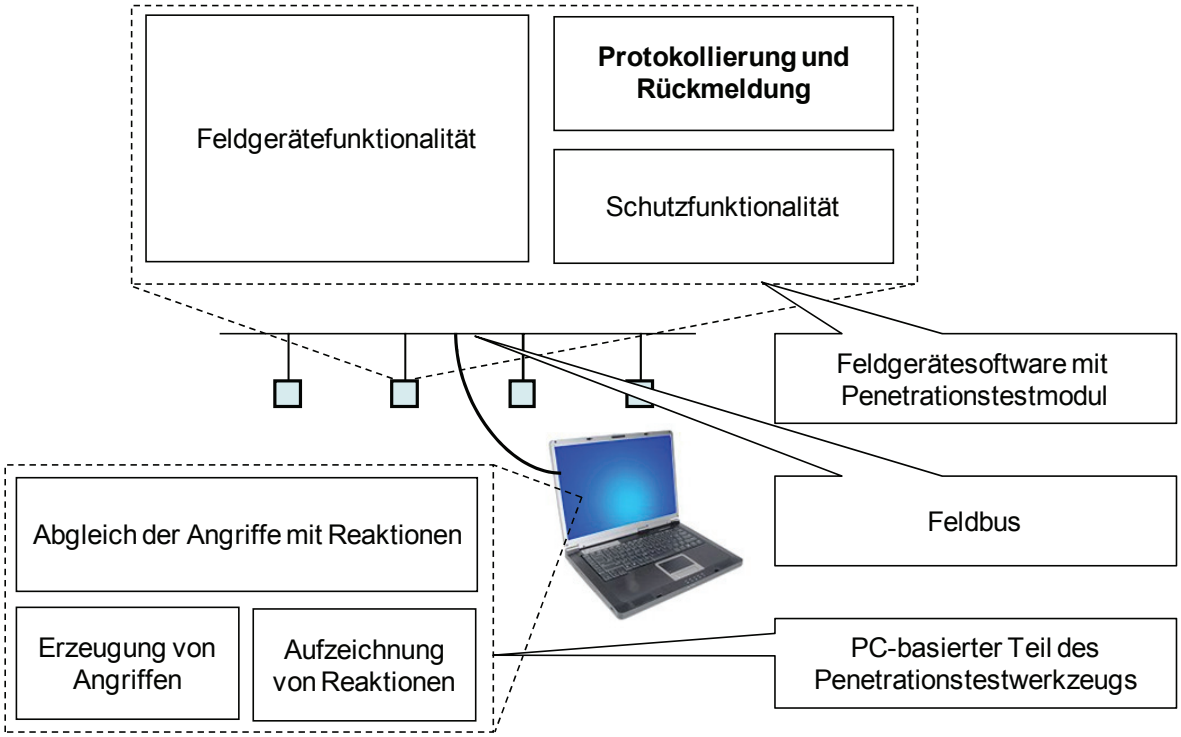

Abbildung 8.3: Integration des Penetrationssystemmoduls in die Feldgerätesoftware

In diesem Kapitel wurde eine Vorgehensweise beschrieben, anhand derer die zum Schutz der IT-Sicherheit der Feldebene eines Automatisierungssystems erforderlichen Schutzfunktionalitäten und Schutzmechanismen ausgewählt werden. Die Vorgehensweise orientiert sich an den im Security-Engineering üblichen Tätigkeiten und besteht aus einer Bedrohungsanalyse, einer Risikobewertung, einem Entwurf des Schutzes entsprechend den identifizierten relevanten Bedrohungen sowie einer Prüfung des Schutzes in Bezug auf Effektivität und Effizienz. Im folgenden Kapitel wird die technische Realisierung des Schutzes dargestellt. Im darauffolgenden Kapitel 10 wird der Schutz der IT-Sicherheit auf der Feldebene anhand der in diesem Kapitel beschriebenen Vorgehensweise bei zwei unterschiedlichen Automatisierungssystemen angewendet und gegen die in Abschnitt 5.1 hergeleiteten Anforderungen validiert.

9 Realisierung des Schutzes

Während in den vorangegangenen Kapiteln die Konzeption des Schutzes beschrieben wurde, behandelt dieses Kapitel die Realisierung des Schutzes. Das wichtigste Ziel des vorgestellten Konzeption ist die Effizienz des Schutzes, welche durch strikte Modularisierung und die Auswahl sowie die Verknüpfung der erforderlichen Module erreicht wird. Die Eigenschaften Modularisierung und Verknüpfbarkeit müssen sich daher auch in der Art und Weise der Realisierung widerspiegeln. Um eine hohe Effizienz des Schutzes im Betrieb zu erreichen, darf die zur Realisierung heranzuziehende Technologie keinen „Overhead“ erfordern, der den Ressourcenbedarf des Schutzes erhöhen würde. Ebenso soll die Konstruktion des Schutzes für die Feldebene eines konkreten Automatisierungssystems effizient, d. h. mit geringem Aufwand durchführbar sein. Dennoch ist eine hohe Qualität der Schutzsoftware sicherzustellen, um Angreifern keine Ansatzpunkte für Manipulationen zu liefern. Damit der Schutz von spezifischen Feldgerätestandards unabhängig ist, muss sichergestellt werden, dass die Schutzsoftware auf unterschiedlichen Feldgerätetypen lauffähig ist.

9.1 Softwarekomponenten für Feldgeräte

Eingebettete Software wird üblicherweise in der strukturierten Programmiersprache C [KeRi88] implementiert [LiYa03] [BeHa04], da in C hardwarenah und somit mit guter Ausnutzung der Ressourcen des Zielsystems programmiert werden kann und zudem aufgrund der Standardisierung der Sprache [ISO9899] für praktisch jeden am Markt verfügbaren Mikrocontroller entsprechende Übersetzungswerkzeuge vorliegen.

Software wird heutzutage zumeist nicht mehr „von Scratch“, sondern durch die Übernahme und die Anpassung bestehender Softwareteile realisiert („Copy, Paste and Modify“). Diese Vorgehensweise wird auch auf eingebettete C-Programme angewendet. Durch die Anpassung betriebsbewährter Softwareprogramme an einen neuen Einsatzkontext entstehen häufig Fehler [Dujm04]. Im Bereich der Standard-IT-Softwareentwicklung wurde dieses Dilemma durch die Einführung von Methoden wie der objektorientierten Softwareentwicklung [Balz99] und der komponentenbasierten Softwareentwicklung [Szyp98] verbessert. Diese Methoden ermöglichen einen hohen Grad an Wiederverwendung, wodurch Fehler bei der Anpassung reduziert werden. Daher sind solche Methoden vielversprechend, um eine hohe Softwarequalität zu erreichen. Sie lassen sich jedoch aufgrund ihres „Overheads“ und der geringen Ressourcen von Feldgeräten nicht auf der Feldebene einsetzen [GeHa03].

Um die Vorteile der Wiederverwendung von Software mit denen der Programmiersprache C zu verbinden und somit effiziente und qualitativ hochwertige eingebettete Software entwickeln zu können, wurde das Konzept der Strukturierten Komponenten erarbeitet [EbGö04a]. Strukturierte

Komponenten sind Softwarebausteine, welche funktional geschlossen und für Dritte unveränderlich sind. Dabei sehen Strukturierte Komponenten die Anpassung an unterschiedliche Randbedingungen vor und sind aufgrund einer strikten Trennung von Schnittstelle und Implementierung strukturell unabhängig.

Strukturierte Komponenten können zudem mit den verbreiteten Beschreibungsmitteln der Unified Modeling Language (UML) [OMG07] anschaulich dargestellt werden. Zur Überführung von UML-Modellen in C-Code wurde ein Idiom konzipiert, das Abbildungsvorschriften von UML-Beschreibungsmitteln auf C-Sprachmittel festlegt [EbGö04b]. Zudem wurde ein UML-Profil geschaffen, welches Stereotypen (z. B. «StructuredComponent» zur Identifikation einer *Strukturierten* Komponente) definiert und somit eine genauere Beschreibung der Modellelemente ermöglicht [GuWe05]. Abbildung 9.1 stellt eine Strukturierte Komponente C*omponentA* dar, welche auf eine andere Strukturierte Komponente C*omponentB* zugreift. Dazu ist *ComponentA* mit der Schnittstelle *InterfaceB* verknüpft, die von *ComponentB* implementiert wird.

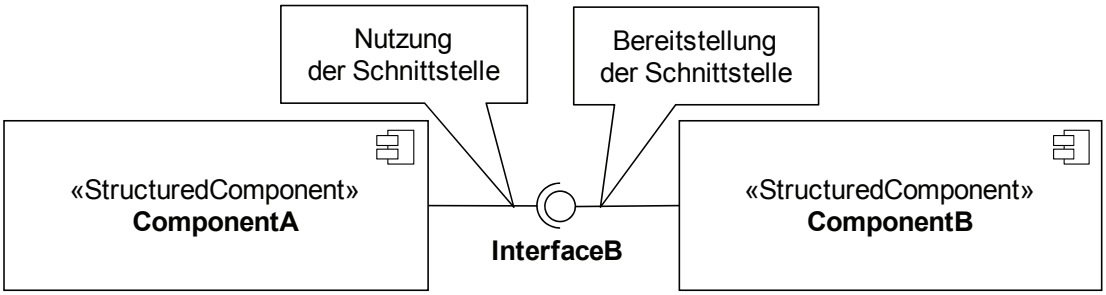

Abbildung 9.1: UML-Modellierung von Strukturierten Komponenten

Aufgrund der funktionalen Geschlossenheit von Strukturierten Komponenten lässt sich die geforderte Modularisierung des Schutzes mit ihnen umsetzen. Da sie für Dritte unveränderlich sind, können Fehler durch „Copy, Paste and Modify“ nicht auftreten. Dennoch erlaubt die Anpassbarkeit die Verwendung der gleichen Strukturierten Komponente unter unterschiedlichen Umgebungsbedingungen, und es können aufgrund der Trennung von Schnittstelle und Implementierung unterschiedliche Realisierungen zur Implementierung einer Schnittstelle herangezogen werden. Zudem können Strukturierte Komponenten in C realisiert werden und sich die Vorteile dieser Sprache in Bezug auf die Effizienz der Software zunutze machen. Aufgrund dieser Eigenschaften empfehlen sich Strukturierte Komponenten für die Realisierung des Schutzes der IT-Sicherheit auf der Feldebene. Daher werden die in Abschnitt 7.5.4 beschriebenen Softwarebausteine in Form von Strukturierten Komponenten umgesetzt.

9.2 Schnittstellen zur Spezifikation der Dienste der Schutzschichten und Schutzmechanismen

Das Schutzkonzept für die IT-Sicherheit auf der Feldebene fordert eine Entkopplung zwischen konkreten Softwarebausteinen, um beliebige Implementierungen von Schutzschichten und Schutzmechanismen kombinieren zu können. Dazu wurde eine abstrakte Schnittstelle gefordert, welche die Dienste definiert, die von einem die Schnittstelle implementierenden Softwarebaustein realisiert werden müssen. Strukturierte Komponenten erfüllen diese Anforderung durch ihre expliziten, von der Implementierung getrennten Schnittstellen, welche von mehreren Strukturierten Komponenten implementiert werden können. Daher sind entsprechende Schnittstellen für Strukturierte Komponenten zu spezifizieren, welche die Dienste der Schutzschichten und Schutzmechanismen zur Verfügung stellen.

9.2.1 Schnittstellen der Schutzschichtkomponenten

Die unterschiedlichen Schutzschichten realisieren eine Integration von verschiedenen Unterscheidungsmerkmalen in Nachrichten bzw. Programmcodes sowie eine Überprüfung dieser Unterscheidungsmerkmale. Somit ist zur Bereitstellung dieser Funktionen eine einzige Schnittstelle ausreichend, welche von allen Schutzschichtkomponenten implementiert werden kann. Im Fall der Überprüfung müssen die jeweiligen Funktionen das Prüfergebnis an die aufrufende Schicht bzw. an den für die Reaktion auf detektierte Angriffe verantwortlichen Softwarebaustein zurückgeben. Die eigentliche Generierung der Unterscheidungsmerkmale findet durch die Schutzmechanismen statt. Daher benötigen die Schutzschichtkomponenten auch die Möglichkeit, Schnittstellen zu nutzen, welche den Zugriff auf Schutzmechanismuskomponenten ermöglichen. Abbildung 9.2 stellt die Schnittstellen einer Schutzschichtkomponente dar.

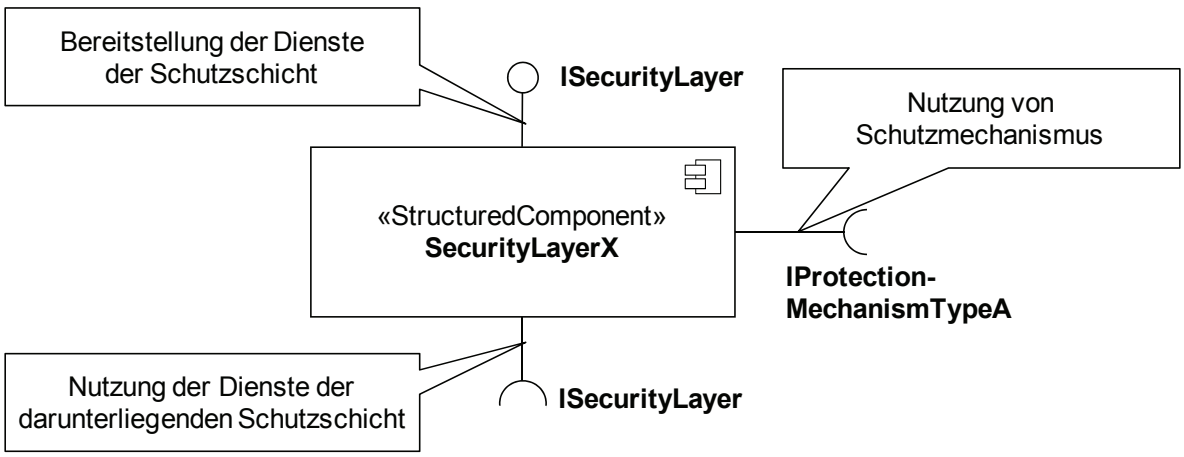

Abbildung 9.2: Schnittstellen einer Schutzschichtkomponente

Somit erfordert die Schnittstelle *ISecurityLayer* zwei Funktionen: eine zur Integration von Unterscheidungsmerkmalen (*Integrate*) und eine zur Überprüfung von Unterscheidungsmerkmalen (V*erify*). Beide Funktionen erfordern eine Referenz auf eine Nachricht bzw. Programmcode (*artifact*). Die Funktion V*erify* muss zudem ein Überprüfungsergebnis zurückliefern. Dazu wird ein numerischer Wert (des Typs *u8_t*) verwendet, welcher die Schutzschichten identifiziert, in denen Manipulationen detektiert wurden. Abbildung 9.3 stellt die Schnittstelle *ISecurityLayer* in UML-Notation dar.

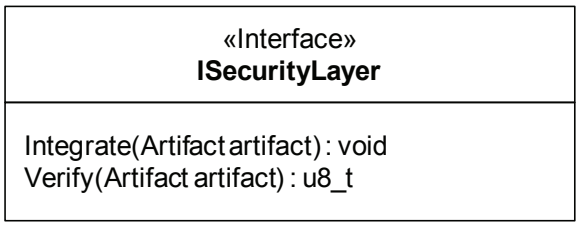

Abbildung 9.3: Schnittstelle *ISecurityLayer* in UML-Notation

Die Verwendung einer einzigen Schnittstelle zum Zugriff auf alle Schutzschichten ist zwar elegant, weist aber einen Nachteil auf. Es ist syntaktisch möglich, beliebige Schutzschichtkomponenten miteinander zu verknüpfen. Wird die Verknüpfung der Schutzschichtkomponenten jedoch nicht entsprechend der in Abschnitt 7.5.2 beschriebenen strikten, linearen Ordnung durchgeführt, ist der Schutz der IT-Sicherheit nicht gegeben. Aus diesem Grund wird für jede Schutzschicht jeweils eine eigene Schnittstelle zur Verfügung gestellt (*ISecurityLayer1*, *...*, *ISecurityLayer5*), welche die Funktionen *Integrate* und *Verify* zur Verfügung stellt. Abbildung 9.4 stellt dies beispielhaft anhand der Schnittstellen von Schutzschicht 3 dar (ohne die Schnittstelle zu einer Schutzmechanismuskomponente). Da die bei einer Verknüpfung von Strukturierten Komponenten beteiligten Schnittstellen anhand ihres Typs vom C-Compiler identifiziert werden, ist die Übersetzung von Quellcode mit unkorrekten Verknüpfungen somit nicht möglich.

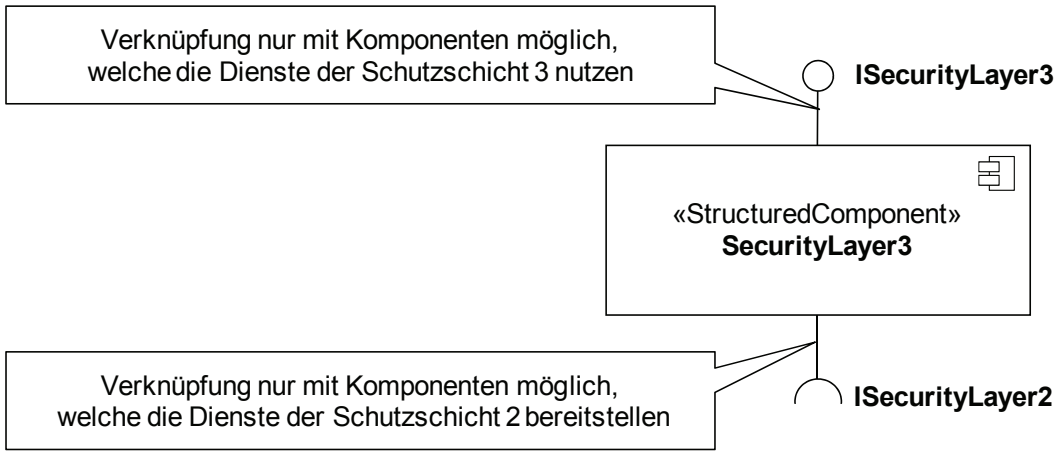

Abbildung 9.4: Schnittstellen der Schutzschicht 3

Die Einführung zusätzlicher Schnittstellen widerspricht nicht der Forderung nach geringem Ressourcenverbrauch, da die Schnittstellen im C-Code durch Funktionszeiger realisiert sind, deren

Nutzung sich im ausführbaren Programm jeweils als Sprungbefehl zu der entsprechenden Speicheradresse der aufzurufenden Funktion manifestiert. Somit belegt die Einführung unterschiedlicher Schnittstellen für die Schutzschichten keinen zusätzlichen Speicherplatz.

9.2.2 Schnittstellen der Schutzmechanismuskomponenten

In Abschnitt 8.4.2 wurde beschrieben, dass unter bestimmten Umständen andere als die vorgeschlagenen Schutzmechanismen verwendet werden müssen. Daher abstrahieren die Schnittstellen der Schutzmechanismen von der konkreten Funktionsweise der Schutzmechanismen und ermöglichen so die Verknüpfung unterschiedlicher Schutzmechanismuskomponenten mit der entsprechenden Schutzschichtkomponente. Analog zu den Schnittstellen zur Verknüpfung der Schutzschichten untereinander wird somit auch sichergestellt, dass eine Schutzschichtkomponente nicht mit einer Schutzmechanismuskomponente verknüpft werden kann, die auf der entsprechenden Schutzschicht nicht verwendet werden darf. Der Aufbau der Schnittstelle von Schutzmechanismuskomponenten wird im Folgenden anhand der Hashwertberechnung exemplarisch dargestellt.

Um die zur Berechnung des Hashwerts benötigten Variablen auf korrekte Startwerte zu setzen, müssen Hashalgorithmen zunächst initialisiert werden. Daraufhin können beliebig viele, unterschiedlich lange Datenblöcke in die Berechnung des Hashwerts einbezogen werden. Schließlich muss die Erzeugung des Hashwerts fertiggestellt und der Hashwert an den Aufrufer zurückgegeben werden. Daraus ergeben sich für die Schnittstelle *IHash* die Funktionen *Init* zur Initialisierung des Algorithmus, *Update* zum Einbeziehen von Daten und *Finish* zur Fertigstellung des Hashwerts.

9.3 Softwarekomponenten zur Implementierung der Dienste

9.3.1 Schutzschichtkomponenten

Die Schutzschichtkomponenten realisieren die Operationen, welche auf den einzelnen Ebenen der Schichtenarchitektur angesiedelt sind, d. h., sie integrieren bzw. überprüfen unterschiedliche Unterscheidungsmerkmale. Zur Erzeugung dieser Unterscheidungsmerkmale werden zudem jeweils unterschiedliche Informationen herangezogen, welche an verschiedenen Stellen in einer Nachricht bzw. im Programmcode abgelegt sind. Die Schutzschichtkomponenten müssen daher diese Informationen akquirieren und den jeweiligen Schutzmechanismen zur Verfügung stellen. Im Fall der Integration von Unterscheidungsmerkmalen müssen die Schutzschichtkomponenten das von den Schutzmechanismen erzeugte Unterscheidungsmerkmal integrieren; im Fall der Überprüfung ist das bereits vorhandene Unterscheidungsmerkmal mit dem erzeugten zu vergleichen.

Die Positionen, an denen sich die erforderlichen Informationen befinden, sind von dem verwendeten Feldbustyp bzw. Feldgerätetyp und dem Format der Daten in der Applikationsebene des ISO/OSI-Referenzmodells abhängig. Daher sind die Schutzschichtkomponenten entsprechend parametrierbar ausgelegt.

Dies soll im Folgenden beispielhaft anhand der Schutzschicht 2 dargestellt werden. In der Schutzschicht 2 werden kryptografische Hashwerte als Unterscheidungsmerkmal verwendet. Zur Berechnung des Hashwerts müssen somit die Speicherorte sowie die Länge der zu schützenden, also in den Hashwert einzubeziehenden Informationen parametriert werden können. Ebenso müssen der Speicherort sowie die Länge des Hashwerts in der Nachricht eingestellt werden können.

Daraus ergeben sich die folgenden Parameter: die Anzahl der zusammenhängenden Informationsblöcke, welche in den Hashwert eingehen sollen (*HASH_DATA_COUNT*), sowie deren Speicherort und Länge, welche jeweils in einer Struktur *HASH_DATA_INFO* des Typs *THashDataInfo* zusammengefasst sind. Schließlich werden noch der Speicherort des Hashwerts in der zu sichernden Nachricht und dessen Länge benötigt. Diese Informationen sind in der Struktur *HASH_VALUE_INFO* des Typs *THashValueInfo* zusammengefasst. Abbildung 9.5 stellt die entsprechende Schutzschichtkomponente mit ihren Parametern dar.

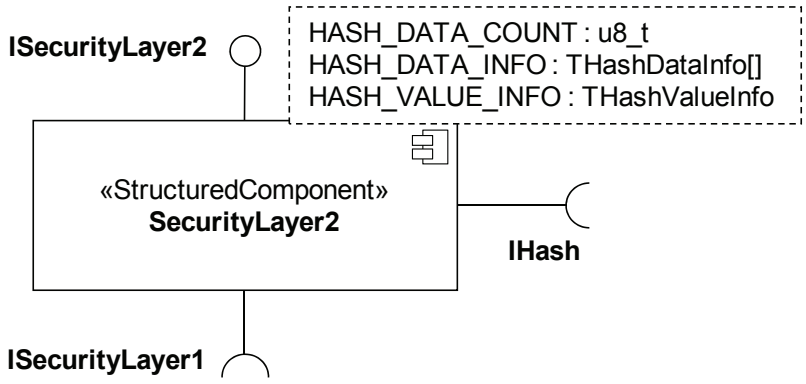

Abbildung 9.5: Strukturierte Komponente der Schutzschicht 2 mit Parametern

In Abbildung 9.6 ist die Integration eines Unterscheidungsmerkmals in eine Nachricht auf der Schutzschicht 2 dargestellt. Die Funktion *SecurityLayer2Integrate* kann über die Schnittstelle *ISecurityLayer2* aufgerufen werden und erhält die abzusichernde Nachricht als Parameter. Zunächst wird dann die mit dieser Komponente verknüpfte Hashfunktion initialisiert (1). Daraufhin werden alle zu schützenden Informationen unter Nutzung der eingestellten Parameter in die Hashwertberechnung einbezogen (2). Danach wird der Hashwert erzeugt und abgefragt (3) sowie an die korrekte Stelle in die Nachricht eingefügt (4). Schließlich wird der Dienst der darunterliegenden Schutzschicht über die Schnittstelle *ISecurityLayer1* aufgerufen. Der C-Quellcode der Funktion *SecurityLayer2Integrate* ist in Anhang A.2 angegeben.

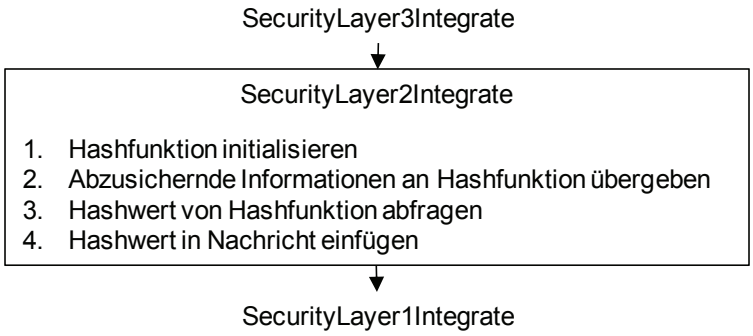

Abbildung 9.6: Ablauf der Integration eines Unterscheidungsmerkmals (Schutzschicht 2)

9.3.2 Schutzmechanismuskomponenten

Die Schutzmechanismuskomponenten realisieren die Funktionen der Schutzmechanismen. Dazu implementieren sie eine Schnittstelle, welche die für die Nutzung der jeweiligen Schutzmechanismusklasse erforderlichen Funktionen enthält. Beispielsweise realisiert die Strukturierte Komponente *Md5* die kryptografische Hashfunktion *Message Digest 5* (MD5) und stellt deren Funktionalität über die Schnittstelle *IHash* zur Verfügung. Abbildung 9.7 stellt die Strukturierte Komponente *Md5* in UML-Notation dar.

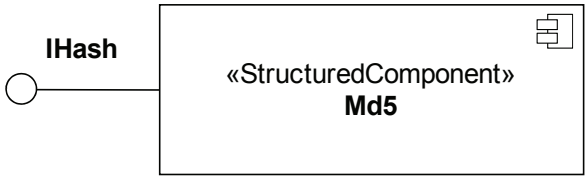

Abbildung 9.7: Schutzmechanismuskomponente *Md5*

Im Rahmen dieser Arbeit wurden die in Abschnitt 8.4.2 vorgeschlagenen Schutzmechanismen in Form von Strukturierten Komponenten realisiert. Um bewerten zu können, wie sich die unterschiedlichen Schutzstärken auf den Ressourcenkonsum der Komponenten auswirken, wurden darüber hinaus weitere Schutzmechanismen auch aus anderen Schutzmechanismusklassen realisiert. Tabelle 9.1 führt die realisierten Schutzmechanismuskomponenten auf.

Tabelle 9.1: Realisierte Schutzmechanismuskomponenten

Name	**Schutzmechanismus**	**Schutzmechanismus-klasse**	**Schutz-stärke**
Aes	Advanced Encryption Standard (AES): Schlüssellänge 128 Bit [FIPS197]	Symmetrische, blockorientierte Verschlüsselung	Mittel
3Des	Triple Data Encryption Standard (3DES) [FIPS46-3]	Symmetrische, blockorientierte Verschlüsselung	Schwach
Twofish	Twofish, Schlüssellänge: 128 Bit [SKW+98]	Symmetrische, blockorientierte Verschlüsselung	Mittel
Otp	One Time Pad [MvOV96]	Symmetrische, stromorientierte Verschlüsselung	Hoch
Rc4	Ron's Code 4 (RC4) [Ster94]	Symmetrische, stromorientierte Verschlüsselung	Mittel
Rsa	Rivest, Shamir, Adleman (RSA), Schlüssellänge: 1024 Bit [MvOV96]	Asymmetrische Verschlüsselung	Mittel
Md5	Message Digest 5 (MD5) [RFC1321]	Kryptografische Hashfunktion	Schwach
Sha1	Secure Hash Algorithm 1 (SHA1) [RFC3174]	Kryptografische Hashfunktion	Mittel
Sha256	Secure Hash Algorithm 2 (SHA-256) [FIPS180-2]	Kryptografische Hashfunktion	Hoch
VarId	Numerisches Identifikationsmerkmal variabler Länge [BMB+05]	Identifikationsmerkmal	Schwach – Hoch
UnixTime[12]	Echtzeitstempel mit variabler Auflösung [Gräf03]	Zeitmarkierung	Schwach – Hoch
SeqNum	Sequenznummernzähler [BMB+05]	Zeitmarkierung	Schwach
Code-Verifier	Verteilte Codeüberwachung [Gutb07b]	Anomaliedetektion	Schwach – Hoch

[12] Die Komponente *UnixTime* enthält einen plattformabhängigen Teil für den Zugriff auf einen Timer des verwendeten Mikrocontrollers.

9.4 Ressourcenbedarf der Softwarekomponenten

Der Ressourcenbedarf der Strukturierten Komponenten lässt sich in drei Kategorien einteilen: Speicherplatz-, Rechenzeit- und Bandbreitenbedarf. Dabei wird der jeweilige Bedarf von unterschiedlichen Faktoren bestimmt. Um von der Zielplattform unabhängig zu sein, werden Strukturierte Komponenten nicht als ausführbare Programme, sondern stets in Form von C-Code zur Verfügung gestellt. Dadurch hängt der Verbrauch des Speicherplatzes sowohl von der vorliegenden Implementierung als auch von dem verwendeten Compiler und dessen Einstellungen ab. Da bei Mikrocontrollersystemen unterschiedliche Arten von Speichern (RAM und ROM) existieren, die üblicherweise unterschiedliche Größen aufweisen, wird bei der Angabe des Speicherplatzes zwischen variablen Daten, die im RAM abgelegt werden müssen, und konstanten Daten sowie Programmcode, welche üblicherweise im ROM abgelegt werden, unterschieden.

Die erforderliche Rechenzeit ist außer der Implementierung und dem verwendeten Compiler auch von der Rechengeschwindigkeit der Zielplattform abhängig. Die Bandbreite, d. h. die Datenmenge, welche ein von einem Schutzmechanismus erzeugtes Unterscheidungsmerkmal in einer abzusichernden Nachricht belegt, ist dagegen ausschließlich durch die Spezifikation des Schutzmechanismus festgelegt. Tabelle 9.2 fasst die Faktoren zusammen, die den Ressourcenbedarf der Softwarekomponenten bestimmen.

Tabelle 9.2: Ressourcenbedarfsbestimmende Faktoren

Ressourcenbedarf	**Abhängig von**
Bandbreite	Spezifikation des Schutzmechanismus (unabhängig von konkreter Implementierung)
Speicherplatz	Spezifikation und Implementierung des Schutzmechanismus Compiler und Compilereinstellungen
Rechenzeit	Spezifikation und Implementierung des Schutzmechanismus Compiler und Compilereinstellungen Rechengeschwindigkeit der Zielplattform

Aufgrund der dargestellten Abhängigkeiten kann der Speicherplatz- und Rechenzeitbedarf der Strukturierten Komponenten nicht allgemeingültig angegeben werden. Die im Folgenden aufgeführten erforderlichen Ressourcen beziehen sich auf eine Kompilierung mit dem Tasking Embedded Development Environment Version 2.3, wobei alle möglichen Optimierungsoptionen deaktiviert sind. Als Zielplattform wurde ein am IAS entwickeltes Mikrocontroller-Board verwendet, das IAS-WebBoard (Rev. 1.2). Das IAS-WebBoard ist mit einem 16-bit-Mikrocontroller des Typs *Renesas M16C/62P* [RM16C06] ausgestattet, der eine Taktfrequenz von 24 MHz und eine Speicherkapazität von 384 kB (ROM) bzw. 31 kB (RAM) aufweist. Abbildung 9.8 stellt das IAS-WebBoard dar.

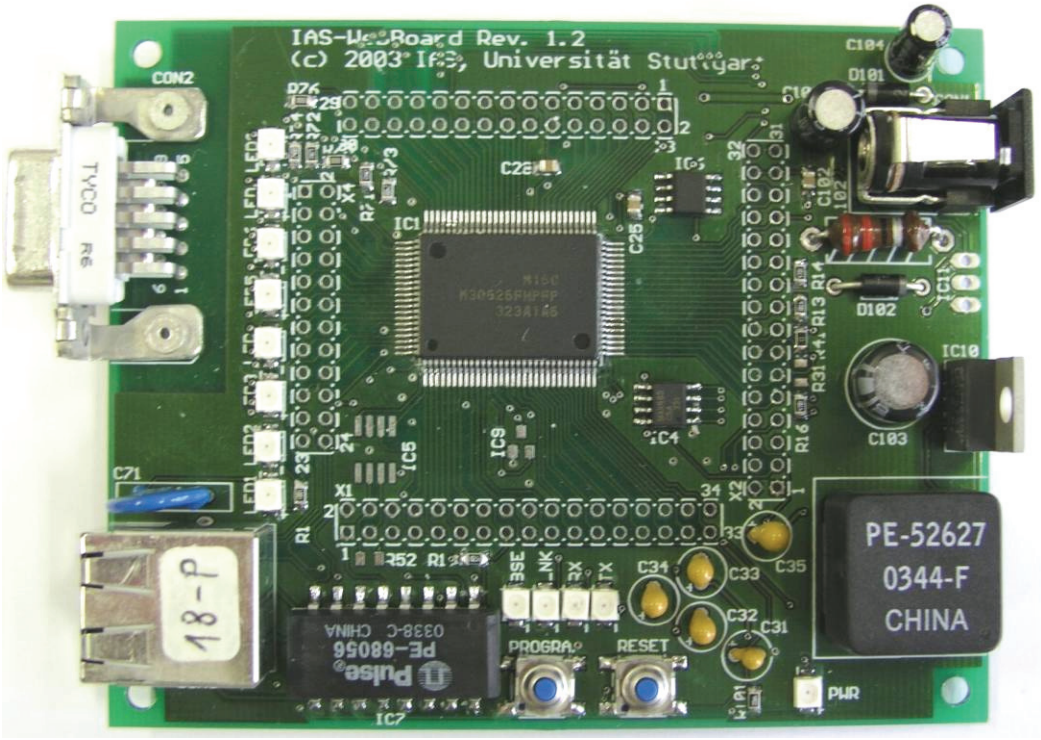

Abbildung 9.8: IAS-WebBoard mit Mikrocontroller Renesas M16C/62P

9.4.1 Schutzschichtkomponenten

In diesem Abschnitt wird der Ressourcenkonsum der Schutzschichtkomponenten dargestellt. Die Rechenzeiten der Schutzschichtkomponenten (ohne die Rechenzeiten, die zur Ausführung der mit den Schutzschichtkomponenten verknüpften Schutzmechanismuskomponenten benötigt werden) sowie ihr Speicherplatzbedarf sind in Tabelle 9.3 aufgeführt.

Tabelle 9.3: Erforderliche Rechenzeit und Speicherplatzbedarf der Schutzschichtkomponenten

Strukturierte Komponente	**Rechenzeit [ms]**	**Code (ROM) [Byte]**	**Konstante Daten (ROM) [Byte]**	**Variable Daten (RAM) [Byte]**
SecurityLayer1	0,095	656	28	0
SecurityLayer2	0,105	478	16	0
SecurityLayer3	0,020	304	16	0
SecurityLayer4	0,022	288	20	0
SecurityLayer5	0,259	343	16	208

Da die erforderliche Bandbreite nur von den verwendeten Schutzmechanismen, nicht aber von den Schutzschichtkomponenten abhängig ist, werden in diesem Abschnitt keine Bandbreitenbetrachtungen durchgeführt. Die Bandbreiten der Schutzmechanismen sind im folgenden Abschnitt aufgeführt.

9.4.2 Schutzmechanismuskomponenten

Dieser Abschnitt beschreibt den Ressourcenbedarf der Schutzmechanismuskomponenten. Dabei wird der Speicherplatzbedarf, der Rechenzeitbedarf und der Bandbreitenverbrauch dargestellt. Tabelle 9.4 enthält den erforderlichen Speicherplatz der Schutzmechanismuskomponenten.

Tabelle 9.4: Speicherplatzbedarf der Schutzmechanismuskomponenten

Strukturierte Komponente	**Code (ROM) [Byte]**	**Konstante Daten (ROM) [Byte]**	**Variable Daten (RAM) [Byte]**
Aes	2640	2648	0
3Des	42147	2176	28
Twofish	76667	518	5383
Otp	210	20	variabel[13]
Rc4	755	20	0
Rsa	8112	20	1608
Md5	10666	24	64
Sha1	3444	44	20
Sha256	45928	24	64
VarId	472	48	20
UnixTime	772	40	20
SeqNum	110	8	12
CodeVerifier[14]	20013 / 4214	16 / 1024	64 / 4

Im Folgenden sind die erforderlichen Rechenzeiten der Schutzmechanismuskomponenten angegeben. Abbildung 9.9 stellt die Rechenzeiten der symmetrischen, blockorientierten Verschlüsselungsalgorithmen AES, 3DES und Twofish sowie die der symmetrischen, stromorientierten Algorithmen OTP und RC4 für Datenmengen zwischen 1 Byte und 100 Byte dar[15].

[13] Abhängig von Größe des Schlüsselspeichers
[14] Der Schutzmechanismus *CodeVerifier* besteht, wie in Abschnitt 7.3.5 dargestellt, aus zwei Teilen, die auf dem überwachten bzw. dem überwachenden Feldgerät eingesetzt werden.
[15] Die Rechenzeiten für Ver- und Entschlüsselung sind bei symmetrischen Verschlüsselungsalgorithmen jeweils identisch.

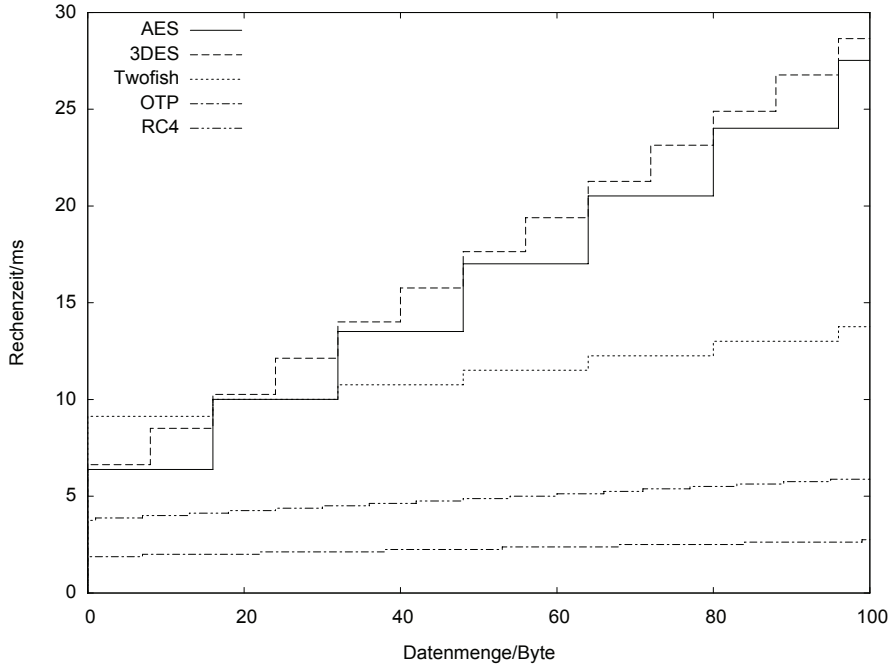

Abbildung 9.9: Rechenzeiten der symmetrischen Verschlüsselungsverfahren

Zum Vergleich zwischen symmetrischen und asymmetrischen Verschlüsselungsverfahren wurde auch das asymmetrische Verschlüsselungsverfahren RSA als Strukturierte Komponente realisiert. Die Rechenzeiten der Strukturierten Komponente *Rsa* sind im betrachteten Bereich (1 Byte bis 100 Byte) konstant. Im Unterschied zu den symmetrischen Verfahren sind die Rechenzeiten für Ver- und Entschlüsselung nicht identisch. In der Literatur (z. B. [MvOV96]) wird insbesondere die erforderliche Rechenzeit von asymmetrischer Verschlüsselung als wesentlich größer als diejenige von symmetrischer Verschlüsselung charakterisiert. Dies konnte anhand der gemessenen Werte bestätigt werden, die in Tabelle 9.5 aufgeführt sind.

Tabelle 9.5: Rechenzeiten des asymmetrischen Verschlüsselungsalgorithmus RSA

Strukturierte Komponente	**Rechenzeit Verschlüsselung [ms]**	**Rechenzeit Entschlüsselung [ms]**
Rsa	1470	156115

Abbildung 9.10 stellt die Rechenzeiten der realisierten kryptografischen Hashfunktionen MD5, SHA-1 und SHA-256 für Datenmengen zwischen 1 Byte und 100 Byte dar.

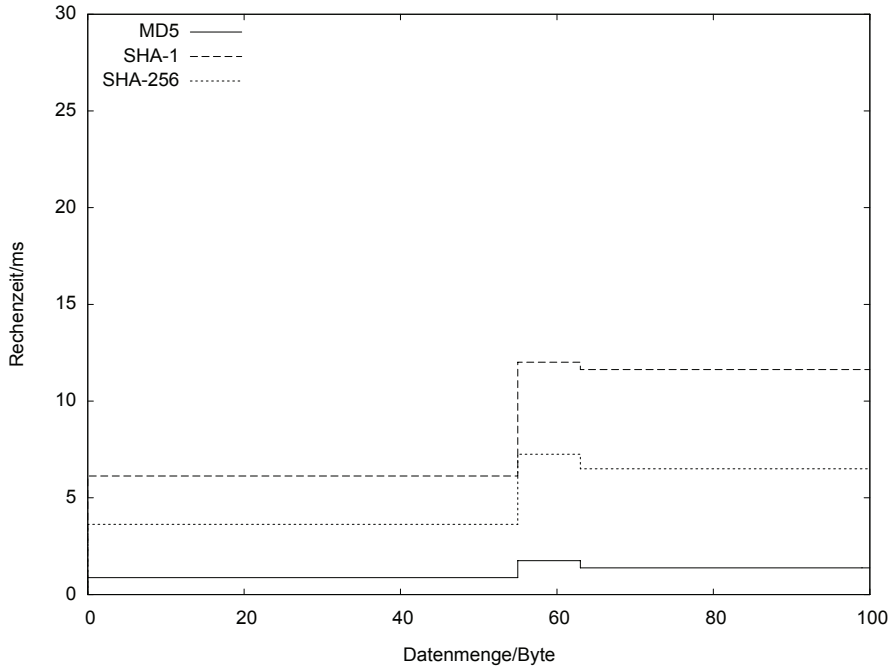

Abbildung 9.10: Rechenzeiten der Hashfunktionen

Die Erzeugung der Unterscheidungsmerkmale der Schutzschicht 3 (Identifikation, Rechteverwaltung) und der Schutzschicht 4 (Zeitstempel) ist von der Länge der abzusichernden Nachricht unabhängig. Tabelle 9.6 stellt die Rechenzeiten der entsprechenden Strukturierten Komponenten dar.

Tabelle 9.6: Rechenzeiten der Strukturierten Komponenten *VarId*, *UnixTime* und *SeqNum*

Strukturierte Komponente	**Rechenzeit [ms]**
VarId	0,102
UnixTime	0,022
SeqNum	0,021

Die Strukturierte Komponente *CodeVerifier* zur Code-Überwachung besteht, wie in Abschnitt 7.3.5 erläutert, aus zwei Teilen, die auf unterschiedlichen Feldgeräten ausgeführt werden. Die Zeit für die Ausführung des Teils, der auf einem überwachenden Feldgerät ausgeführt wird, beträgt stets 0,215 ms. Die Rechenzeit des Teils, der auf dem überwachten Feldgerät ausgeführt wird, ist von der Länge des überwachten Programmcodes abhängig. Abbildung 9.11 stellt diese Rechenzeiten dar.

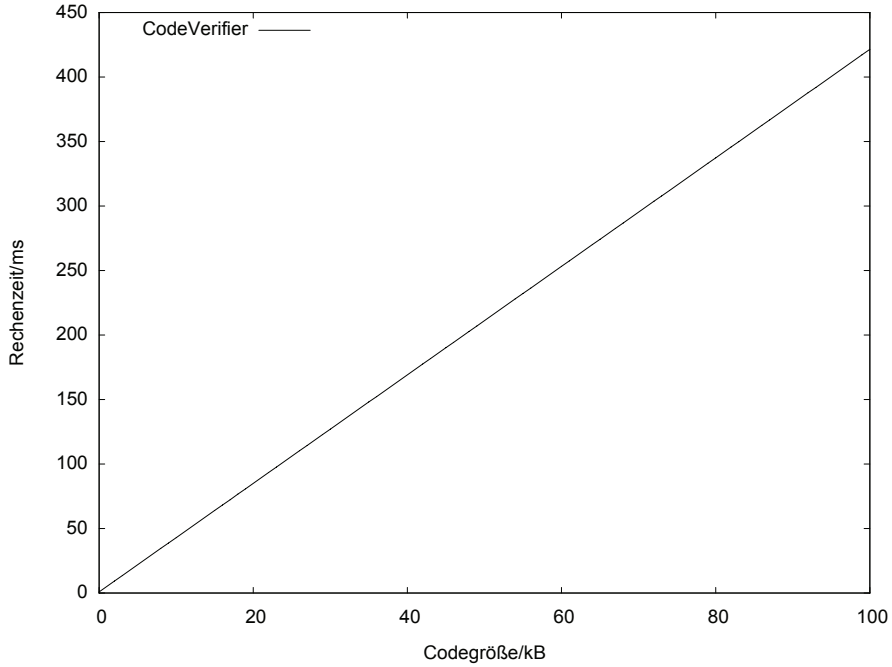

Abbildung 9.11: Variable Rechenzeit der Strukturierten Komponente *CodeVerifier*

Die zur Absicherung von Nachrichten erforderliche Bandbreite wird durch die Spezifikation der Schutzmechanismen bedingt. Die Bandbreiten der realisierten Schutzmechanismen sind in Tabelle 9.7 aufgeführt.

Tabelle 9.7: Bandbreitenbedarf der Schutzmechanismuskomponenten

Strukturierte Komponente	**Bandbreite [Byte]**	**Bemerkung**
Aes	0	Vorhandene Daten werden ersetzt.
3Des	0	Vorhandene Daten werden ersetzt.
Twofish	0	Vorhandene Daten werden ersetzt
Otp	0	Vorhandene Daten werden ersetzt.
Rc4	0	Vorhandene Daten werden ersetzt.
Rsa	0	Vorhandene Daten werden ersetzt.
Md5	16	-
Sha1	20	-
Sha256	32	-
VarId	variabel	Abhängig von der Anzahl unterschiedlicher Identitäten.
UnixTime	variabel	Abhängig von zeitlicher Auflösung (z. B. Auflösung 1 s erfordert 32 bit, Auflösung 1 ms entspricht 34 bit).
SeqNum	variabel	Abhängig von der Anzahl abzusichernder Nachrichten.
CodeVerifier	32	-

Die Ressourcenbetrachtungen bestätigen, dass es nicht den einen optimal für die Feldebene geeigneten Schutzmechanismus (innerhalb einer Schutzschicht) gibt. Die unterschiedlichen Schutzmechanismen haben jeweils Vor- und Nachteile bezüglich der unterschiedlichen Ressourcenkategorien und der Schutzstärke. So ist beispielsweise der Schutzmechanismus MD5 hochgradig rechenzeiteffizient und weist die geringste Bandbreite der Schutzmechanismen von Schutzschicht 2 auf. Andererseits gilt MD5 als die am wenigsten sichere der betrachteten kryptografischen Prüfsummen [WaYu05] [LuWe05] und benötigt mehr Speicherplatz als der sicherere Schutzmechanismus SHA-1.

9.5 Werkzeugunterstützung zur Modellierung und Codegenerierung

Mit den in den vorangegangenen Abschnitten vorgestellten Strukturierten Komponenten kann der Schutz der IT-Sicherheit auf der Feldebene in UML modelliert werden. Dazu sind die erforderlichen Schutzschichtkomponenten sowie geeignete Schutzmechanismuskomponenten auszuwählen. Die Schutzschichtkomponenten sind zu parametrieren sowie untereinander und mit den Schutzmechanismuskomponenten zu verknüpfen. Ausgehend von dem erstellten Modell muss C-Code geschrieben werden, der die Parameterwerte für die Stukturierten Komponenten verfügbar macht und die modellierten Verknüpfungen zwischen den Strukturierten Komponenten in Form von C-Code implementiert. Diese Transformationen sind in dem Idiom [EbGö04b] zur Abbildung von UML-Modellen von Strukturierten Komponenten in C-Code festgelegt.

Zur Unterstützung der Modellierung und zur Automatisierung der Code-Erzeugung wurde das Werkzeug *Component Development Environment* (CDE) entwickelt. Um Funktionalitäten wie Darstellung, Speicherung von grafischen Modellen sowie die C-Code-Generierung nicht von Grund auf neu entwickeln zu müssen, wurde das CDE-Werkzeug als Eclipse-Applikation [Ecli07] [GaBe04] realisiert. Dabei wurden unterschiedliche Frameworks herangezogen, welche für Eclipse zur Verfügung stehen: das Eclipse-Modeling-Framework (EMF) [EMF05] [BSM+04] zur Repräsentation der Modelle, das Graphical-Modeling-Framework (GMF) [GMF07] zur grafischen Darstellung der Modelle sowie die Java-Emitter-Templates (JET) [JET04] zur C-Code-Generierung. Abbildung 9.12 zeigt die Benutzungsoberfläche des Werkzeugs CDE.

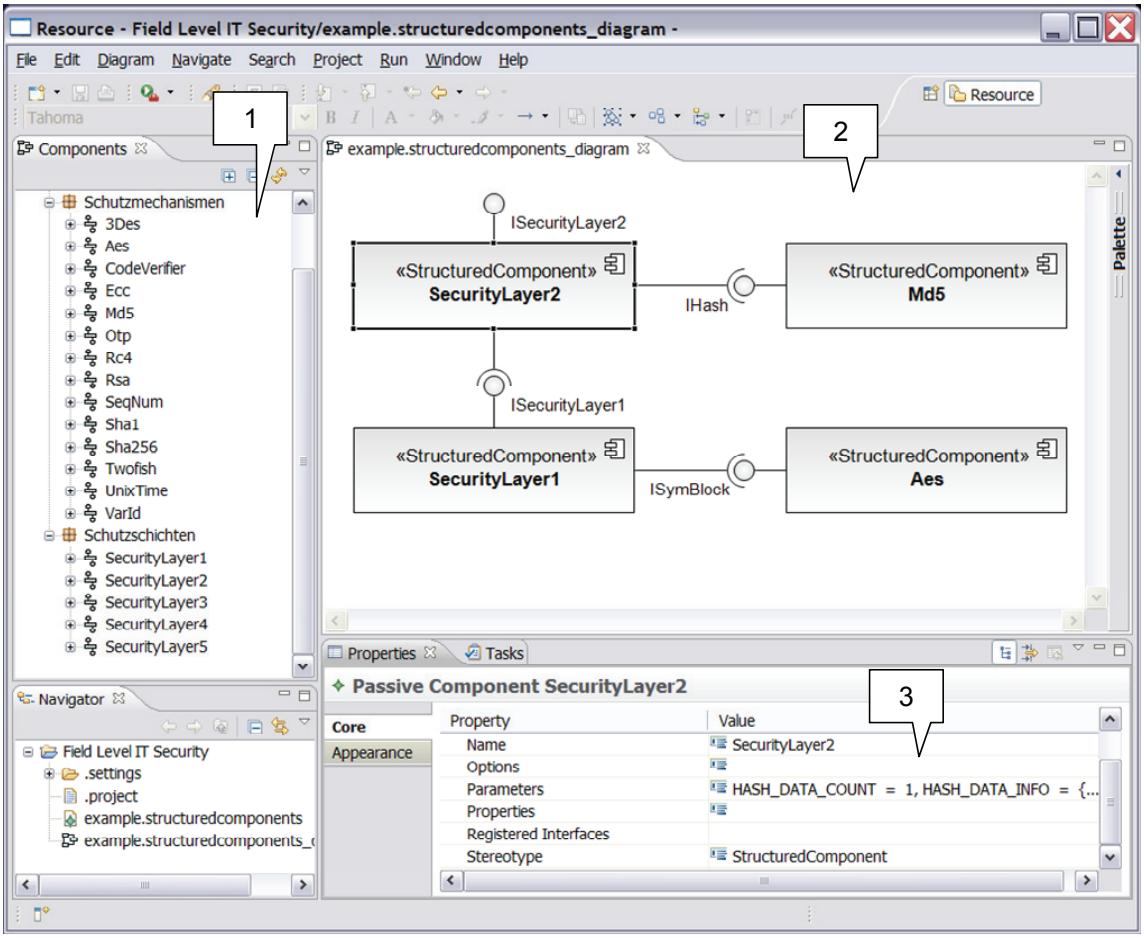

Abbildung 9.12: Benutzungsoberfläche des Werkzeugs CDE

In der Ansicht[16] *Components* (1) sind die verfügbaren Strukturierten Komponenten dargestellt. Im Editor-Bereich (2) werden die ausgewählten Strukturierten Komponenten und deren Schnittstellen dargestellt sowie die Verknüpfungen vorgenommen. In der Ansicht *Properties* (3) kann die Parametrierung der im Editor ausgewählten Strukturierten Komponente vorgenommen werden.

Bei der Codegenerierung werden zwei C-Header-Dateien generiert. Die Datei *link.h* stellt die Verknüpfungen zwischen den Strukturierten Komponenten her, die Datei *config.h* enthält die eingestellten Parameterwerte. Ausschnitte aus den generierten Dateien sind in Anhang A.3 dargestellt.

In diesem Kapitel wurde die Realisierung des Schutzes der IT-Sicherheit auf der Feldebene beschrieben. Um eine hohe Softwarequalität zu erreichen, wurde das Konzept der Strukturierten Komponenten ausgewählt, das es ermöglicht, bewährte Softwarebausteine wiederzuverwenden, diese miteinander zu verknüpfen und dabei den Overhead zu vermeiden, der bei vergleichbaren Konzepten im Standard-IT-Bereich existiert. Darauf aufbauend wurden Schnittstellen für Schutzschichtkomponenten und Schutzmechanismuskomponenten definiert. Schutzschichtkom-

[16] Eclipse stellt für Benutzungsoberflächen zwei unterschiedliche Elemente zur Verfügung: Ansichten (Views) und Editoren. Die jeweiligen Eigenschaften sind in [GaBe04, S. 221] beschrieben.

ponenten realisieren die Operationen, welche auf den einzelnen Schichten angesiedelt und unabhängig von konkreten Schutzmechanismen sind. Schutzmechanismuskomponenten realisieren die Operationen konkreter Schutzmechanismen, z. B. kryptographischer Algorithmen. Die konzipierten Komponenten sind dabei parametrierbar ausgelegt, damit sie an unterschiedliche Umgebungsbedingungen (wie den verwendeten Feldbusstandard) anpassbar sind. Die Ressourcen, die von den Komponenten benötigt werden, wurden mithilfe eines typischen Feldgeräts gemessen, welches einen 16-bit-Microcontroller des Typs *Renesas M16C* enthält. Dabei bewegt sich der Speicherverbrauch für die Schutzmechanismen zwischen 118 Byte und 77,2 kB (ROM) bzw. zwischen 0 Byte und 5,4 kB (RAM). Die Rechenzeiten hängen bei den meisten Schutzmechanismen von der Länge der zu schützenden Information ab. Die Dauer für die Absicherung einer Information mit einer Länge von 50 Byte beträgt bei keinem der Schutzmechanismen mehr als 20 ms[17]. Um aufwändige und repetitive Aufgaben bei der Realisierung des Schutzes zu automatisieren, wurde das Softwarewerkzeug *Component Development Environment* (CDE) eingeführt. Das Softwarewerkzeug CDE ermöglicht die Auswahl, Verknüpfung und Parametrierung der Strukturierten Komponenten sowie die C-Code-Generierung. Das folgende Kapitel stellt die Validierung des Konzepts anhand der Anwendung auf zwei Fallbeispiele dar.

[17] Der asymmetrische Verschlüsselungsalgorithmus RSA, welcher nur zu Vergleichszwecken zwischen symmetrischer und asymmetrischer Verschlüsselung implementiert wurde, ist bei dieser Angabe nicht berücksichtigt.

10 Fallbeispiele zur Validierung des Schutzes

In den Kapiteln 7–9 wurden das Konzept zum effizienten Schutz der IT-Sicherheit auf der Feldebene, eine Vorgehensweise zur Ermittlung von Schutzfunktionalitäten und Schutzmechanismen sowie die Realisierung des Schutzes vorgestellt. Im Rahmen dieses Kapitels wird nun geprüft, ob der Schutz die gestellten Anforderungen erfüllt. Dazu wird die Anwendung des Schutzes anhand von zwei Fallbeispielen dargestellt. Als zu schützende Automatisierungssysteme werden zwei Demonstratoren herangezogen: ein Netzwerk von Kfz-Steuergeräten und eine Fernsteuerung für einen industriellen Kaffeeautomaten. Zur Beschreibung der Anwendung des Schutzes wird zunächst der Aufbau dieser Demonstratoren skizziert, daraufhin werden Bedrohungen der IT-Sicherheit auf der Feldebene dieser Demonstratoren identifiziert und die dazugehörigen Risiken ermittelt. Ausgehend von diesen Risiken wird der Schutzentwurf durchgeführt und die Schutzsoftware generiert sowie auf die Demonstratoren aufgebracht. Zur Prüfung der Wirksamkeit des Schutzes werden Penetrationstests durchgeführt, d. h. Manipulationen, die so ausgeführt werden, wie dies auch bei einem realen Angriff der Fall wäre. Zur Bewertung der Effizienz des Schutzes wird der jeweilige Ressourcenverbrauch betrachtet.

10.1 Ziel der Validierung

Das Ziel der Validierung besteht darin, zu prüfen, ob das im Rahmen dieser Arbeit entwickelte Konzept geeignet ist, den Schutz der IT-Sicherheit auf der Feldebene zu gewährleisten. Dazu werden die in Abschnitt 5.1 hergeleiteten Anforderungen verwendet.

Es wird untersucht, ob der realisierte Schutz in der Lage ist, Übergänge des Automatisierungssystems in unerwünschte Fehlerzustände zu unterbinden. Um dies unter realistischen Bedingungen zu prüfen, werden mit einem Penetrationstestsystem Angriffe durchgeführt, die sich nicht von solchen Angriffen unterscheiden, die ein realer Angreifer ausführen würde.

Um zu prüfen, ob der Schutz an unterschiedliche Feldgeräte- und Feldbusstandards angepasst werden kann, kommen zwei Demonstratoren zum Einsatz, welche mit unterschiedlichen Mikrocontroller- und Feldbustypen aufgebaut sind. Diese Mikrocontrollertypen verfügen über wesentlich weniger Speicherplatz und Rechenkapazität als Standard-IT-Rechner. Die verwendeten Feldbussysteme weisen zum Teil deutlich weniger Kommunikationsbandbreite auf als Kommunikationssysteme der Standard-IT. Daher ist zu zeigen, dass die in Form von Software realisierten Schutzfunktionalitäten so effizient sind, dass sie zeitlich deterministisch sowie trotz der geringen Ressourcen in vertretbarer Dauer von den Feldgeräten ausgeführt werden können.

10.2 Fallbeispiel Kfz-Steuergerätenetzwerk

10.2.1 Aufbau des Kfz-Steuergerätenetzwerks

Die Funktionalitäten moderner Kraftfahrzeuge werden zunehmend durch programmierbare elektronische Steuergeräte (SG) realisiert, die mit Feldbussen untereinander vernetzt sind. Der Demonstrator *Kfz-Steuergerätenetzwerk* bildet einen solchen Verbund von Kfz-Steuergeräten nach. Dazu wird ein Modell eines Kraftfahrzeugs als technischer Prozess verwendet, welcher durch sieben Steuergeräte geführt wird. Dabei wird die Steuerung der Blinkerlampen, der Hupe, der Bremsen, der Scheinwerfer, des Motors und des faltbaren Dachs des verwendeten Modellfahrzeugs mit jeweils einem Steuergerät durchgeführt. Zur Bedienung der Kfz-Funktionen wird ein Armaturenbrett verwendet, das ebenfalls über ein Steuergerät verfügt. Zur Vernetzung der Steuergeräte kommen die Feldbussysteme CAN und Ethernet zum Einsatz. Zur Visualisierung wird zusätzlich noch ein Industrie-PC eingesetzt. Abbildung 10.1 stellt die Elemente des Demonstrators dar.

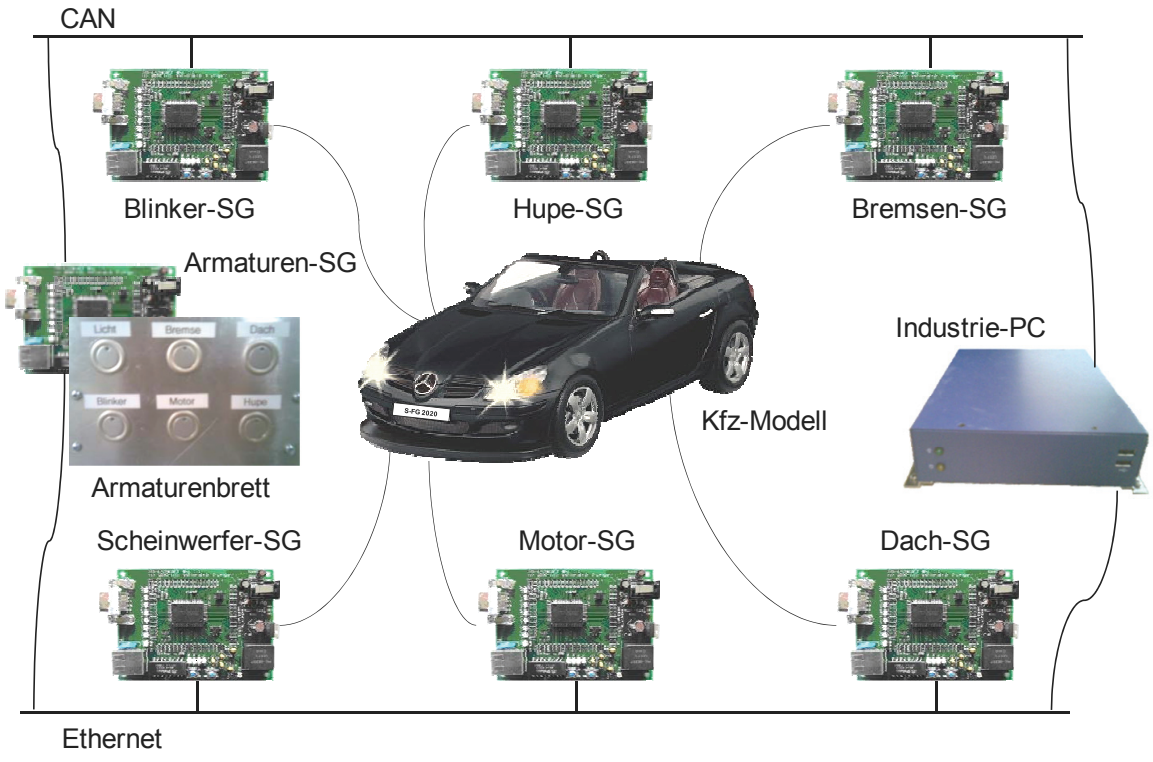

Abbildung 10.1: Aufbau des Demonstrators Kfz-Steuergerätenetzwerk

Der Demonstrator Kfz-Steuergerätenetzwerk wurde explizit zur Evaluierung des Konzepts zum Schutz der IT-Sicherheit auf der Feldebene realisiert. Daher verfügen die Steuergeräte über Anzeigeelemente, mit welchen die zur Verfügung stehenden Schutzschichten anhand von LEDs sowie Angriffe auf die IT-Sicherheit anhand von Blinkleuchten visualisiert werden können. Abbildung 10.2 stellt diese Anzeigeelemente dar.

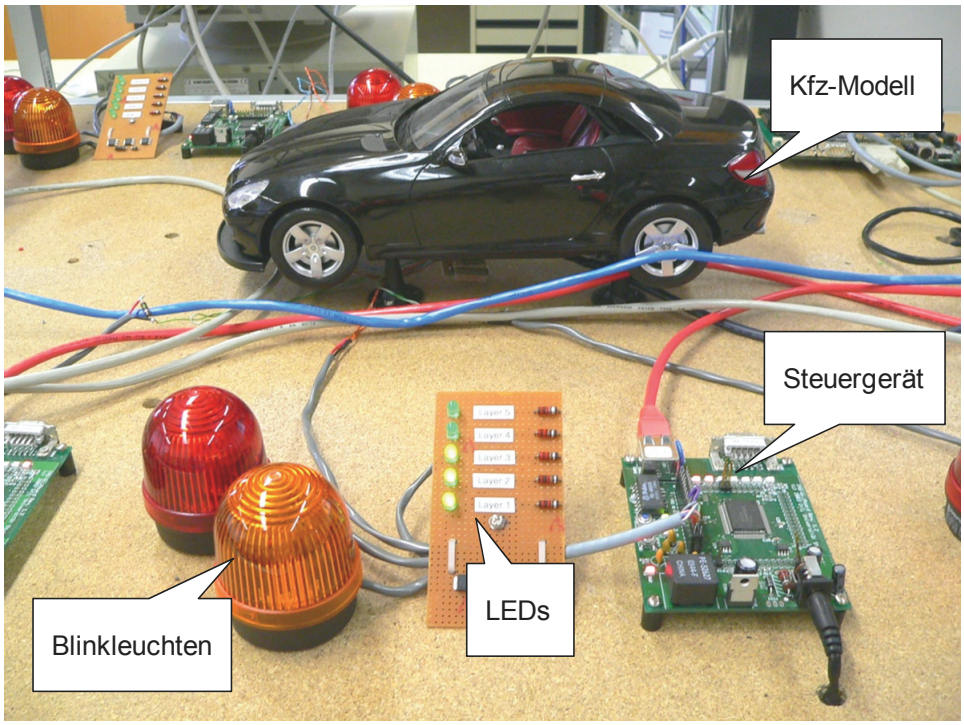

Abbildung 10.2: Anzeigeelemente der Steuergeräte

10.2.2 Bedrohungsanalyse und Risikobewertung

Aufgrund der hohen Diebstahlwahrscheinlichkeit bei Sportwagen [Masc08] wird ein Misuse-Case als Beispiel für die Bedrohungsanalyse und die Risikobewertung herangezogen, bei dem ein Dieb durch das Öffnen des Dachs Zugang zum Fahrzeuginneren erlangen möchte. Dabei werden die folgenden Annahmen zugrunde gelegt:

- Der Angreifer verfügt über die Fähigkeiten eines Crackers.
- Die Fahrgeschwindigkeit des Kfz weist während des Angriffs den Wert 0 km/h auf, d. h., das Fahrzeug wird nicht bewegt.
- Das Armaturensteuergerät befindet sich hinter dem Armaturenbrett im Fahrzeuginnenraum.

Der Elektromotor, welcher das Dach des Kfz-Modells zusammenfaltet und in den Kofferraum bewegt, wird vom Dachsteuergerät kontrolliert. Das Öffnen des Dachs kann von einem Bediener des Demonstrators mit einem Knopf auf dem Armaturenbrett ausgelöst werden, das vom Armaturensteuergerät kontrolliert wird. Da das Dach nur bei niedrigen Fahrgeschwindigkeiten bewegt werden darf, liest das Dachsteuergerät vor Beginn der Dachbewegung die momentane Fahrgeschwindigkeit vom Feldbus ein, auf den diese vom Motorsteuergerät regelmäßig gesendet wird. Abbildung 10.3 stellt die beteiligten Steuergeräte und die erforderlichen Nachrichten dar.

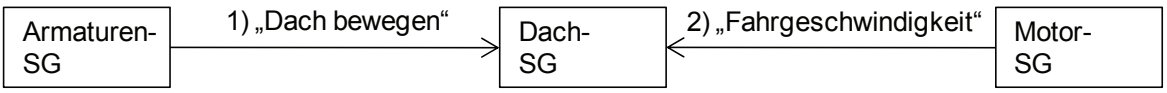

Abbildung 10.3: Kommunikationsvorgänge zum Auslösen der Dachöffnung

Führt man die in Abschnitt 8.2 beschriebene Prozedur zur Identifikation der abzusichernden Feldgerätefunktionalitäten und Feldbuskommunikationen durch (vgl. Anhang A.1), ergibt sich folgendes Ergebnis: Die Funktionalität, die zur Erreichung des Misuse-Case-Ziels ausgeführt werden muss, ist die *Dach-bewegen*-Funktionalität, welche von dem Dachsteuergerät ausgeführt wird. Zur Ausführung dieser Funktionalität werden die in den Feldbusnachrichten *Dach bewegen* sowie *Fahrgeschwindigkeit* enthaltenen Informationen benötigt. Diese Informationen werden wiederum durch Funktionalitäten erzeugt, die auf dem Armaturensteuergerät bzw. dem Motorsteuergerät ausgeführt werden. Diese Funktionen benötigen keine weiteren, über einen Feldbus übertragenen Informationen.

Daraus ergeben sich die Möglichkeiten für den Angreifer, den Dachöffnungsvorgang auszulösen. Durch die Veränderung der Funktionalität des Dachsteuergeräts kann die Dachöffnung herbeigeführt werden. Daher ist die Wahrscheinlichkeit für einen derartigen Angriff als hoch zu betrachten.

Mit einer Manipulation der *Dach-bewegen*-Nachricht kann das (unmanipulierte) Dachsteuergerät angewiesen werden, das Dach zu öffnen. Daher sind die Wahrscheinlichkeiten der unterschiedlichen Angriffsarten auf diese Nachricht zu betrachten. Ein Abhören hat keine unmittelbaren Auswirkungen auf das System, führt also nicht zu einem Öffnen des Dachs und ist daher unwahrscheinlich. Die inhaltliche Veränderung der Nachricht müsste genau zu dem Zeitpunkt stattfinden, in dem ein legitimer Nutzer das Dach schließt. Dies erscheint wenig wahrscheinlich. Durch das Erzeugen einer eigenen *Dach-bewegen*-Nachricht wird das Ziel des Angreifers erreicht, ebenso wie durch das Wiederholen einer legitimen *Dach-bewegen*-Nachricht. Daher erscheinen diese Angriffe als wahrscheinlich.

Die in der *Fahrgeschwindigkeit*-Nachricht enthaltene Information wird dazu verwendet, die Öffnung des Dachs bei hohen Geschwindigkeiten zu unterbinden. Bei dem betrachteten Misuse-Case befindet sich das Kfz jedoch in Ruhe, sodass diese Nachricht die Erreichung des Misuse-Case-Ziels nicht verhindern kann. Somit besteht keinen Anlass für den Angreifer, die Nachricht zu manipulieren, die Wahrscheinlichkeit für einen Angriff auf diese Nachricht ist also gering. Aus demselben Grund ist es auch nicht zielführend für den Angreifer, die Funktionalität des Motorsteuergeräts zu manipulieren, wodurch die Wahrscheinlichkeit für diesen Angriff als gering anzusehen ist.

Das Armaturensteuergerät befindet sich im Fahrzeuginneren. Um physisch auf dieses Steuergerät zugreifen zu können, muss sich der Angreifer ebenfalls im Fahrzeuginneren befinden. In die-

sem Fall wäre das Ziel des Angreifers bereits erreicht, die Manipulation des Armaturensteuergeräts somit nicht mehr erforderlich und daher auch nicht wahrscheinlich.

Tabelle 10.1 fasst die ermittelten Auswirkungen und Wahrscheinlichkeiten sowie die sich daraus ergebenden Risiken zusammen. Dabei wird aufgrund des für den Fahrzeughalter beträchtlichen finanziellen Schadens ein mittlerer Schweregrad (Stufe 2) der Auswirkung für die Öffnung des Dachs angenommen.

Tabelle 10.1: Risikobewertung der identifizierten Gefährdungen (Kfz-Steuergerätenetzwerk)

Bedrohung	**Auswirkung**	**Wahrscheinlichkeit**	**Risiko**
Funktionalitäten			
Armaturen-SG	Dachöffnung (2)	Gering (2)	4
Dach-SG	Dachöffnung (2)	Hoch (4)	8
Motor-SG	Keine (0)	Gering (2)	0
Kommunikation Armaturen-SG/Dach-SG			
Abhören	Keine (0)	Gering (2)	0
Manipulation inh. Integrität	Dachöffnung (2)	Gering (2)	4
Erzeugung eigener Nachrichten	Dachöffnung (2)	Hoch (4)	8
Manipulation zeitl. Integrität	Dachöffnung (2)	Hoch (4)	8
Kommunikation Dach-SG/Motor-SG			
Abhören	Keine (0)	Gering (2)	0
Manipulation inh. Integrität	Keine (0)	Gering (2)	0
Erzeugung eigener Nachrichten	Keine (0)	Gering (2)	0
Manipulation zeitl. Integrität	Keine (0)	Gering (2)	0

10.2.3 Entwurf und Realisierung des Schutzes

Das größte Risiko, den unerwünschten Zustand *Dach offen* zu erreichen, besteht bei der Manipulation der Dachsteuergerät-Funktionalität, der Erzeugung einer eigenen *Dach-bewegen*-Nachricht und der Manipulation der zeitlichen Integrität dieser Nachricht. Bei allen anderen möglichen Angriffen besteht maximal ein halb so großes Risiko. Daher sind die Funktionalität des Dachsteuergeräts sowie die Autorisierung und die zeitliche Integrität der *Dach-bewegen*-Nachricht zu schützen. Aus dem in Abschnitt 7.5.3 hergeleiteten Zusammenhang zwischen Schutzfunktionalitäten und Schutzschichten folgt, dass die Schutzschichten 1–5 auf dem Dach-

steuergerät und dem Armaturensteuergerät eingesetzt werden müssen[18]. Aufgrund der mittleren Fähigkeiten des Angreifers ist eine mittlere Stärke des Schutzes erforderlich (vgl. Abschnitt 3.2.3). Daher werden die in Abschnitt 8.4.2 vorgeschlagenen Schutzmechanismen verwendet.

Die Reaktion des Schutzes muss darin bestehen, das Dach des Fahrzeugs nicht zu öffnen, d. h., als manipuliert erkannte Nachrichten müssen von der Reaktionskomponente (*RoofReaction*) ausgefiltert werden und nicht an die Steuergeräteapplikation (*RoofControl*) weitergeleitet werden. Bei einer Manipulation der Funktionalität soll das Dachsteuergerät abgeschaltet werden. Zusätzlich soll die Reaktionskomponente Nachrichten an das Hupensteuergerät senden, um die Hupe des Kfz zur Signalisierung eines Einbruchsversuchs zu verwenden.

Die Reaktionskomponente muss zur Ausfilterung der manipulierten Nachrichten zwischen dem Feldbuskommunikationsstack (*Ethernet*) und der Steuergeräteapplikation angeordnet werden. Da die Reaktionskomponente für unterschiedliche Automatisierungssysteme aufgrund der jeweils erforderlichen Reaktionen individuell entwickelt werden muss, bietet sie sich zudem als Bindeglied zwischen den Detektionskomponenten mit ihren stets gleichen Schnittstellen und den Feldgeräteapplikationen und Kommunikationsstacks mit jeweils unterschiedlichen Schnittstellen an. Sie dient somit als Adapter [GHRV96] zur Integration des Schutzes in die Feldgerätesoftware. Daraus ergibt sich das in Abbildung 10.4 dargestellte Komponentenmodell.

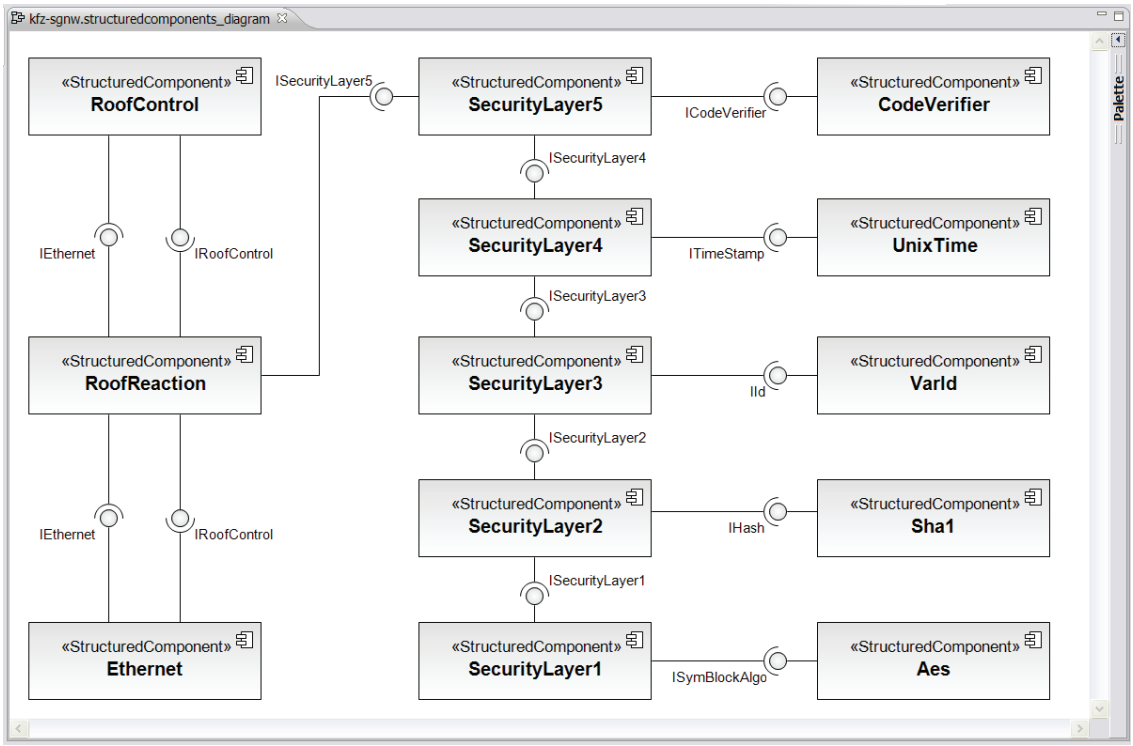

Abbildung 10.4: Komponentenmodell des Schutzes für das Dachsteuergerät

[18] Die Überwachung des Dachsteuergeräts mittels der Schutzschicht 5 kann prinzipiell von jedem anderen Steuergerät durchgeführt werden. Da zum Einsatz von Schutzschicht 5 die unteren vier Schutzschichten erforderlich sind, welche auf dem Armaturensteuergerät zur Absicherung der Kommunikation vorhanden sein müssen, bietet sich dieses Steuergerät zur Ausführung der Schutzschicht 5 an.

Die von den Schutzmechanismen erzeugten Unterscheidungsmerkmale müssen in die *Dach-bewegen*-Nachricht zwischen Armaturensteuergerät und Dachsteuergerät integriert werden. Die Nachricht ist entsprechend dem Ethernet-Frame-Format [IEEE802.3] aufgebaut und enthält Ziel- und Quell-MAC-Adressfelder zur Identifizierung von Empfänger und Absender, ein Typ-Feld (T) zur Bezeichnung des Formats, dem die transportierten Daten entsprechen, die Daten selbst, ein Padding-Feld, um eine Frame-Länge von mindestens 64 Byte sicherzustellen, und ein CRC-Feld zur Erkennung von Übertragungsfehlern. Bei der vorliegenden Anwendung kann die MAC-Adresse des Armaturensteuergeräts als Identifikationsmerkmal dienen, sodass das Quell-MAC-Adressfeld zur Aufnahme des entsprechenden Unterscheidungsmerkmals genutzt werden kann. Die restlichen Unterscheidungsmerkmale Zeitmarkierung und Hashwert werden nach den Nutzdaten (ND), welche die Anweisung zum Öffnen des Dachs bzw. das Überprüfungsergebnis der verteilten Code-Überwachung enthalten, in das Datenfeld des Ethernet-Frames eingefügt. Die Länge des zu verschlüsselnden Hashwerts erfordert zwei AES-Blöcke. Abbildung 10.5 stellt den Aufbau der abgesicherten *Dach-bewegen*-Nachricht dar.

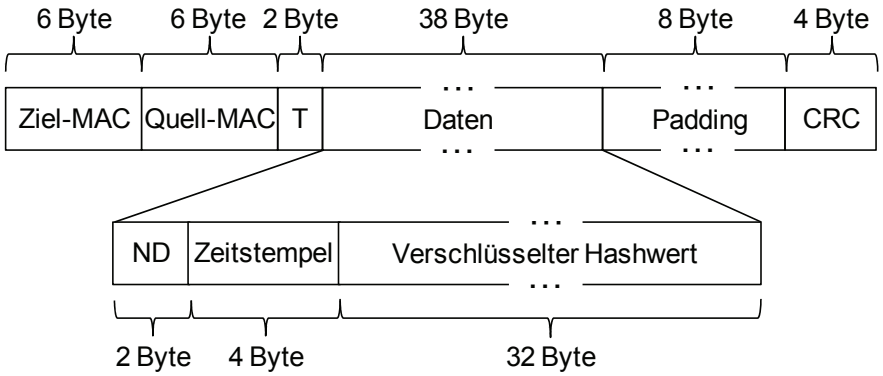

Abbildung 10.5: Format der abgesicherten *Dach-bewegen*-Nachricht

10.2.4 Prüfung des Schutzes

Die Prüfung des Schutzes besteht, wie in Abschnitt 8.5 beschrieben, aus der Untersuchung der Effektivität und der Effizienz des Schutzes. Zur Untersuchung der Wirksamkeit wurde ein Rechner einerseits physisch an die Programmierschnittstelle des Dachsteuergeräts, andererseits an den Ethernet-Feldbus des Demonstrators angebunden, um die Funktionalität des Dachsteuergeräts und die Kommunikation zwischen Armaturensteuergerät und Dachsteuergerät zu manipulieren. Abbildung 10.6 stellt diese physische Ankopplung an das Feldbuskabel dar.

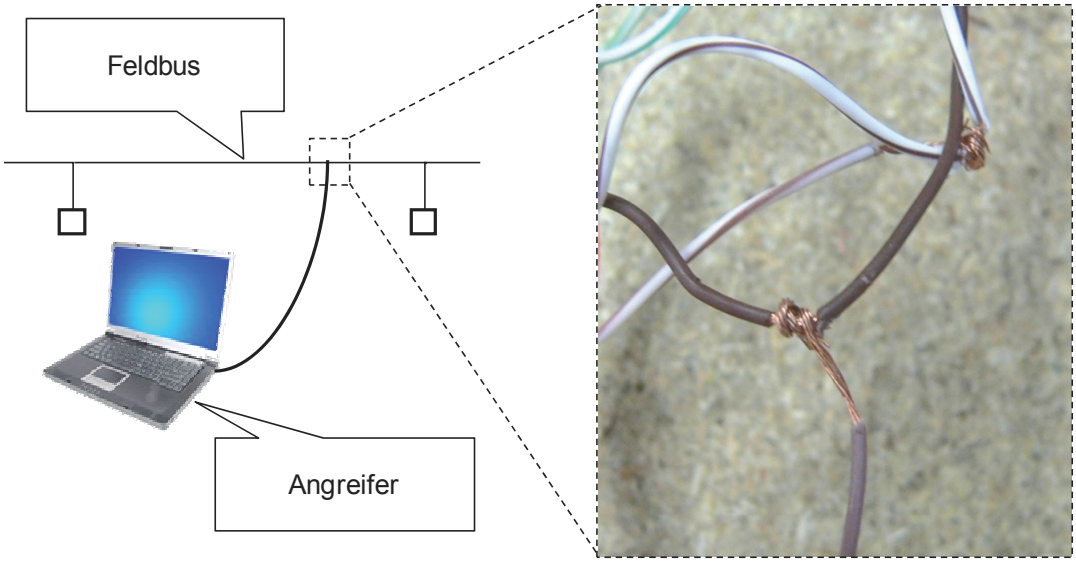

Abbildung 10.6: Physische Ankopplung an den Feldbus des Kfz-Steuergerätenetzwerks

Der angebundene Rechner führt das in Abschnitt 8.5 beschriebene Penetrationstestsystem aus, mit dem Ziel, das Dach des Kfz-Modells zu öffnen. Zur Prüfung des Schutzes der Kommunikation wurden sowohl eigene Nachrichten erzeugt und an das Dachsteuergerät gesendet als auch aufgezeichnete, legitime Nachrichten auf den Feldbus aufgebracht. Um auch fälschlicherweise als Angriff detektierte legitime Nachrichten (*false positives*) erkennen zu können, wurden während des Penetrationstests auch legitime Nachrichten durch das Armaturenbrett gesendet. Zur Prüfung der Wirksamkeit des Schutzes der Funktionalität wurde die Feldgerätesoftware durch manipulierte Programme überschrieben.

Im Rahmen der Prüfung wurden 150 Nachrichten durch das Penetrationstestsystem erzeugt. Durch Betätigung des Dachbedienschalters am Armaturenbrett wurden 42 legitime Nachrichten gesendet[19], die zusätzlich von dem Penetrationstestsystem zu einem späteren Zeitpunkt erneut eingespielt wurden. Dabei wurden alle Angriffe als solche erkannt, und alle legitimen Nachrichten wurden als legitim erkannt.

Zudem wurden zwei manipulierte Softwareprogramme auf das Dachsteuergerät „geflasht“. Bei einem Angriff wurde die *Dach-bewegen*-Funktionalität manipuliert, die Schutzfunktionalitäten aber intakt gelassen. Bei dem zweiten Angriff wurden sowohl die *Dach-öffnen*-Funktionalität als auch die Schutzfunktionalität manipuliert. In beiden Fällen wurde die Manipulation durch das überwachende Steuergerät erkannt.

Zur Bestimmung der Effizienz des Schutzes wurde der Ressourcenbedarf des Schutzes bestimmt. Um auch experimentell zu zeigen, dass die Schutzsoftware zeitlich deterministisch ausgeführt wird, wurden die Ausführungszeiten der einzelnen Schutzmechanismen bei allen Angriffen gemessen und miteinander verglichen. Die Zeiten waren stets identisch. Aufgrund der verwendeten Strukturierten Komponenten und deren Konfiguration ergibt sich der in Tabelle 10.2

[19] Die legitimen Nachrichten wurden durch Betätigung des Bedienelements auf dem Armaturenbrett erzeugt.

dargestellte Ressourcenbedarf zum Schutz der Kommunikation, der jeweils von den beteiligten Steuergeräten aufgebracht werden muss.

Tabelle 10.2: Ressourcenbedarf des Kommunikationsschutzes auf dem Dachsteuergerät und dem Armaturensteuergerät

Komponente	**Rechen-zeit [ms]**	**Speicherplatz Code / konst. Daten [Byte]**	**Speicherplatz var. Daten [Byte]**	**Bandbreite [Byte]**
SecurityLayer1	0,095	684	0	0
SecurityLayer2	0,105	494	0	0
SecurityLayer3	0,02	320	0	0
SecurityLayer4	0,022	308	0	0
Aes	10,008	5288	0	0
Sha1	6,255	3488	20	20
VarId	0,102	520	48	0
UnixTime	0,022	812	49	4
RoofReaction	0,005	80	0	0
Summe	**16,634**	**11994**	**117**	**24**

Der Ressourcenbedarf des Schutzes der Funktionalität ist für das überwachende Steuergerät und für das überwachte Steuergerät unterschiedlich. Tabelle 10.3 stellt den gesamten Ressourcenbedarf für das überwachte Steuergerät (Dachsteuergerät) dar, Tabelle 10.4 für das überwachende Steuergerät (Armaturensteuergerät). Da zur Durchführung einer Code-Überprüfung zwei Nachrichten (zur Anforderung der Überprüfung und zur Meldung des Ergebnisses) zu schützen sind, beträgt die Rechenzeit zum Schutz der Kommunikation das Doppelte des Werts zur Absicherung einer einzigen Nachricht.

Tabelle 10.3: Ressourcenbedarf des gesamten Schutzes auf dem Dachsteuergerät

Komponente	**Rechen-zeit [ms]**	**Speicherplatz Code / konst. Daten [Byte]**	**Speicherplatz var. Daten [Byte]**	**Bandbreite [Byte]**
Schutz der Kommunikation	33,268	11994	117	24
SecurityLayer5	0,259	359	208	0
CodeVerifier	137,042	20029	64	32
Summe	**170,569**	**32382**	**389**	**56**

Tabelle 10.4: Ressourcenbedarf des gesamten Schutzes auf dem Armaturensteuergerät

Komponente	Rechen-zeit [ms]	Speicherplatz Code / konst. Daten [Byte]	Speicherplatz var. Daten [Byte]	Bandbreite [Byte]
Schutz der Kommunikation	33,268	11994	117	24
SecurityLayer5	0,259	359	208	0
CodeVerifier	0,215	5238	4	32
Summe	**33,742**	**17591**	**329**	**56**

Somit erfordert der Schutz einen Speicherplatz von 32,4 kB bzw. 17,6 kB im ROM und jeweils 0,4 kB im RAM der beteiligten Steuergeräte. Die Rechenzeit, welche zur Integration und zur Prüfung der Unterscheidungsmerkmale der *Dach-bewegen*-Nachricht benötigt wird, beträgt zusammengenommen weniger als 34 ms. Verglichen mit Speicherplatzanforderungen von mehreren Megabyte, die zur Absicherung von Standard-IT-Rechnern erforderlich sind, ist die hier veranschlagte Speicherbelegung als sehr gering anzusehen und kann auf 16-bit-Mikrocontrollern und leistungsfähigeren Rechnern eingesetzt werden (vgl. Abschnitt 2.3.1). Setzt man die erforderliche Rechenzeit von 34 ms in Relation zu den bei automobilen Bedienfunktionen üblichen Taster-Entprellzeiten von zumeist über 150 ms, kann die Rechenzeit zum Schutz der Kommunikation als ausreichend kurz betrachtet werden.

Die Überprüfung des Programmcodes und die Absicherung der Nachrichten zur Kommunikation im Rahmen der verteilten Codeüberwachung erfordern mit ca. 204 ms eine deutlich längere Rechenzeit. Es ist jedoch ausreichend, diese Prüfung zu bestimmten Zeitpunkten durchzuführen, beispielsweise bei dem Anlassen des Fahrzeugs. Daher ist die benötigte Rechenzeit akzeptabel.

10.3 Fallbeispiel Fernbedienung eines Kaffeeautomaten

10.3.1 Aufbau der Fernbedienung des Kaffeeautomaten

Das zweite Fallbeispiel verwendet den Demonstrator *Industrieller Kaffeeautomat*, der zur drahtlosen Fernbedienung mit einer Infrarot-Schnittstelle ausgestattet ist. Als Fernbediengerät kann jedes programmierbare Gerät herangezogen werden, welches über eine Schnittstelle gemäß der IrDA-1.1-Spezifikation [IrDA07] verfügt, wie die verwendete PDA-Uhr [Geig07]. Abbildung 10.7 stellt die Infrarotkommunikation zwischen der PDA-Uhr und dem IAS-Web-Board dar, das die Applikation zur Steuerung des Kaffeeautomaten ausführt.

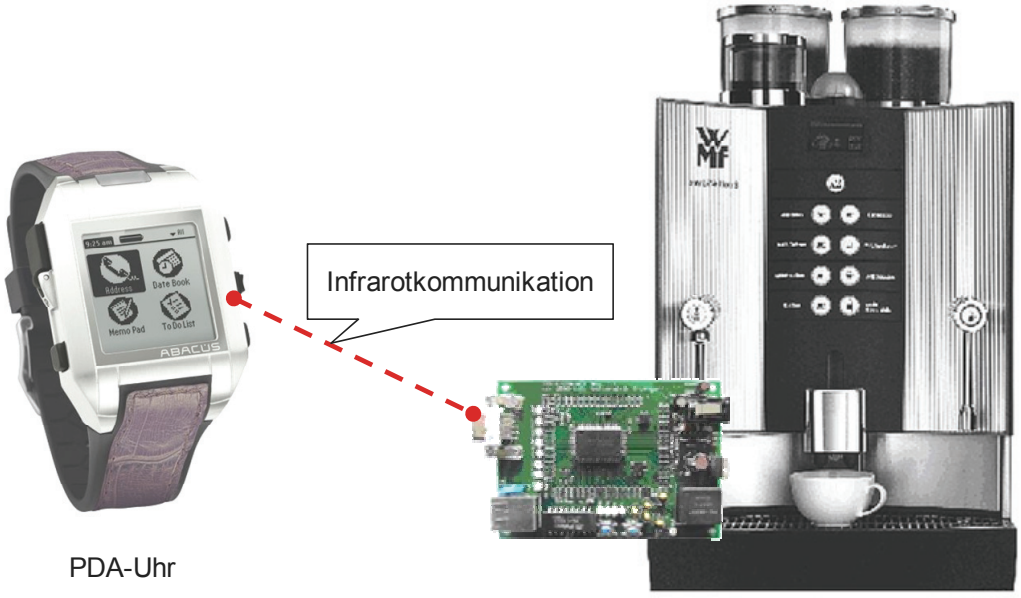

Abbildung 10.7: Fernbedienung des industriellen Kaffeeautomaten über Infrarot

Der Benutzer der PDA-Uhr kann die vom Kaffeeautomaten angebotenen Getränke herstellen lassen. Dazu wird die Information, welches Getränk gewählt wird, an den Kaffeeautomaten übermittelt.

10.3.2 Bedrohungsanalyse und Risikobewertung

Wie in Abschnitt 1.1 dargestellt, sind auch Kaffeeautomaten durch Angriffe auf die IT-Sicherheit bedroht. In dem geschilderten Fall war es Angreifern möglich, die Art des Getränks während des Bestellvorgangs zu verändern. Dies soll im Rahmen des Fallbeispiels *Fernbedienung eines Kaffeeautomaten* beispielhaft als Misuse-Case untersucht werden. Dabei werden die folgenden Annahmen zugrunde gelegt:

- Der Angreifer verfügt über die Fähigkeiten eines Skript-Kids.
- Der Besitzer der PDA-Uhr schützt diese vor unautorisiertem physischem Zugriff.
- Das Feldgerät im Inneren des Kaffeeautomaten verfügt über maskenprogrammierten Speicher.

Aufgrund der drahtlosen Kommunikationsverbindung bei der Getränkebestellung ist der physische Zugriff auf das Feldbusmedium (elektromagnetische Wellen) des hier betrachteten Kaffeeautomaten mit minimalem Aufwand möglich. Die Möglichkeit, auf Infrarot-Kommunikation zuzugreifen, ist in verbreitete Mobiltelefone und PDAs integriert, entsprechende Software steht als Free- und Shareware zur Verfügung, vgl. z. B. [VITO04] und [Omni05].

Dadurch ergeben sich Bedrohungen der Kommunikation zwischen PDA-Uhr und IAS-WebBoard sowie der Funktionalitäten dieser Geräte. Die beteiligten Geräte und die Kommunikationsnachrichten sind in Abbildung 10.8 dargestellt.

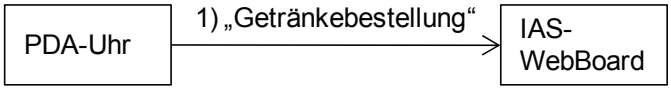

Abbildung 10.8: Kommunikationsvorgang zum Auslösen der Getränkeherstellung

Führt man die in Abschnitt 8.2 beschriebene Prozedur zur Identifikation der abzusichernden Feldgerätefunktionalitäten und Feldbuskommunikationen durch, ergibt sich folgendes Ergebnis: Die Funktionalitäten, die zur Erreichung des Misuse-Case-Ziels ausgeführt werden müssen, sind die Bestellfunktionen auf der PDA-Uhr und dem IAS-WebBoard. Diese Funktionalitäten kommunizieren über die *Getränkebestellung*-Nachricht, welche die Information enthält, welches Getränk gewünscht wird.

Daraus ergeben sich die Möglichkeiten für den Angreifer, den Bestellvorgang zu manipulieren. Durch die Veränderung der Funktionalitäten der beteiligten Feldgeräte kann eine fehlerhafte Bestellung erreicht werden. Es ist für den Angreifer jedoch aufgrund der getroffenen Annahmen nicht möglich, auf die PDA-Uhr zuzugreifen. Daher ist die Wahrscheinlichkeit für einen derartigen Angriff als gering einzuschätzen. Das eingesetzte Feldgerät verfügt über einen maskenprogrammierten Speicher, dessen Inhalt physikalisch unveränderlich ist. Aus diesem Grund ist die Wahrscheinlichkeit für eine Veränderung dieser Feldgerätefunktionalität ebenfalls gering.

Durch eine Manipulation der *Getränkebestellung*-Nachricht kann die Getränkewahl des Benutzers von dem Angreifer verändert werden. Das Abhören dieser Nachricht hat keine Auswirkung auf den Kaffeeautomaten, würde also das Ziel des Angreifers nicht erreichen. Eine inhaltliche Veränderung der in der Nachricht enthaltenen Getränkeinformation führt zum Ziel des Misuse-Cases und ist daher wahrscheinlich. Das Senden eigener Nachrichten oder die Manipulation der zeitlichen Eigenschaften von Nachrichten würde das Ziel des Misuse-Cases nicht erreichen. Diese Manipulationen werden daher als unwahrscheinlich eingestuft.

Die Risiken, welche sich aus den identifizierten Manipulationen ergeben, sind in Tabelle 10.5 dargestellt. Die Störung einer Getränkebestellung stellt im Wesentlichen ein Ärgernis dar und hat nur geringe finanzielle Folgen. Daher wird die Schwere der Auswirkung als gering (Stufe 1) eingeschätzt.

Tabelle 10.5: Risikobewertung der identifizierten Gefährdungen (Kaffeeautomat)

Bedrohung	Auswirkung	Wahrscheinlichkeit	Risiko
Kommunikation PDA-Uhr/IAS-WebBoard			
Abhören	Keine (0)	Gering (2)	0
Manipulation inh. Integrität	Änderung der Getränkeart (1)	Hoch (4)	4
Erzeugung eigener Nachrichten	Keine (0)	Gering (2)	0
Manipulation zeitl. Integrität	Keine (0)	Gering (2)	0
Funktionalitäten			
PDA-Uhr	Änderung der Getränkeart (1)	Gering (2)	2
IAS-WebBoard	Änderung der Getränkeart (1)	Gering (2)	2

10.3.3 Entwurf und Realisierung des Schutzes

Das größte Risiko wurde dem Angriff auf die inhaltliche Integrität der *Getränkebestellung*-Nachricht zugeordnet. Alle anderen möglichen Angriffe weisen nur ein maximal halb so großes Risiko auf. Daher ist die inhaltliche Integrität der *Getränkebestellung*-Nachricht zu schützen. Aus dem Zusammenhang zwischen Schutzfunktionalitäten und Schutzschichten (vgl. Abschnitt 7.5.3) folgt, dass die Schutzschichten 1–2 erforderlich sind. Aufgrund der geringen Fähigkeiten des prognostizierten Angreifers können schwächere als die in Abschnitt 8.4.2 vorgeschlagenen Schutzmechanismen herangezogen werden. In dem gegebenen Fall werden der Verschlüsselungsalgorithmus RC4 und die Hashfunktion MD5 verwendet.

Die Reaktion des Schutzes muss verhindern, dass ein Benutzer eine Getränkesorte erhält, die er nicht bestellt hat. Werden Manipulationen erkannt, soll kein Getränk ausgegeben werden und stattdessen eine Meldung auf dem Display des Kaffeeautomaten angezeigt werden. Somit dürfen als manipuliert erkannte Nachrichten von der für dieses Szenario entwickelten Reaktionskomponente (*CoffeeReaction*) nicht an die Applikation zur Steuerung des Kaffeeautomaten (*CoffeeControl*) weitergeleitet werden. Somit ergibt sich das in Abbildung 10.9 dargestellte Komponentenmodell zum Schutz der Getränkebestellung.

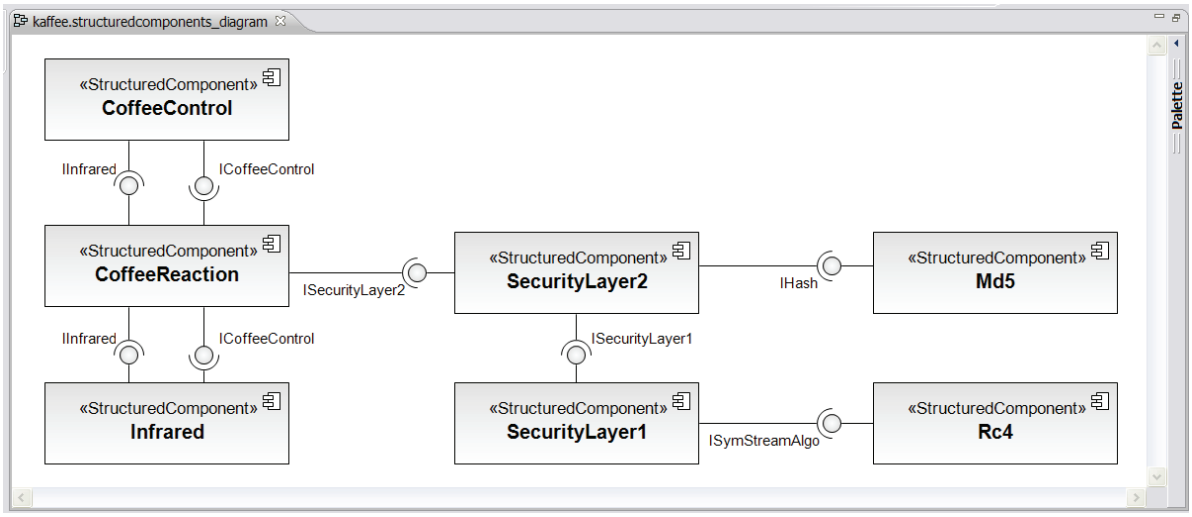

Abbildung 10.9: Komponentenmodell des Schutzes für die Getränkebestellung

Die von den Schutzmechanismen erzeugten Unterscheidungsmerkmale müssen in die Nachrichten zwischen PDA-Uhr und IAS-WebBoard integriert werden. Die Nachrichten zwischen den beiden Geräten sind in XML formatiert. Um die gewählte Getränkesorte an den Kaffeeautomaten zu übermitteln, ist in der *Getränkebestellung*-Nachricht das XML-Attribut *product* enthalten. Abbildung 10.10 stellt eine solche XML-Nachricht dar.

```
<?xml version="1.0" ?>
<MakeProduct product="CAPPU" />
```

Abbildung 10.10: Beispielhafte *Getränkebestellung*-Nachricht

Um eine unautorisierte Veränderung der Getränkesorte detektieren zu können, muss der Wert des XML-Attributs *product* durch die Hashfunktion abgesichert werden. Der Hashwert wird nach den XML-Daten übertragen. Der Aufbau[20] der zu übertragenden Nutzdaten und Unterscheidungsmerkmale ist in Abbildung 10.11 dargestellt.

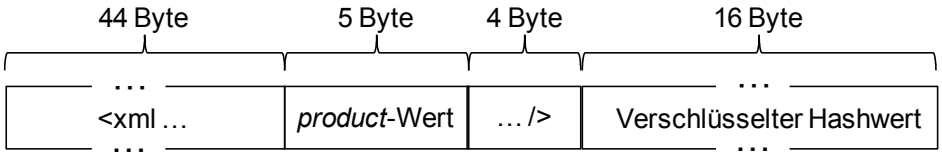

Abbildung 10.11: Format der geschützten Nachricht zur Kaffeebestellung

[20] Das Format der XML-Nachricht ist so definiert, dass die XML-Elemente immer dieselbe Länge aufweisen.

10.3.4 Prüfung des Schutzes

Die Validierung des Schutzes der Getränkebestellung erfolgt analog zu der in Abschnitt 10.2.4 beschriebenen Verifikation des Kfz-Steuergerätenetzwerk-Schutzes. Zur Durchführung der Penetrationstests wurde ein PC mit zwei Infrarotschnittstellen ausgerüstet und mit entsprechender Software zur Fehlersuche von Infrarotverbindungen ausgestattet, welche u. a. das Abhören und die Veränderung eigener Infrarotnachrichten ermöglicht.

Im Rahmen der Penetrationstests wurden 30 Nachrichten von der PDA-Uhr erzeugt, wobei 15 Nachrichten manipuliert und 15 unmanipuliert belassen wurden. Dabei wurden alle manipulierten Nachrichten als manipuliert und alle legitimen Nachrichten als legitim erkannt. Zudem wurden wieder die Rechenzeiten der Schutzmechanismen gemessen. Diese waren stets gleich.

Zur Beurteilung der Effizienz des Schutzes führt Tabelle 10.2 den Ressourcenbedarf der verwendeten Strukturierten Komponenten auf.

Tabelle 10.6: Ressourcenbedarf zur Absicherung der Kaffeebestellung

Komponente	**Rechenzeit [ms]**	**Speicherplatz Code / konst. Daten [Byte]**	**Speicherplatz var. Daten [Byte]**	**Bandbreite [Byte]**
SecurityLayer1	0,095	684	0	0
SecurityLayer2	0,105	494	0	0
Rc4	4,128	775	0	0
Md5	0,876	10690	64	16
CoffeeReaction	0,005	80	0	0
Gesamt	**5,209**	**12723**	**64**	**16**

Somit erfordert der Schutz der Kaffeebestellung einen zusätzlichen Speicherplatz von ca. 13,0 kB im ROM und weniger als 0,1 kB im RAM der beteiligten Geräte. Die erforderliche Bandbreite nimmt um 16 Byte zu. Die Rechenzeit beträgt für Absicherung und Prüfung zusammengenommen weniger als 11 ms. Der erforderliche Speicherplatz ist so gering, dass er von typischen 8-bit-Mikrocontrollern aufgebracht werden kann (vgl. Abschnitt 2.3.1). Die Rechenzeit, die zusätzlich zur Absicherung der Nachrichten anfällt, ist im Kontext einer Kaffeebestellung als vernachlässigbar zu betrachten.

Im Rahmen dieses Kapitels wurde anhand der beiden Fallbeispiele gezeigt, dass der Schutz die Anforderungen an den Schutz der IT-Sicherheit auf der Feldebene erfüllt:

- Wirksamkeit gegen Angriffe auf Funktionalität und Kommunikation bei physikalischem Zugriff auf die Systemelemente der Feldebene
- Sehr hohe Ressourceneffizienz im Vergleich zu Standard-IT-Schutz
- Anpassbarkeit an unterschiedliche Feldgerätetypen und Feldbussysteme
- Zeitlich deterministisches Verhalten
- Hohe Korrektheit der Implementierung durch die Verwendung von betriebsbewährten Strukturierten Komponenten

Die Ressourcenbetrachtungen zeigen, dass der Ressourcenbedarf des Schutzes je nach erforderlichem Schutzumfang und eingesetzten Schutzmechanismen unterschiedlich ausfällt und der Schutz somit an unterschiedlich leistungsfähige Feldgeräte angepasst werden kann. Der vollständige Schutzumfang kann, wenn er mit den vorgeschlagenen Schutzmechanismen mittlerer Schutzstärke realisiert wird, in 16-bit-Mikrocontrollern sowie leistungsfähigeren Rechnersystemen eingesetzt werden. Bei reduziertem Schutzumfang oder reduzierter Schutzstärke kann der Schutz auch in 8-bit-Mikrocontrollern ausgeführt werden. Mit dem vorgestellten Werkzeug CDE kann der Schutz der IT-Sicherheit auf der Feldebene modellbasiert erstellt und generiert werden, wodurch der Aufwand zur Absicherung der Feldebene von konkreten Automatisierungssystemen gering gehalten wird. Im folgenden letzten Kapitel wird das Konzept zum effizienten Schutz der Feldebene abschließend bewertet sowie die Voraussetzungen und Grenzen für seinen Einsatz zusammengefasst. Schließlich wird ein Ausblick auf mögliche Erweiterungen der Arbeit gegeben.

11 Schlussbetrachtungen und Ausblick

11.1 Ergebnisse

In der vorliegenden Arbeit wurde das Konzept des effizienten Schutzes der IT-Sicherheit auf der Feldebene von Automatisierungssystemen hergeleitet. Aufgrund der exponierten Lage vieler Automatisierungssysteme wurde als Prämisse angenommen, dass ein Angreifer über physischen Zugriff auf die Systemelemente der Feldebene verfügt. Ausgehend von dieser Randbedingung hat sich gezeigt, dass es nicht ausreichend ist, die Feldebene vor Angriffen aus höheren Ebenen der Automatisierungspyramide abzusichern. Wegen der Eigenschaften vieler Arten von Automatisierungssystemen kommt eine physische Absicherung ebenfalls nicht infrage. Stattdessen müssen informationstechnische Schutzfunktionalitäten auf die einzelnen Systemelemente der Feldebene verteilt werden, um die Feldbuskommunikation und die Feldgerätefunktionalitäten wirksam schützen zu können.

Aufgrund der geringen, je nach Feldgerätetyp bzw. Feldbussystem unterschiedlich großen Ressourcen, die den Feldgeräten und Feldbussen zur Ausführung von Schutzfunktionalitäten zur Verfügung stehen, müssen diese Schutzfunktionalitäten über eine hohe Effizienz verfügen. Um eine hohe Effizienz zu erreichen, wurden zwei Strategien verfolgt. Zum einen wurde eine Anpassbarkeit an die Art der Bedrohungen und die Schwere der zu erwartenden Auswirkungen realisiert, sodass ausschließlich die minimal erforderlichen Schutzfunktionalitäten durch Schutzmechanismen realisiert werden können, die eine minimal erforderliche Schutzstärke aufweisen. Bezüglich der Wahl der Schutzmechanismen besteht eine große Flexibilität, sodass diese entsprechend ihrer Schutzstärke und ihrem Konsum der unterschiedlichen Ressourcen – ROM- und RAM-Speicherplatz, Rechenzeit und Bandbreite – ausgewählt werden können. Zum Zweiten wurde eine Softwarearchitektur definiert, die vorschreibt, dass der Schutz in Schutzschichten angeordnet wird, wobei die oberen Schutzschichten auf den unteren Schutzschichten aufbauen. Dadurch können Operationen der unteren Schutzschichten durch die oberen Schutzschichten verwendet werden, wodurch ebenfalls Ressourcen eingespart werden.

Die auf der Feldebene oftmals erforderliche Echtzeitfähigkeit ergibt sich aus der Verwendung von Schutzmechanismen, die eine zeitliche Determiniertheit aufweisen. Daher können mit dem Konzept Schutzfunktionalitäten realisiert werden, welche harten Echtzeitanforderungen genügen.

11.2 Bewertung des Konzepts

Wirksamkeit des Schutzes

Im Rahmen dieser Arbeit wurde davon ausgegangen, dass ein Angreifer über physischen Zugriff auf die Systemelemente der Feldebene verfügt und somit die Feldbuskommunikation und Feldgerätefunktionalität manipulieren kann. Aus diesem Grund ist das Ziel des vorliegenden Konzepts die Verhinderung von schädlichen Auswirkungen durch Detektion der Angriffe und eine entsprechende Reaktion.

Die Wirksamkeit des vorgeschlagenen Konzepts basiert auf Funktionsprinzipien von bewährten Ansätzen zum Schutz der IT-Sicherheit in der Standard-IT, welche entsprechend den Eigenschaften und Randbedingungen der Feldebene umgesetzt wurden. Diese Wirksamkeit wurde im Rahmen der Validierung des Schutzkonzepts nachgewiesen.

Systemunabhängigkeit des Schutzes

Die zur Detektion von Angriffen auf die Kommunikation verwendeten Unterscheidungsmerkmale werden auf der Applikationsebene des ISO/OSI-Referenzmodells angesiedelt. Sie werden somit von den feldbussystemspezifischen Eigenschaften der darunterliegenden Ebenen nicht berührt. Das Konzept ermöglicht es jedoch, die in Feldbussystemen ggf. vorhandenen Unterscheidungsmerkmale einzubeziehen. Die Absicherung der Feldgerätefunktionalität betrachtet den Programmcode der Feldgeräte als unveränderliche Daten, wobei die darin abgelegte Information und deren Format nicht relevant sind.

Aufgrund der Lage der hinzugefügten Unterscheidungsmerkmale in der Applikationsebene können Nachrichten, die gegen Veränderungen geschützt sind, auch von Feldgeräten empfangen und verarbeitet werden, welche keinerlei Schutzfunktionalitäten enthalten. Das empfangende Feldgerät kann in diesem Fall zwar keine Manipulationen detektieren, das Konzept erfüllt somit aber nicht nur die Anforderung nach Systemunabhängigkeit, sondern es erreicht dadurch zusätzlich die Interoperabilität abgesicherter Kommunikationsnachrichten mit jedem Systemelement auf der Feldebene – unabhängig davon, ob diese über Schutzfunktionalitäten verfügen oder nicht.

Die durch das Konzept vorgegebenen Softwarebausteine können in beliebigen Programmiersprachen realisiert werden. Bei der dargestellten Realisierungsform wurden diese als Strukturierte Komponenten in der Programmiersprache C implementiert, die für nahezu jeden Mikrocontroller übersetzt werden können. In den dargestellten Fallbeispielen wurde gezeigt, dass diese Strukturierten Komponenten zur Absicherung unterschiedlicher Mikrocontrollertypen und unterschiedlicher Feldbusstandards geeignet sind.

Wirtschaftlichkeit des Schutzes

Die Wirtschaftlichkeit des Konzepts an sich ergibt sich aus der Effizienz des Schutzes. Aufgrund des im Vergleich zu Standard-IT-Schutz sehr geringen Ressourcenbedarfs ist es möglich, Schutzfunktionalitäten auf leistungsschwachen Systemelementen der Feldebene auszuführen. Somit können preisgünstige Systemelemente zum Aufbau der Feldebene eines Automatisierungssystems verwendet werden und dennoch wirksam vor Angriffen auf die IT-Sicherheit geschützt werden. Der Schutz kann bei vollständigem Schutzumfang mit mittlerer Schutzstärke auf 16-bit-Mikrocontrollern eingesetzt werden. Bei Reduktion des Schutzumfangs bzw. der Schutzstärke können auch 8-bit-Mikrocontroller abgesichert werden.

Aufgrund der vorgeschlagenen Softwarearchitektur, der realisierten Strukturierten Komponenten sowie der Werkzeugunterstützung zur Modellierung und Codegenerierung ist eine einfache, effiziente Entwicklung des Schutzes für die Feldebene von konkreten Automatisierungssystemen möglich. Auch die Integration weiterer Schutzmechanismen ist durch die definierten Schnittstellen mit geringem Aufwand möglich. Somit ergibt sich die Wirtschaftlichkeit des Konzepts zum effizienten Schutz der IT-Sicherheit auf der Feldebene sowohl im Einsatz als auch beim Entwurf und der Realisierung des Schutzes.

11.3 Voraussetzungen und Grenzen des Konzepts

Voraussetzung für die Absicherung von Kommunikation durch das vorgestellte Konzept ist, dass in einer Informationsquelle auch eine Verarbeitung von Informationen stattfinden kann. Dies ist bei modernen Feldgeräten der Fall, nicht aber bei solchen Sensoren und Aktoren, die rein analoge Signale (d. h. Signale, die durch die Ebene 1 des ISO/OSI-Referenzmodells vollständig beschrieben werden) produzieren bzw. konsumieren. Ein Beispiel dafür ist ein Temperatursensor, der die gemessene Temperatur durch den Betrag einer elektrischen Spannung kommuniziert. Die Erkennung von Manipulationen an der durch diesen Spannungswert übermittelten Information ist durch das vorliegende Konzept nicht möglich.

Zur Manipulation der IT-Sicherheit können außer technischen Angriffen auch *Social-Engineering*-Methoden herangezogen werden. Dabei werden Personen mit legitimem Zugriff auf das System durch Angreifer so beeinflusst, dass sie Eingriffe vornehmen, die dem Willen des Angreifers entsprechen, oder vertrauliche Informationen preisgeben. Gegen derartige Manipulationen sind technische Schutzkonzepte wie das im Rahmen dieser Arbeit vorgeschlagene wirkungslos und sollten daher stets durch entsprechende organisatorische Maßnahmen unterstützt werden.

11.4 Ausblick

Das im Rahmen dieser Arbeit vorgeschlagene Konzept befasst sich im Kern mit der technischen Realisierung des Schutzes der IT-Sicherheit auf der Feldebene. Es wird zwar eine Vorgehensweise vorgeschlagen, welche die Tätigkeiten beschreibt, die zur Absicherung der Feldebene von konkreten Automatisierungssystemen durchgeführt werden müssen; diese Vorgehensweise ist aber unabhängig von den Engineering-, Management- und Unterstützungsprozessen ausgeführt, die bei der Entwicklung von Automatisierungssystemen durchzuführen sind. Daraus ergibt sich als Anknüpfungspunkt für diese Arbeit eine Integration der beschriebenen Tätigkeiten des Security-Engineerings in bestehende Prozessarchitekturen von Herstellern und Anwendern der Automatisierungstechnik.

Dabei kann u. a. betrachtet werden, welche grundlegenden Prozessfähigkeiten von einer Organisation erbracht werden müssen, um ein stringentes Security-Engineering durchführen zu können, und wie Security-Engineering für die Feldebene in Prozessreifegradmodelle wie CMMI [CMMI06] oder Automotive SPICE [ASPI08] eingefügt werden muss.

In diesem Zusammenhang erscheint auch die Betrachtung von Zusammenarbeitsmodellen interessant, welche beschreiben, wie die Kooperation bei der Entwicklung komplexer Automatisierungssysteme zwischen Anwendern, Herstellern und Zulieferern stattfinden muss, um sichere Automatisierungssysteme (im Sinne von IT-Sicherheit) zu realisieren. Bei der IT-Sicherheit handelt es sich um eine Eigenschaft des Gesamtsystems, bei dem das schwächste Glied in der Kette die tatsächliche Wirksamkeit des Schutzes bestimmt. Daher sind z. B. in der Automobilindustrie verbreitete Zusammenarbeitsformen zwischen Systemhersteller (OEM) und (Sub-)Komponentenlieferanten (Tier 1, Tier 2, …) auf Schwächen bezüglich der Entwicklung sicherer Systeme (im Sinne von IT-Sicherheit) zu prüfen.

Ein weiter in die Zukunft reichender Anknüpfungspunkt an die vorliegende Arbeit besteht darin, solche Konzepte unter IT-Sicherheits-Gesichtspunkten zu betrachten, die momentan noch Gegenstand der Forschung sind und nachhaltige Änderungen in der funktionellen Struktur von Automatisierungssystemen (einschließlich der Feldebene) hervorrufen können. Ein Beispiel hierfür ist die Einführung von Selbstorganisation in Automatisierungssysteme [Muba07]. Derartige Konzepte eröffnen neue Möglichkeiten bei der Entwicklung und dem Betrieb von Automatisierungssystemen, unter Umständen bieten sie aber auch Angreifern neue Ansatzpunkte zur Manipulation.

Literaturverzeichnis

[Ande01a] R. Anderson: *Security Engineering: A Guide to Building Dependable Distributed Systems*. Hoboken: Wiley & Sons, 2001.

[Ande01b] R. Anderson: *Why Information Security is Hard – An Economic Perspective*. 17th Annual Computer Security Applications Conference, New Orleans. In: Proceedings of the 17th Annual Computer Security Applications Conference. Los Alamitos: IEEE Computer Society Press, 2001, S. 358–365.

[ASAM08] ASAM e. V.: *Association for Standardisation of Automation and Measuring Systems Web Site*. http://www.asam.net
Seitenabruf: 06.11.2007

[ASPI08] Automotive Special Interest Group: *Automotive SPICE Process Assessment Model v2.4*. 2008. http://www.automotivespice.com

[Aven07] Aventail Corp. : *SSL VPN Appliances*.
http://www.aventail.com/products/appliances/default.asp
Seitenabruf: 11.09.2007

[Bach03] D. Bachfeld: *War der Wurm drin? IT-Sicherheit in der US-Stromversorgung*. In: c't – Magazin für Computertechnik (2003) H. 18, S. 34–35.

[Bach04] D. Bachfeld: *Fremdgesteuert – IT-Sicherheit im intelligenten Haus*. In: c't – Magazin für Computertechnik (2004) H. 5, S. 126–127.

[Balz01] H. Balzert: *Lehrbuch der Software-Technik, Band 1: Software-Entwicklung*. 2. Aufl., Heidelberg: Spektrum Akademischer Verlag, 2001.

[Balz99] H. Balzert: *Lehrbuch der Objektmodellierung: Analyse und Entwurf*. Heidelberg, Berlin: Spektrum Akademischer Verlag, 1999.

[Beck07] K. Beck: *Implementation Patterns*. Upper Saddle River: Addison-Wesley, 2007.

[BeHa04] T. Beierlein, O. Hagenbruch: *Taschenbuch Mikroprozessortechnik*. München: Carl Hanser, 2004.

[BeMo07] K. Bettenhausen, W. Morr: *Informationssicherheit in der Automatisierung – Mehr als eine technologische Herausforderung*. In: atp – Automatisierungstechnische Praxis 49 (2007) H. 4, S. 76–79.

[Beth90] T. Beth: *Zur Sicherheit der Informationstechnik*. In: Informatik Spektrum 13 (1990), H. 4, S. 204–215.

[BMB+05] R. Bless, S. Mink, E.-O. Blaß, M. Conrad, H.-J. Hof, K. Kutzner, M. Schöller: *Sichere Netzwerkkommunikation*. Berlin, Heidelberg: Springer, 2005.

[BSI02] BSI-Referat 125: *Einführung von Intrusion-Detection-Systemen – Grundlagen*. Bundesamt für Sicherheit in der Informationstechnik, 2002.

[BSI03] *Durchführungskonzept für Penetrationstests*. Bundesamt für Sicherheit in der Informationstechnik, 2003. http://www.bsi.bund.de/literat/studien/pentest/penetrationstest.pdf

[BSM+04] F. Budinsky, D. Steinberg, E. Merks, R. Ellersick, T. J. Grose: *Eclipse Modeling Framework*. Boston: Addison-Wesley, 2004.

[BSW01] A. Beutelspacher, J. Schwenk, K.-D. Wolfenstetter: *Moderne Verfahren der Kryptographie*. 4. Aufl., Braunschweig/Wiesbaden: Vieweg Verlag, 2001.

[Buen02] P. Bueno: *Understanding IDS for Linux*. In: Linux Journal (2002) H. 5. Online unter: http://www.linuxjournal.com/article/5616

[Buss96] R. Busse: *Feldbussysteme im Vergleich*. München: Pflaum, 1996.

[Cerf93] V. Cerf: *How the Internet Came to Be*. In: The Online User's Encyclopedia, B. Aboba (Hrsg.). Boston: Addison-Wesley, 1993.

[CERT07] United States Computer Emergency Readiness Team: *NETxAutomation NETxEIB OPC Server fails to properly validate OPC server handles*. http://www.kb.cert.org/vuls/id/296593 Seitenabruf: 04.05.2007

[Chec07a] CheckPoint Software Technologies, Ltd.: *Products & Technologies*. http://www.checkpoint.com/products/index.html Seitenabruf: 05.09.2007

[Chec07b] CheckPoint Software Technologies, Ltd.: *ZoneAlarm Internet Security Suite 7.1*. http://www.zonealarm.com/store/content/home.jsp Seitenabruf: 05.09.2007

[Clau03] H. Claus: *Untersuchung und Implementierung sicherer Verfahren zur Prozessdatenübertragung über das Internet*. Diplomarbeit Nr. 1921, IAS, Universität Stuttgart, 2003.

[CMMI06] Software Engineering Institute: *CMMI for Development, Version 1.2*. Pittsburgh: Carnegie Mellon University, 2006.

[Cool03] J. Cooling: *Software Engineering for Real-Time Systems*. Harlow, UK: Pearson Education, 2003.

[Coul01] D. Coulson: *Mastering iptables*. In: Linux Format 14 (2001) H. 5, S. 82–87.

[Dell07] Dell GmbH: *Dell-Desktops – einfache Heimanwendungen*. http://www1.euro.dell.com/content/products/features.aspx/desktops_good Seitenabruf: 16.05.2007

[Dier04] R. Dierstein: *Sicherheit in der Informationstechnik – der Begriff IT-Sicherheit*. In: Informatik Spektrum (2004) H. 4, S. 343–353.

[Dorr02] J. Dorrer: *Industrie PCs: Embedded Computing im Maschinen- und Anlagenbau*. SPS-Magazin – Zeitschrift für Automatisierungstechnik (2002) H. 8, S 34–36.

[Dujm04] S. Dujmović: *Effiziente Softwareentwicklung mit Komponenten*. In: atp – Automatisierungstechnische Praxis 46 (2004) H. 8, S. 37–45.

[Eber05] S. Eberle: *Adaptive Internetanbindung von Feldbussystemen*. Diss. IAS, Universität Stuttgart, 2005. IAS-Forschungsberichte, Bd. 1/2005.

[EbGö04a] S. Eberle, P. Göhner: *Softwareentwicklung für eingebettete Systeme mit strukturierten Komponenten. Teil 1: Komponentenorientierte Zerlegung*. In: atp – Automatisierungstechnische Praxis 46 (2004) H. 3, S. 41–52.

[EbGö04b] S. Eberle, P. Göhner: *Softwareentwicklung für eingebettete Systeme mit strukturierten Komponenten. Teil 2: Komponentenorientierte Modellierung und Realisierung*. In: atp – Automatisierungstechnische Praxis 46 (2004) H. 4, S. 61–73.

[Ecke03] C. Eckert: *IT-Sicherheit: Konzepte – Verfahren – Protokolle*. 2. Aufl., München, Wien: Oldenbourg, 2003.

[Ecli07] *Eclipse – an open development platform*. Eclipse Foundation, 2007. http://www.eclipse.org

[Eder05] S. Eder: *Entwicklung von Interfaces für kryptographische Komponenten des IAS-WebStacks*. Diplomarbeit Nr. 2016, IAS, Universität Stuttgart, 2005.

[EKM+08] T. Eisenbarth, T. Kasper, A. Moradi, C. Paar, M. Salmasizadeh, M. T. M. Shalmani: *On the Power of Power Analysis in the Real World: A Complete Break of the KeeLoq Code Hopping Scheme*. 28th Annual International Cryptology Conference, Santa Barbara. In: Advances in Cryptology – CRYPTO 2008: 28th Annual International Cryptology Conference, Proceedings. Berlin, Heidelberg: Springer, 2008, S. 203–220.

[EMF05] *The Eclipse Modeling Framework (EMF) Overview*. Eclipse Foundation, 2005. http://www.eclipse.org/modeling/emf/

[EN14908-1] EN 14908-1: *Firmenneutrale Datenkommunikation für die Gebäudeautomation und Gebäudemanagement – Gebäudedatennetzprotokoll – Teil 1: Datenprotokollschichtenmodell*. Europ. Komitee für elektrotechnische Normung, 2006.

[EN16484-5] EN 16484-5: *Building Automation and Control Systems. Data Communication Protocol*. Europ. Komitee für elektrotechnische Normung, 2003.

[EN50090-4] EN 50090-4-2: *Elektrische Systemtechnik für Heim und Gebäude (ESHG) – Teil 4-2: Medienunabhängige Schicht – Transportschicht, Vermittlungsschicht und allgemeine Teile der Sicherungsschicht für ESHG Klasse 1*. Europ. Komitee für elektrotechnische Normung, 2004.

[EN50159-2] EN 50159-2: *Bahnanwendungen – Telekommunikationstechnik, Signaltechnik und Datenverarbeitungssysteme – Teil 2: Sicherheitsrelevante Kommunikation in offenen Übertragungssystemen*. Europ. Komitee für elektrotechnische Normung, 2001.

[EN50254] EN 50254: *High efficiency communication subsystem for small data packages*. Europ. Komitee für elektrotechnische Normung, 1998.

[EN61131-3] EN 61131-3: *Speicherprogrammierbare Steuerungen, Teil 3: Programmiersprachen*. Europ. Komitee für elektrotechnische Normung, 1993.

[EN61784-1] EN 61784-1: *Digitale Datenkommunikationen in der Leittechnik – Teil 1: Feldbus-Kommunikationsprofile für die prozess- und fertigungstechnische Automatisierung*. Europ. Komitee für elektrotechnische Normung, 2004.

[Enst02] U. Enste: *Auf zu neuen Ufern – Wohin steuert die Leittechnik?* In: Chemie Produktion (2002) H. 11.

[EsAt06] J. Eschweiler, D. E. Atencio Psille: *Security@Work*. Berlin, Heidelberg: Springer, 2006.

[Ffiv07] F5 Networks, Inc.: *F5 Product Suite Overview*. http://www.f5.com/products Seitenabruf: 11.09.2007

[FIPS180-2] FIPS 180-2: *Specifications for the Secure Hash Standard*. Federal Information Processing Standards, 2002.

[FIPS197] FIPS 197: *Specification for the Advanced Encryption Standard (AES)*. Federal Information Processing Standards, 2001.

[FIPS46-3] FIPS 46-3: *Specifications of the Data Encryption Standard (DES)*. Federal Information Processing Standards, 1999.

[Flex05a] *FlexRay Communications System Protocol Specification v2.1, Revision A*. FlexRay Consortium, 2005. http://www.flexray.com

[Flex05b] *FlexRay Communications System Preliminary Central Bus Guardian Specification v 2.0.9*. FlexRay Consortium, 2005. http://www.flexray.com

[Flex05c] *FlexRay Communications System Preliminary Node-Local Bus Guardian Specification v 2.0.9*. FlexRay Consortium, 2005. http://www.flexray.com

[Flex06] *FlexRay Communications System Protocol Specification v2.1 Revision A Errata Sheet Version 1*. FlexRay Consortium, 2006. http://www.flexray.com

[Free07] FreeBSD.org: *The FreeBSD Handbook – Firewalls*. http://www.freebsd.org/doc/en_US.ISO8859-1/books/handbook/firewalls-ipfw.html Seitenabruf: 05.09.2007

[Frie94] A. Friedrich: *Mit SPS erfolgreich automatisieren: SPS-Steuerungsprojektierung, Programmentwicklung, praktisch erprobte Anwendungsbeispiele*. Poing: Franzis, 1994.

[FSec07] F-Secure Corp.: *F-Secure Linux Client Security*. http://www.f-secure.com/products/fsavcsl.html Seitenabruf: 14.11.2007

[GaBe04] E. Gamma, K. Beck: *Contributing to Eclipse: Principles, Patterns, and Plug-Ins*. Boston: Addison-Wesley, 2004.

[GaNa00] G. R. Ganger, D. F. Nagle: *Enabling Dynamic Security Management of Networked Systems via Device-Embedded Security*. Carnegie Mellon University School of Computer Science Technical Report CMU-CS-00-174, 2000.

[GeHa03] M. Gerdom, J. Hansemann: *Objektorientierung in Embedded-Systemen*. In: Elektronik (2003), H. 22, S. 66–69.

[Geig07] F. Geiger: *Entwicklung einer Bediensoftware für eine programmierbare Uhr zur Steuerung von Automatisierungssystemen*. Studienarbeit Nr. 2150, IAS, Universität Stuttgart, 2007.

[GGK+04] R. Grimm, K.-E. Großpietsch, H. Keller, I. Münch, K. Rannenberg, F. Saglietti: *Technische Sicherheit und Informationssicherheit – Unterschiede und Gemeinsamkeiten*. Workshop „Begriffsbildung Sicherheit“ des Fachbereichs „Sicherheit – Schutz und Zuverlässigkeit“ der Gesellschaft für Informatik (GI), Regensburg, 2004.

[GHRV96] E. Gamma, R. Helm, R. Johnson, J. Vlissides: *Entwurfsmuster: Elemente wiederverwendbarer objektorientierter Software*. München: Addison-Wesley, 1996.

[GMF07] *Eclipse Graphical Modeling Framework (GMF).* Eclipse Foundation, 2007. http://www.eclipse.org/modeling/gmf/

[GNU07] GNU.org: *The GNU Transport Layer Security Library*. http://www.gnu.org/software/gnutls/ Seitenabruf: 10.09.2007

[Göhn07] P. Göhner: *Softwaretechnik I.* Vorlesungsskript, IAS, Universität Stuttgart, WS 2007/08.

[Gott02] N. Gottschalk-Mazouz: *Risiko*. In: Handbuch Ethik, M. Düwell, C. Hübenthal, M. Werner (Hrsg.). Stuttgart: Metzler, 2002, S. 485–491.

[Gräf03] M. Gräfe: *C und Linux*. 2. Aufl., München, Wien: Carl Hanser, 2003.

[Grel05] D. Grell: *Tacho-Tüfteln – Was tun gegen die Kilometerstandfälscher?* In: c’t – Magazin für Computertechnik (2005) H. 8, S. 78–81.

[GuGö06] F. Gutbrodt, P. Göhner: *IT-Sicherheit auf der Feldebene von Automatisierungssystemen*. 9. Fachtagung Entwurf komplexer Automatisierungssysteme (EKA) 2006, Braunschweig. In: Tagungsband Entwurf komplexer Automatisierungssysteme. Braunschweig: Technische Universität Braunschweig, 2006, S. 175–191.

[Gutb07a] F. Gutbrodt: *Automatisierte, systematische Evaluierung der IT-Sicherheit auf der Feldebene von Automatisierungssystemen*. GMA-Kongress 2007, Baden-Baden. In: VDI-Berichte 1980. Düsseldorf: VDI-Verlag, 2007, S. 563–574.

[Gutb07b] F. Gutbrodt: *IT Security Protection at Field Level of Industrial Automation Systems*. The 2007 International Conference on Embedded Systems and Applications (ESA ’07), Las Vegas. In: Proceedings of the 2007 International Conference on Embedded Systems and Applications, H. R. Arabnia, L. T. Yang (Hrsg.). Las Vegas: CSREA Press, 2007, S. 9–15.

[GuWe05] F. Gutbrodt, M. Wedel: *Entwicklung eingebetteter Softwaresysteme mit Strukturierten Komponenten*. Workshop Modellbasierte Entwicklung eingebetteter Systeme (MBEES), Dagstuhl. In: Tagungsband Dagstuhl-Workshop MBEES: Modellbasierte Entwicklung eingebetteter Systeme 2005, T. Klein, B. Rumpe, B. Schätz (Hrsg.). Dagstuhl: Internationales Begegnungs- und Forschungszentrum für Informatik (IBFI), 2005, S. 105–112.

[Hama07] LogMeIn, Inc.: *Hamachi – VPN Evolved*. http://www.logmein.com Seitenabruf: 11.09.2007

[HART07] Analog Services, Inc.: *The HART Book*. http://www.analogservices.com/about_part0.htm Seitenabruf: 25.05.2007

[Hasc06] W. Hascher: *Wireless-Technologien im Überblick*. In: Elektronik Scout (2006) H. 1, S. 28–29.

[Hauf06] T. Hauf: *Prozessleitsysteme: Lebenszyklus und Qualität*. In: atp – Automatisierungstechnische Praxis 48 (2006) H. 2, S. 34–42.

[Heis08a] Heise Online: *Saboteur schaltet Turbine des britischen Kohlekraftwerks Kingsnorth aus*. 2008. http://www.heise.de/newsticker/meldung/120271 Seitenabruf: 16.12.2008

[Heis08b] Heise Online: *Hackerangriff auf Kaffeemaschine möglich*. 2008. http://www.heise.de/newsticker/meldung/109625 Seitenabruf: 22.06.2008

[Hell04] W. Hellmann: *Sicherer E-Mail-Versand für eingebettete Systeme*. Studienarbeit Nr. 1957, IAS, Universität Stuttgart, 2004.

[HiLi97] F. Hils, K.-P. Lindner: *Feldbus*. In: Handbuch der Prozeßautomatisierung, Prozeßleittechnik für verfahrenstechnische Anlagen, K. F. Früh (Hrsg.). 2. Aufl., München: Oldenbourg Industrieverlag, 1997, S. 223–254.

[Holl04] C. Holland: *CAN bus analysis on the move*. Embedded.com, 12.07.2004. http://www.embedded.com/showArticle.jhtml?articleID=55300105 Seitenabruf: 27.07.2007

[HuTh03] A. Hunt, D. Thomas: *Der Pragmatische Programmierer*. München, Wien: Carl Hanser, 2003.

[IEC14543-3] IEC 14543-3-2: *Informationstechnik – Architektur für Heim-Elektronik-Systeme (HES) – Teil 3-2: Kommunikationsschichten – Transportschicht, Vermittlungsschicht und allgemeine Teile der Sicherungsschicht für HES Klasse 1*. International Electrotechnical Commission, 2006.

[IEC60812] IEC 60812 Ed. 2.0 (2006): *Analysis techniques for system reliability – Procedure for failure mode and effects analysis (FMEA)*. International Electrotechnical Commission, 2006.

[IEC61025] IEC 61025 Ed. 2.0 (2006): *Fault Tree Analysis (FTA)*. International Electrotechnical Commission, 2006.

[IEC61508-4] IEC 61508-4:1998: *Functional safety of electrical/electronic/programmable electronic safety-related systems – Part 4: Definitions and abbreviations*. International Electrotechnical Commission, 1998.

[IEC61508-7] IEC 61508-7:1998: *Functional safety of electrical/electronic/programmable electronic safety-related systems – Part 7: Overview of techniques and measures*. International Electrotechnical Commission, 1998.

[IEC61784-3] IEC 61784-3: *Digitale Datenkommunikation in der Leittechnik – Teil 3: Profile für funktional sichere Kommunikation in industriellen Netzwerken*. International Electrotechnical Commission, 2005.

[IEEE802.3] IEEE 802.3: *Information Technology – Telecommunication & Information Exchange Between Systems – LAN/MAN – Specific Requirements – Part 3: Carrier Sense Multiple Access with Collision Detection (CSMA/CD) Access Method and Physical Layer Specifications.* Institute of Electrical and Electronics Engineers, 2002.

[IrDA07] *IrDA Data Specifications (SIR/MIR/FIR/VFIR)*. Infrared Data Association, 2007. http://www.irda.org

[ISO11898] ISO 11898: *Road vehicles –- Controller area network (CAN).* International Organization for Standardization, 2003.

[ISO7498-1] ISO/IEC 7498-1: *Information technology – Open Systems Interconnection – Basic Reference Model: The Basic Model*. International Organization for Standardization, 1996.

[ISO9126-1] ISO/IEC 9126-1: *Software-Engineering – Qualität von Softwareprodukten – Teil 1: Qualitätsmodell*. International Organization for Standardization, 2001.

[ISO9899] ISO/IEC 9899:TC3 (Committee Draft): *Programming languages – C*. International Organization for Standardization, 2007.

[Jahn03] M. Jahnke: *Schutz verteilter Intrusion-Detection-Systeme gegen Denial-of-Service-Angriffe*. 10. DFN-CERT/PCA-Workshop „Sicherheit in vernetzten Systemen“, Hamburg. In: Sicherheit in vernetzten Systemen. 10. DFN-CERT/PCA-Workshop, R. Schaumburg, M. Thorbrügge (Hrsg.). Hamburg, DFN-CERT GmbH Publications, 2003.

[JET04] *JET Tutorial Part 1 (Introduction to JET)*. Eclipse Foundation, 2004. www.eclipse.org/emft/projects/jet/

[Jin07] L. Jin: *Automatisierte Evaluierung von Schutzmechanismen für die IT-Sicherheit auf der Feldebene von Automatisierungssystemen*. Diplomarbeit Nr. 2125, IAS, Universität Stuttgart, 2007.

[John08] R. C. Johnson: *Security Goes Embedded*. EE Times, 2008. http://www.eetimes.com/showArticle.jhtml?articleID=206504092

[Juni07] Juniper Networks, Inc.: *Products and Services*. http://www.juniper.net/products_and_services Seitenabruf: 11.09.2007

[Karr02] M. Karresand: *A Proposed Taxonomy of Software Weapons*. Scientific Report, Linköping: FOI – Swedish Defence Research Agency, 2002.

[Käst07] J. Kästner: *Umfassendes Security-Konzept für Prozessleitsysteme (defense in depth)*. In: atp – Automatisierungstechnische Praxis 49 (2007) H. 4, S. 70–75.

[Kast08] W. Kastner: *Dezentrale Automation*. Vorlesungsskript, Automation Systems Group, Technische Universität Wien, SS 2008.

[KeRi88] B. W. Kernighan, D. Ritchie: *The C Programming Language*. 2. Aufl., Upper Saddle River: Prentice Hall International, 1988.

[Knap08] S. Knapton: *Power station break-in sparks security review*. The Telegraph, 2008. http://www.telegraph.co.uk/news/uknews/3705073/Power-station-break-in-sparks-security-review.html
Seitenabruf: 16.12.2008

[Kona05] R. M. Konakovsky: *Zuverlässigkeit und Sicherheit von Automatisierungssystemen*. Vorlesungsskript, IAS, Universität Stuttgart, SS 2005.

[Kowa06] W. Kowalk: *CRC-Cyclic-Redundancy-Check-Analyseverfahren mit Bitfiltern*. Universität Oldenburg, 2006. http://einstein.informatik.uni-oldenburg.de/forschung/crc/Bitfilter-Lange%20Version.pdf
Seitenabruf: 05.07.2007

[LaGö99a] R. Lauber, P. Göhner: *Prozessautomatisierung 1*. 3. Aufl., Berlin, Heidelberg: Springer, 1999.

[LaGö99b] R. Lauber, P. Göhner: *Prozessautomatisierung 2*. Berlin, Heidelberg: Springer, 1999.

[Less06] A. G. Lessig: *Linux Firewalls – Ein praktischer Einstieg*. 2. Aufl., Köln: O'Reilly, 2006.

[LiFä00] B. Liccardi, G. Färber: *Template-basierte Entwicklung von eingebetteten Systemen für Produkte mittlerer Seriengröße*. In: atp – Automatisierungstechnische Praxis 42 (2000) H. 6, S. 26–36.

[LiLa73] C. Liu, J. Layland: *Scheduling algorithms for multiprogramming in a hard real-time environment*. In: Journal of the ACM 20 (1973) H. 1, S. 46–61.

[LIN06] *LIN Specification Package, Revision 2.1*. LIN Consortium, 2006. http://www.lin-subbus.org

[Lind06] T. Linde: *Kontrollierbarer Datenfluss im Ethernet*. In: IEE 51 (2006) H. 11, S. 22–24.

[LiYa03] Q. Li, C. Yao: *Real-Time Concepts for Embedded Systems*. San Francisco, New York, Lawrence: CMP Books, 2003.

[LuWe05] S. Lucks, R. Weis: *Zur Sicherheit kryptographischer Hashfunktionen und digitaler Signaturen*.
http://www.cryptolabs.org/hash/LucksWeisSicherheitHash0305.html
Seitenabruf: 30.04.2008

[Masc08] O. im Masche: *Polizei: Keine heiße Spur von Autodieben*. Stuttgarter Zeitung vom 15.05.2008, S. 28.

[Meie06] U. Meier: *Völlig ungestört? Untersuchungen zur Störfestigkeit und Übertragungssicherheit der Bluetooth-Technologie*. In: MessTec & Automation (2006) H. 11, S. 64–65.

[Meye07] M. Meyer: *Entwurf digitaler Systeme*. Vorlesungsskript, IKR, Universität Stuttgart, SS 2007.

[Micr04] Microsoft GmbH: *Understanding Windows Firewall*. http://www.microsoft.com/windowsxp/using/security/internet/sp2_wfintro.mspx Seitenabruf: 05.09.2007

[Micr06] Microsoft GmbH: *Microsoft Internet Security and Acceleration Server 2006*. http://www.microsoft.com/isaserver/default.mspx Seitenabruf: 05.09.2007

[MiFo04] Z. Michalewicz, D. Fogel: *How to Solve It: Modern Heuristics*. 2. Aufl., Berlin: Springer, 2004.

[MoSc05] W. Morr, W. Schmidt: *IT-Security und Leittechnik*. In: atp – Automatisierungstechnische Praxis 47 (2005) H. 1, S. 30–35.

[MOST06] *MOST Specification Rev 2.5*. MOST Cooperation, 2006. http://www.mostcooperation.com/

[Müll07] O. Müller: *Überläufer – Systemeinbruch via Stack-Overflow*. In: iX – Magazin für professionelle Informationstechnik (2007) H. 2, S. 100–105.

[Muba07] H. Mubarak: *Unterstützung der Leitebene durch Selbstmanagement-Funktionalität*. GMA-Kongress 2007, Baden-Baden. In: VDI-Berichte 1980. Düsseldorf: VDI-Verlag, 2007, S. 195–294.

[MvOV96] J. Menezes, P. van Oorschot, S. Vanstone: *Handbook of Applied Cryptography*. Boca Raton: CRC Press, 1996.

[Naed03] M. Naedele: *IT-Security for Automation Systems – Motivation and Mechanisms*. In: atp – Automatisierungstechnische Praxis 45 (2003), H. 5, S. 84–91.

[Naed04] M. Naedele: *Herausforderungen bei der Sicherung von Automatisierungssystemen gegen netzwerkbasierte Angriffe*. Informatik 2004, Workshop „Sicherheit industrieller Rechnersysteme“, Ulm. In: Informatik 2004 – Informatik verbindet, Band 1, P. Dadam, M. Reichert (Hrsg.). Bonn: Köllen, 2004, S. 62–66.

[Netf07] Netfilter.org: *The netfilter.org Project*. http://www.netfilter.org Seitenabruf: 05.09.2007

[Olza07] T. Olzak: *Protect your network against fiber hacks*. 03.05.2007. http://blogs.techrepublic.com.com/security/?p=222&tag=nl.e036 Seitenabruf: 26.11.2007

[OMG07] *OMG Unified Modeling Language (OMG UML), Infrastructure, V2.1.2*. Object Management Group (OMG), 2007. http://www.omg.org

[Omni05] *OmniRemote V2.13*. Pacific Neo-Tek, 2005. http://www.pacificneotek.com/omniProfsw.htm

[OSSL07] OpenSSL.org: *About the OpenSSL Project*. http://www.openssl.org/about/ Seitenabruf: 10.09.2007

[OVPN07] OpenVPN.net: *OpenVPN Security Overview*. http://openvpn.net/security.html Seitenabruf: 10.09.2007

[Pech07] S. Pech: *Untersuchung und Evaluierung der IT-Sicherheit von Feldbussystemen mit deterministischen Buszugriffsverfahren*. Diplomarbeit Nr. 2130, IAS, Universität Stuttgart, 2007.

[Plew06] M. Plewan: *Untersuchung von Angriffsmöglichkeiten auf die Feldebene von Automatisierungssystemen*. Studienarbeit Nr. 2073, IAS, Universität Stuttgart, 2006.

[R4283G06] *Renesas 4283 Group Hardware Manual, Rev. 1.01*. Renesas Technology Corp., 2006.

[Reiß02] B. Reißenweber: *Feldbussysteme zur industriellen Kommunikation*. 2. Aufl., München: Oldenbourg Industrieverlag, 2002.

[Rent04] C. Rentrop: *Hacker, Cracker, Script-Kiddie: Eine Begriffserklärung*. In: Netzwelt, 2004. http://www.netzwelt.de/news/67766-hacker-cracker-scriptkiddie-eine-begriffserklaerung.html Seitenabruf: 12.07.2007

[Resc01] E. Rescorla: *SSL and TLS – Designing and Building Secure Systems*. Boston: Addison-Wesley, 2001.

[RFC1321] RFC 1321: *The MD5 Message-Digest Algorithm*. Internet Society, 1992. http://www.ietf.org/rfc/rfc1321.txt

[RFC3174] RFC 3174: *US Secure Hash Algorithm 1 (SHA1)*. Internet Society, 2001. http://www.ietf.org/rfc/rfc3174.txt

[RFC4301] RFC 4301: *Security Architecture for the Internet Protocol*. Internet Society, 2005. http://www.ietf.org/rfc/rfc4301.txt

[RH8G03] *Renesas H8/330 Group Hardware Manual*. Renesas Technology Corp., 2003.

[RM16C06] *Renesas M16C/62P Group Hardware Manual, Rev. 2.41*. Renesas Technology Corp., 2006.

[RM32R07] *Renesas 32192/32195/32196 Group Hardware Manual, Rev. 1.10*. Renesas Technology Corp., 2007.

[Roed02] U. Roedig: *Firewall-Architekturen für Multimedia-Applikationen*. Diss. Fachbereich 20, Technische Universität Darmstadt, 2002.

[Rusc06] T. Ruschival: *Untersuchung von Angriffsmöglichkeiten auf Feldgeräte von Automatisierungssystemen*. Studienarbeit Nr. 2105, IAS, Universität Stuttgart, 2006.

[Sail99] R. Sailer: *Sicherheitsarchitektur für mehrseitig sichere Kommunikationsdienste am Beispiel ISDN*. Diss. Institut für Nachrichtenvermittlung und Datenverarbeitung (IND), Universität Stuttgart, 1999.

[Scha05] A. Scharf: *Schneller als geahnt ist der Wurm in der Fertigung drin – Die offene Kommunikation birgt ungeahnte Gefahrenquellen*. In: VDI-Nachrichten (2005) H. 13, S. 19.

[Schn01] B. Schneier: *Secrets & Lies: IT-Sicherheit in einer vernetzten Welt*. Heidelberg: dpunkt, 2001.

[Schn06] B. Schneier: *Beyond Fear: Thinking Sensibly About Security in an Uncertain World*. 2. Aufl., Berlin: Springer, 2006.

[Schr06] W. Schröder-Preikschat: *Echtzeitsysteme – Grundlagen von Echtzeitbetriebssystemen*. Vorlesungsskript, Lehrstuhl Informatik 4 (Verteilte Systeme und Betriebssysteme), Friedrich-Alexander-Universität Erlangen-Nürnberg, WS 2006/07.

[Schw06] J. Schwager: *Ethernet erreicht das Feld*. In: Elektronic Scout (2006) H. 1, S. 30–39.

[SEI03] Software Engineering Institute: *Software Architecture for Software-Intensive Systems – Published Software Architecture Definitions*. Pittsburgh: Carnegie Mellon University – Software Engineering Institute, 2003. http://www.sei.cmu.edu/architecture/published_definitions.html Seitenabruf: 10.04.2008

[Seid04] F. Seidel: *Aspekte des Sicherheitsnachweises zum Einsatz rechnergestützter Leittechnik in kerntechnischen Anlagen*. Informatik 2004, Workshop „Sicherheit industrieller Rechnersysteme“, Ulm. In: Informatik 2004 – Informatik verbindet, Band 1, P. Dadam, M. Reichert (Hrsg.). Bonn: Köllen, 2004, S. 67–71.

[Seit03] M. Seitz: *Speicherprogrammierbare Steuerungen – Von den Grundlagen der Programmierung bis zur vertikalen Integration*. Leipzig: Hanser, 2003.

[Serv05] *Trends in der Automatisierungstechnik*. Forum Servotechnik, 2005. http://www.servotechnik.de/fachbeitraege/trends/f_beitr_00_10.htm Seitenabruf: 11.05.2007

[SHW99] B. Scherf, E. Haese, H. R. Wenzek: *Feldbussysteme in der Praxis*. Berlin, Heidelberg: Springer, 1999.

[Sick08] B. Sick: *Absicherung einer Infrarot-Kommunikationsverbindung zur Fernbedienung von Automatisierungsgeräten.* Studienarbeit Nr. 2191, IAS, Universität Stuttgart, 2008.

[Siem07] Siemens AG: *SCALANCE S Security Modules*. http://www.automation.siemens.com/net/html_00/produkte/040_ind_sec_scalance_s.htm Seitenabruf: 05.09.2007

[SiOp05] G. Sindre, A. L. Opdahl: *Eliciting Security Requirements with Misuse Cases*. Requirements Engineering Journal 10 (2005) H. 1, S. 34–44.

[SKW+98] B. Schneier, J. Kelsey, D. Whiting, D. Wagner, C. Hall, N. Ferguson: *On the Twofish Key Schedule*. Workshop Selected Areas in Cryptography ’98, Kingston. In: Proceedings of the Selected Areas in Cryptography, S. Tavares, H. Meijer (Hrsg.). London: Springer, 1998.

[SMR00] M. Schumacher, M.-L. Moschgath, U. Roedig: *Angewandte Informationssicherheit – Ein Hacker-Praktikum an Universitäten*. In: Informatik Spektrum 23 (2000) H. 3, S. 202–211.

[Snor07] Snort.org: *Snort – the De Facto Standard for Intrusion Detection/Prevention*. http://www.snort.org
Seitenabruf: 14.11.2007

[Soff03] V. Soffel: *The Evolution of FLASH Microcontrollers*. http://www.ucpros.com
Seitenabruf: 14.11.2007

[Soft03] Softing AG: *CAN- und PROFIBUS-Interfaces mit CE.NET-Support*. In: Industrial Newsletter 03/2003. Haar: Softing AG, 2003, S. 6.

[Soph07] Sophos GmbH: *Sophos Endpoint Security and Control*. http://www.sophos.de/products/enterprise/endpoint/security-and-control
Seitenabruf: 14.11.2007

[Spin06] D. Spinellis: *Code Quality – The Open Source Perspective*. Boston: Addison-Wesley, 2006.

[Ster94] D. Sterndark: *RC4 Algorithm revealed*. Posting in Newsgroup comp.security.misc, Message-ID: <sternCvKL4B.Hyy@netcom.com>, 1994.

[Syma07a] Symantec (Deutschland) GmbH: *Norton Personal Firewall 3.0*. http://www.symantec.com/de/de/home_homeoffice/products/overview.jsp?pcid=ma&pvid=npf30mac
Seitenabruf: 05.09.2007

[Syma07b] Symantec (Deutschland) GmbH: *Norton AntiVirus 2008*. http://www.symantec.com/norton/products/overview.jsp
Seitenabruf: 14.11.2007

[Szyp98] C. Szyperski: *Component Software: Beyond Object-Oriented Programming*. Boston: Addison-Wesley, 1998.

[Trau07] O. J. Traumüller: *Flexible internetbasierte Ferndiagnose eingebetteter Systeme*. Diss. IAS, Universität Stuttgart, 2007. IAS-Forschungsberichte, Bd. 3/2007.

[Trip07] Tripwire, Inc.: *Tripwire Enterprise*. http://www.tripwire.com/products
Seitenabruf: 14.11.2007

[UrhG07] *Gesetz über Urheberrecht und verwandte Schutzrechte (UrhG) §95a (2)*. Berücksichtigter Stand der Gesetzgebung: 26. Juni 2007.

[VáCa07] H. Vázquez Caramés: *CheckPoint Secure Platform Hack*. Version 1.0 (Okt. 2007). http://www.pentest.es/checkpoint_hack.pdf

[VDI3542-4] VDI/VDE 3542: *Sicherheitstechnische Begriffe für Automatisierungssysteme – Zuverlässigkeit und Sicherheit komplexer Systeme (Begriffe) – Blatt 4*. Verein deutscher Ingenieure / Verband der Elektrotechnik, Elektronik, Informationstechnik, 2000.

[Vect07] Vector Informatik GmbH: *Interfaces for CAN, LIN, FlexRay, Most*. http://www.vector-informatik.com/vi_interfaces_de,,2816.html Seitenabruf: 27.07.2007

[Vida08] J. Vidal: *No new coal – the calling card of the 'green Banksy' who breached fortress Kingsnorth*. The Guardian, 2008. http://www.guardian.co.uk/environment/2008/dec/11/kingsnorth-green-banksy-saboteur Seitenabruf: 16.12.2008

[VITO04] *VITO Oscilloscope V2.4*. VITO Technology, 2004. http://pocketland.de/product.php?prod_id=9144

[Wagn07] B. Wagner: *Industrielle Steuerungstechnik*. Vorlesungsskript, Fachgebiet Echtzeitsysteme, Leibniz Universität Hannover, SS 2007.

[Watc07] WatchGuard Technologies, Inc: *Produkte*. http://www.watchguard.com/international/de/products Seitenabruf: 05.09.2007

[WaYu05] X. Wang, H. Yu: *How to Break MD5 and Other Hash Functions*. 24th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Aarhus, Dänemark. In: Advances in Cryptology – EUROCRYPT 2005, R. Cramer (Hrsg.). Berlin: Springer, 2005, S. 19–35.

[WGU03] T. Wagner, P. Göhner, P. de A. Urbano: *Softwareagenten – Einführung und Überblick über eine alternative Art der Softwareentwicklung. Teil 1: Agentenorientierte Softwareentwicklung*. In: atp – Automatisierungstechnische Praxis 45 (2003) H. 10, S. 48–57.

[Wiki07] Wikipedia: *Block cipher – Common Algorithms*. Version vom 02.12.2007. http://en.wikipedia.org/w/index.php?title=Block_cipher&oldid=175365669

[Wiki08a] Wikipedia: *Stream ciphers – Comparison of Stream Ciphers*. Version vom 03.04.2008. http://en.wikipedia.org/w/index.php?title=Stream_cipher&oldid=202994068

[Wiki08b] Wikipedia: *Public-key cryptography*. Version vom 11.04.2008. http://en.wikipedia.org/w/index.php?title=Public-key_cryptography&oldid=204924432

[Wiki08c] Wikipedia: *List of hash functions*. Version vom 14.02.2008. http://en.wikipedia.org/w/index.php?title=List_of_hash_functions&oldid=191486033

[Wiki08d] Wikipedia: *Message authentication code*. Version vom 24.03.2008. http://en.wikipedia.org/w/index.php?title=Message_authentication_code&oldid=200527004

[Will93] R. Williams: *A Painless Guide to CRC Error Detection Algorithms*. Ross.net, 1993. http://www.ross.net/crc/download/crc_v3.txt
Seitenabruf: 26.08.2007

[Wrig08] C. Wright: *Hacking Coffee Makers*. BugTraq-Mailingliste, 2008. http://www.securityfocus.com/archive/1/493387
Seitenabruf: 21.06.2008

[WWP04] M. Wolf, A. Weimerskirch, C. Paar: *Sicherheit in Automobilen Bussystemen*. Automotive – Safety & Security 2004, Stuttgart. In: Automotive – Safety & Security: Sicherheit und Zuverlässigkeit für Automobile Informationstechnik. Aachen: Shaker, 2004, S. 9–20.

[Xele06] Xelerance Corporation: *Openswan*. http://www.openswan.org
Seitenabruf: 11.09.2007

[Zitt07] M. Zitterbart: *Netzsicherheit: Architekturen und Protokolle*. Vorlesungsskript, Institut für Telematik, Universität Karlsruhe, SS 2007.

A Anhang

A.1 Beispiel für die Ermittlung bedrohter Feldgerätefunktionalitäten und Feldbusnachrichten

Zur Verdeutlichung der in Abschnitt 8.2 eingeführten Prozedur zur Ermittlung der bedrohten Feldgerätefunktionalitäten und Feldbusnachrichten wird deren Anwendung in diesem Abschnitt anhand eines Beispiels Schritt für Schritt dargestellt. Als Beispiel dient der Misuse-Case *Dachöffnung* (vgl. Abschnitt 10.2.2) des Demonstrators *Kfz-Steuergerätenetzwerk*. Abbildung A.1 stellt noch einmal die Schritte der Prozedur dar.

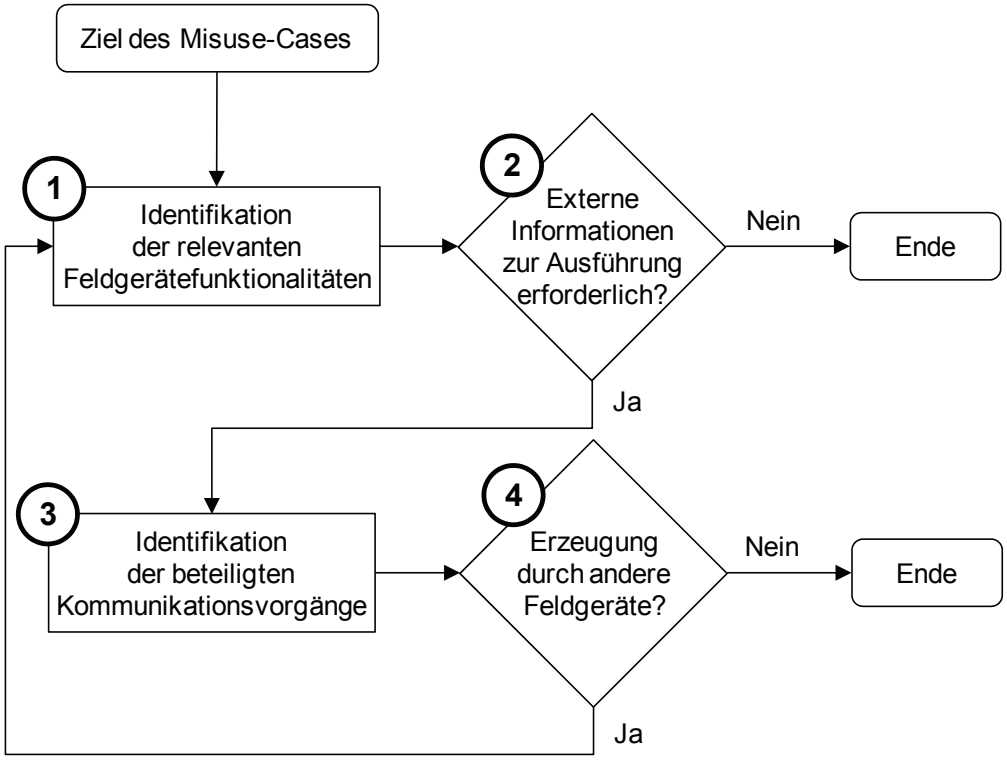

Abbildung A.1: Übersicht über die Schritte zur Ermittlung der bedrohten Feldgerätefunktionalitäten und Feldbusnachrichten

Das Ziel des Misuse-Cases besteht darin, durch unautorisierte Nutzung der Dach-öffnen-Funktionalität Zugang zum Fahrzeuginneren zu erlangen. In der folgenden Tabelle A.1 ist der Ablauf des Verfahrens beschrieben, mit dem die bedrohten Feldgerätefunktionalitäten und Feldbusnachrichten ausgehend von dem Misuse-Case-Ziel ermittelt werden. Da der *Dach-öffnen*-Vorgang von dem Dachsteuergerät geführt wird, wird dieses zum Einstieg in das Verfahren herangezogen.

Tabelle A.1: Durchgeführte Schritte zur Ermittlung der bedrohten Feldgerätefunktionalitäten und Feldbusnachrichten des Misuse-Cases *Dach öffnen*

Schritt[21]	**Beschreibung**	**Ergebnis**
1 (Dachsteuergerät)	Zum Öffnen des Dachs verfügt das Dachsteuergerät über die in Software realisierte Funktionalität *Dach öffnen*. → Schritt 2 für Funktionalität *Dach öffnen*	Funktionalität *Dach öffnen* des Dachsteuergeräts
2 (*Dach öffnen*-Funktionalität)	Die Ausführung der *Dach-öffnen*-Funktionalität wird durch die Übermittlung des entsprechenden Benutzerwunsches angestoßen. → Schritt 3 für Benutzerwunsch Da das Öffnen des Dachs nur unterhalb einer festgelegten Fahrgeschwindigkeit geschehen darf, wird die aktuelle Fahrgeschwindigkeit benötigt. → Schritt 3 für Fahrgeschwindigkeit	
3 (Benutzerwunsch)	Der Benutzerwunsch wird durch die *Dach-bewegen*-Nachricht kommuniziert. → Schritt 4 für *Dach-bewegen*-Nachricht	*Dach-bewegen*-Nachricht
4 (*Dach-bewegen*-Nachricht)	Die durch die *Dach-bewegen*-Nachricht kommunizierte Information wird durch das Armaturensteuergerät erzeugt. → Schritt 1 für Armaturensteuergerät	
1 (Armaturensteuergerät)	Die Überwachung der Bedienelemente des Armaturenbretts wird durch die in Software realisierte Funktionalität *Armaturen überwachen* durchgeführt. → Schritt 2 für Funktionalität *Armaturen überwachen*	Funktionalität *Armaturen überwachen* des Armaturensteuergeräts
2 (*Armaturen-überwachen*-Funktionalität)	Die *Armaturen-überwachen*-Funktionaliät zieht keine Informationen von anderen Steuergeräten heran. → Ende	

[21] Die unter *Schritt* angegebene Zahl bezieht sich auf die Nummerierung der Schritte in Abbildung A.1.

Schritt[21]	**Beschreibung**	**Ergebnis**
3 (Fahrgeschwindigkeit)	Die Fahrgeschwindigkeit wird durch die *Fahrgeschwindigkeit*-Nachricht übertragen. → Schritt 4 für *Fahrgeschwindigkeit*-Nachricht	*Fahrgeschwindigkeit*-Nachricht
4 (*Fahrgeschwindigkeit*-Nachricht)	Die durch die *Fahrgeschwindigkeit*-Nachricht kommunizierte Information wird durch das Motorsteuergerät erzeugt. → Schritt 1 für Motorsteuergerät	
1 (Motorsteuergerät)	Die momentane Fahrgeschwindigkeit wird durch die in Software realisierte Funktionalität *Motor regeln* bereitgestellt. → Schritt 2 für Funktionalität *Motor regeln*	Funktionalität *Motor regeln* des Motorsteuergeräts
2 (*Motor-regeln*-Funktionalität)	Die *Motor-regeln*-Funktionalität zieht keine Informationen von anderen Steuergeräten heran. → Ende	

Somit ergibt die Durchführung der Prozedur die in der Spalte *Ergebnis* von Tabelle A.1 angegebenen Funktionalitäten und Nachrichten, welche im Rahmen der Risikobewertung weiter zu betrachten sind: die Funktionalitäten des Dachsteuergeräts, des Armaturensteuergeräts und des Motorsteuergeräts sowie die *Dach-bewegen*-Nachricht und die *Fahrgeschwindigkeit*-Nachricht.

A.2 Definition der Funktion *SecurityLayer2Integrate*

Abbildung A.2 stellt die Integration des Unterscheidungsmerkmals in eine Nachricht auf der Schutzschicht 2 in Form von C-Code dar. Zunächst wird die mit der Schutzschichtkomponente verknüpfte Hashfunktion initialisiert (1). Daraufhin werden alle zu schützenden Informationen unter Nutzung der eingestellten Parameter (vgl. Abbildung A.3) in die Hashwertberechnung einbezogen (2). Daraufhin wird der Hashwert erzeugt (3) und an die korrekte Stelle in die Nachricht eingefügt (4). Schließlich wird der Dienst der darunterliegenden Schutzschicht über die Schnittstelle *ISecurityLayer1* aufgerufen (5).

```
static void SecurityLayer2Integrate(Artifact *pMsg)
{
  // Definition der lokalen Variablen (hier ausgelassen)
  // ...

  IHash->Init(pContext);                                              (1)

  for (i = 0; i < HASH_DATA_COUNT; i++) {
    IHash->Update(pContext,
      pMsg + HASH_DATA_INFO[i].Offset,                                (2)
      HASH_DATA_INFO[i].Length);
  }

  IHash->Finish(pContext, pBuffer);                                   (3)

  ElcMemcpy(pMsg + HASH_VALUE_INFO.Offset, pBuffer,                   (4)
    HASH_VALUE_INFO.Length);

  ISecurityLayer1->Integrate(pMsg);                                   (5)
}
```

Abbildung A.2: Absicherung einer Nachricht in Schutzschicht 2 als C-Code

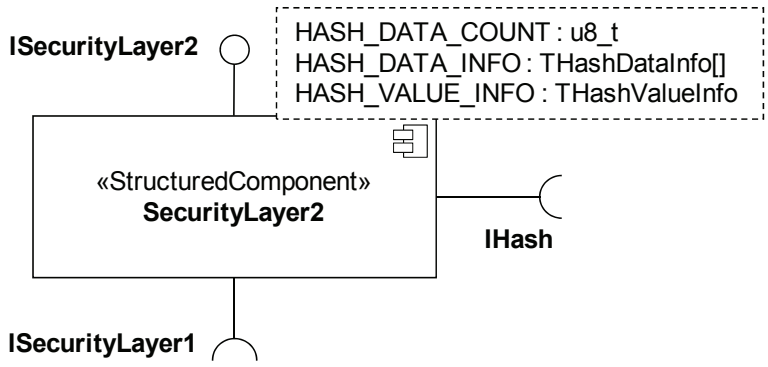

Abbildung A.3: Schutzschichtkomponente *SecurityLayer2*

A.3 Ausschnitte aus den Dateien *link.h* und *config.h*

Die Abbildung A.4 und die Abbildung A.5 zeigen Ausschnitte aus den Dateien *link.h* und *config.h*, welche von dem Softwarewerkzeug CDE generiert werden.

```
/************************************************
*   File        link.h
*   GENERATED BY COMPONENT DEVELOPMENT ENVIRONMENT
************************************************/

#ifndef __LINK_H
#define __LINK_H

/*** Component linking ********************/

/*** SecurityLayer2-related link settings ***/
#define SECURITYLAYER2_SECURITYLAYER1_LINKMODE _STATIC
#define SECURITYLAYER2_SECURITYLAYER1_LINK { &SecurityLayer1Impl }

#define SECURITYLAYER2_HASH_LINKMODE _STATIC
#define SECURITYLAYER2_HASH_LINK { &Md5HashImpl }

// Verknüpfungen von SecurityLayer1
// (hier ausgelassen)

#endif /* __LINK_H */
```

Verknüpfung zwischen Schutzschicht 2 und Schutzschicht 1

Verknüpfung zwischen Schutzschicht 2 und Schutzmechanismus MD5

Abbildung A.4: Ausschnitt aus der generierten Datei *link.h*

```
/************************************************
*   File        config.h
*   GENERATED BY COMPONENT DEVELOPMENT ENVIRONMENT
************************************************/

#ifndef __CONFIG_H
#define __CONFIG_H

/*** SecurityLayer2 parameters ***/
#define SECURITYLAYER2_HASH_DATA_COUNT          1
#define SECURITYLAYER2_HASH_DATA_INFO           { {10, 16} }
#define SECURITYLAYER2_HASH_VALUE_INFO          {26, 16}

// Parameter von SecurityLayer1, Aes, Md5
// (hier ausgelassen)

#endif /* __CONFIG_H */
```

Anzahl der Datenblöcke, die in die Hashwert-berechnung eingehen

Position und Länge des Datenblocks

Position und Länge des Hashwerts

Abbildung A.5: Ausschnitt aus der generierten Datei *config.h*

Lebenslauf

Felix Gutbrodt

Persönliche Daten

15.06.1976	geboren in Stuttgart

Schulbildung

1983 – 1987	Weilerhau-Grundschule, Filderstadt
1987 – 1994	Ev. Mörike-Gymnasium, Stuttgart
1994 – 1996	Paracelsus-Gymnasium, Stuttgart-Hohenheim, Abschluss Abitur

Wehrdienst

1996 – 1997	Fallschirmjägerbataillon 252 (Nagold) und Kommando Spezialkräfte (Calw)

Studium

1997 – 2002	Studium der Elektrotechnik an der Universität Stuttgart Studienrichtung: Informationstechnik
	Fachpraktikum und Nebentätigkeit als Softwareentwickler bei Honeywell AG in Schönaich
14.11.2002	Abschluss als Diplom-Ingenieur

Berufstätigkeit

Dezember 2002 – März 2008	Wissenschaftlicher Mitarbeiter am Institut für Automatisierungs- und Softwaretechnik der Universität Stuttgart
seit Juni 2008	Consultant bei Vector Consulting Services GmbH in Stuttgart