FAKULTÄT FÜR !NFORMATIK
Faculty of Informatics

Datensicherheit in KNX

DIPLOMARBEIT

zur Erlangung des akademischen Grades

Diplom-Ingenieur

im Rahmen des Studiums

Technische Informatik

eingereicht von

Thomas Hassler
Matrikelnummer 0425981

an der
Fakultät für Informatik der Technischen Universität Wien

Betreuung: Ao. Univ. Prof. Dr. Wolfgang Kastner
Mitwirkung: Univ.-Ass. Dipl. Ing. Lukas Krammer

Wien, 04.12.2014

(Unterschrift Verfasser) (Unterschrift Betreuung)

Erklärung zur Verfassung der Arbeit

Thomas Hassler
Senfgasse 1/16/1, 1100 Wien

Hiermit erkläre ich, dass ich diese Arbeit selbständig verfasst habe, dass ich die verwendeten Quellen und Hilfsmittel vollständig angegeben habe und dass ich die Stellen der Arbeit - einschließlich Tabellen, Karten und Abbildungen -, die anderen Werken oder dem Internet im Wortlaut oder dem Sinn nach entnommen sind, auf jeden Fall unter Angabe der Quelle als Entlehnung kenntlich gemacht habe.

(Ort, Datum) (Unterschrift Verfasser)

Abstract

Building automation systems are composed of mechanical and electrical equipment in a building. The core domains of building automation are heating, ventilation and air conditioning as well as lighting and shading. Building automation systems include security-critical services such as alarm or access control systems. Services from the security domain are typically implemented by application specific subsystems. An integration of these subsystems can be handled, if at all, at the management level. A tighter integration of the security-critical systems into an overall building automation system is desirable because of cost reduction, improvements in building control as well as easier management. Therefore, the underlying communication system has to be robust and reliable against malicious manipulations.

KNX is a well-established control network technology tailored to the special needs of buildings. Originally, it was not designed for the use in security-critical environments and security relies on physical isolation. Preventing physical access to the network by isolation is not possible in general, especially because wireless communication (KNX RF) is emerging. Therefore, the specification of the KNX Secure Application Layer (S-AL) introduces a security concept allowing the integration of security-critical services.

This thesis provides an overview of the security mechanisms of established building automation systems. Subsequently, the security concept of the recently announced KNX S-AL is analysed. An evaluation examines the S-AL by means of fulfilling security objectives and resistance against typical attacks. Moreover, the security mechanisms are compared with BACnet and ZigBee.

In order to show the feasibility of the S-AL, a prototype is implemented. Issues concerning the specification that came up during implementation are addressed. Furthermore, the performance and the memory consumption of the implementation are analysed.

Kurzfassung

Gebäudeautomation bedeutet Überwachen, Messen, Steuern und Regeln der Anlagen und Geräte der technischen Gebäudeausrüstung. Typische Anwendungen der Gebäudeautomation sind Heizungs-, Lüftungs- und Klimatechnik sowie Beleuchtung und Verschattung. Überdies umfasst ein Gebäudeautomationssystem auch sicherheitsrelevante Applikationen, wie Alarm- und Zutrittskontrollsysteme. Dienste dieser Domänen werden zumeist von eigenständigen, anwendungsspezifischen Technologien abgedeckt. Eine Integration solcher Bereiche in ein Gebäudeautomationssystem findet, wenn überhaupt, erst auf der Managementebene statt. Die engere Integration der sicherheitskritischen Systeme in das zentrale Gebäudeautomationssystem ist aufgrund reduzierter Kosten, verbesserter Gebäudesteuerung sowie einfacherer Wartung erwünscht. Das zugrunde liegende Kommunikationssystem muss dazu eine sichere Kommunikation erlauben.

KNX ist ein etablierter Standard der Gebäudeautomation, der allerdings keinerlei relevante Sicherheitsmechanismen aufweist. Die Gründe dafür liegen in der ursprünglichen Annahme, auf physische Isolation durch das Gebäude vertrauen zu können, die insbesondere durch den vermehrt aufkommenden Einsatz von drahtloser Kommunikation (KNX RF) nicht länger gegeben ist. Mit der Spezifikation des KNX Secure Application Layers (S-ALs) wird ein Sicherheitskonzept eingeführt, das die Verwendung des KNX-Standards auch in sicherheitsrelevanten Domänen ermöglichen soll.

Diese Arbeit stellt die Sicherheitsmechanismen gängiger Gebäudeautomationssysteme vor, um anschließend den KNX S-AL zu analysieren. Die Anforderungen an ein sicheres Kommunikationssystem werden für den S-AL überprüft und typische Angriffe analysiert. Darüber hinaus werden die Sicherheitsmechanismen mit jenen von BACnet und ZigBee verglichen.

Um die Realisierbarkeit des S-ALs zu zeigen, wurde eine prototypische Implementierung durchgeführt. Unklarheiten und Probleme der Spezifikation wurden im Zuge der praktischen Umsetzung festgestellt und festgehalten. Des Weiteren wurden Performance und Speicherbedarf der Implementierung untersucht.

Inhaltsverzeichnis

Abkürzungen **xi**

1 Einleitung **1**
1.1 Motivation . . . 1
1.2 Gliederung der Arbeit . . . 2

2 Gebäudeautomation **3**
2.1 Einleitung . . . 3
2.1.1 Schichtenmodell . . . 3
2.2 Sicherheit in der Gebäudeautomation . . . 6
2.2.1 Angriffe . . . 7
2.2.2 Anforderungen an eine sichere Kommunikation . . . 10

3 Stand der Technik **15**
3.1 Kryptographie . . . 15
3.1.1 Advanced Encryption Standard (AES) . . . 17
3.1.2 Betriebsmodi von Blockverschlüsselungen . . . 22
3.1.3 CBC-MAC . . . 27
3.1.4 CCM (Counter mit CBC-MAC) . . . 27
3.2 Security in der Gebäudeautomation . . . 31
3.2.1 LonWorks . . . 32
3.2.2 BACnet . . . 34
3.2.3 ZigBee . . . 36
3.2.4 Zusammenfassung . . . 38

4 KNX **41**
4.1 Konfiguration . . . 41
4.2 Topologie und Adressierung . . . 42
4.2.1 Adresstypen . . . 43
4.3 Schichten in KNX . . . 44
4.3.1 Physical Layer . . . 45
4.3.2 Data Link Layer . . . 46
4.3.3 Network Layer . . . 47

4.3.4 Transport Layer . . . 47
4.3.5 Application Layer . . . 47
4.4 Aufbau eines Telegramms . . . 49
4.5 Interworking-Modell . . . 51
4.6 Security . . . 51

5 KNX Datensicherheit **55**
5.1 Einleitung . . . 55
5.1.1 Format der sicheren Daten . . . 56
5.1.2 Schlüssel . . . 57
5.2 Analyse der Security-Mechanismen . . . 58
5.2.1 Erfüllung der Anforderungen . . . 58
5.2.2 CCM in KNX . . . 61
5.2.3 Länge des MACs . . . 63
5.2.4 Schätzung des Speicherbedarfs . . . 65
5.3 Angriffe . . . 67
5.3.1 Mithören der Kommunikation . . . 67
5.3.2 Replay-Angriff . . . 68
5.3.3 Senden von Nachrichten mit hoher Sequenznummer . . . 69
5.3.4 Denial of Service (DoS) . . . 71
5.3.5 Auswirkungen bei Erlangung von Schlüsseln . . . 74
5.3.6 Vergleich . . . 78
5.4 Diskussion der Spezifikation . . . 78
5.4.1 Verschlüsselte Speicherbereiche . . . 78
5.4.2 Fehlende Testvektoren für CCM . . . 79
5.4.3 Widersprüche und Unklarheiten . . . 80

6 Prototyp **85**
6.1 Einleitung . . . 85
6.1.1 Texas Instruments MSP430 . . . 85
6.1.2 Siemens TP-UART 2 . . . 85
6.1.3 RELIC-Toolkit und Erweiterung um AES-CCM . . . 86
6.2 Anmerkungen zur Implementierung . . . 87
6.2.1 Inhalt der Nutzdaten . . . 87
6.2.2 Initialisierungsvektor . . . 88
6.3 Sicheres KNX-Telegramm . . . 89
6.4 Performance . . . 90
6.5 Speicherbedarf . . . 94
6.6 Fazit . . . 94

7 Schlussbemerkung **97**
7.1 Zusammenfassung . . . 97
7.2 Fazit . . . 98
7.3 Weiterführende Arbeit . . . 98

Abbildungsverzeichnis **101**

Tabellenverzeichnis **103**

Literaturverzeichnis **105**

Abkürzungen

AES Advanced Encryption Standard
AIL Application Interface Layer
AL Application Layer
ANSI American National Standards Institute
APCI Application Layer Protocol Control Information
APDU Application Protocol Data Unit
ASAP Application Layer Service Access Point
ASHRAE American Society of Heating, Refrigerating and Air-Conditioning Engineers
BACnet Building Automation and Control Networks
BiBat Bidirectional Battery Driven Devices
BLS Boneh–Lynn–Shacham
BTI Bus Transceiver Interface
CBC Cipher Block Chaining
CCM Counter mit CBC-MAC
CFB Cipher Feedback
CPU Central Processing Unit
CRC Cyclic Redundancy Check
CSMA/CA Carrier Sense Multiple Access/Collision Avoidance
CTR Counter
DDC Direct Digital Control
DES Data Encryption Standard
DoS Denial of Service
E-Mode Easy Mode
ECB Electronic Codebook
ECC Elliptic Curve Cryptography
ECDSA Elliptic Curve Digital Signature Algorithm
ECIES Elliptic Curve Integrated Encryption Scheme
EEPROM Electrically Erasable Programmable Read-Only Memory
EFF Electronic Frontier Foundation
EHS European Home System
EIB Europäischer Installationsbus
ETS Engineering Tool Software
FDSK Factory Default Setup Key
FPGA Field Programmable Gate Array

FSK Frequency Shift Keying
HLK Heizung, Lüftung und Klimatechnik
HMAC Keyed-Hash Message Authentication Code
IC Integrated Circuit
IDS Intrusion Detection System
IEC International Electrotechnical Commission
IEEE Institute of Electrical and Electronics Engineers
I2C Inter-Integrated Circuit
IP Internet Protocol
ISO Internationale Organisation für Normung
IT Informationstechnologie
JTAG Joint Test Action Group
LPCI Data Link Layer Protocol Control Information
LPDU Data Link Protocol Data Unit
LTE Logical Tag Extended
M-Bus Meter-Bus
MAC Message Authentication Code
MaS Management Server
MD5 Message Digest 5
NIST National Institute of Standards and Technology
OCB Offset Codebook
OFB Output Feedback
OPC UA OPC Unified Architecture
OSI Open Systems Interconnection
P-AL Plain Application Layer
PBC Pairing-Based Cryptography
PDU Protocol Data Unit
PL110 Powerline 110
QoS Quality of Service
RF Radio Frequency
RISC Reduced Instruction Set Computer
RSA Rivest, Shamir und Adleman
S-AL Secure Application Layer
S-Box Substitutionsbox
S-Mode System Mode
SCF Security Control Field
SDU Service Data Unit
SeqNr Sequenznummer
SHA Secure Hash Algorithm
SFCC Security Failure Common Counter
SFCL Security Failure Counters on Links
SPI Synchronous Peripheral Interface
SPS Speicherprogrammierbare Steuerung

SRAM Static Random-Access Memory
SRD Short Range Devices
TCP Transmission Control Protocol
TDMA Time Division Multiple Access
TGA Technische Gebäudeausrüstung
TP Twisted Pair
TP1 Twisted Pair 1
TPCI Transport Layer Protocol Control Information
TSAP Transport Layer Service Access Point
TPDU Transport Protocol Data Unit
UART Universal Asynchronous Receiver Transmitter
UDP User Datagram Protocol
USCI Universal Serial Communication Interfaces
WAN Wide Area Network
ZDO ZigBee Device Object

KAPITEL 1

Einleitung

1.1 Motivation

Traditionell finden sich Anwendungen der Gebäudeautomation in dem Bereich Heizung, Lüftung und Klimatechnik (HLK) sowie Beleuchtung und Verschattung. Die Kosten für Betrieb und Wartung des Gebäudes sollen gesenkt und die Energieaufnahme minimiert werden. Des Weiteren ist es Aufgabe, den Nutzern des Gebäudes einen erhöhten Komfort zu bieten. Dienste aus den Bereichen Safety (Betriebssicherheit) und Security (Informationssicherheit) werden üblicherweise von eigenständigen, anwendungsspezifischen Subsystemen abgedeckt. Deren Integration findet, wenn überhaupt, erst auf der Managementebene statt. Da zusätzliche Subsysteme eine zusätzliche Infrastruktur und damit einhergehend auch höhere Kosten für Anschaffung und Wartung bedeuten, ist eine engere Integration der sicherheitskritischen Systeme in das zentrale Gebäudeautomationssystem erwünscht. Gebäudeautomationssysteme und deren Netzwerke können, wie jedes andere System und Netzwerk auch, angegriffen werden. Um die Integration der sicherheitskritischen Systeme zu ermöglichen, muss das zugrunde liegende Kommunikationssystem robust und zuverlässig gegenüber beabsichtigten, schädlichen Manipulationen sein.

Eine in der Gebäudeautomation etablierte Technologie ist der Standard KNX, dessen erste Spezifikation im Jahr 2002 veröffentlicht wurde. Es handelt sich um eine Weiterentwicklung des europäischen Installationsbusses (EIB). KNX bietet keinerlei Sicherheitsmechanismen mit Ausnahme der Autorisierung für Managementfunktionen. Die Passwörter dazu werden im Klartext übertragen, der Vorgang ist somit keineswegs sicher. Der Austausch von Prozessdaten ist durch keinerlei Maßnahmen geschützt. Der Standard kann weder Datenintegrität, Datenaktualität noch Datenvertraulichkeit garantieren. Bei der Entwicklung von KNX vertraute man auf die physische Isolation der Komponenten und Netzwerke innerhalb des Gebäudes, welche insbesondere durch den vermehrt aufkommenden Einsatz von drahtloser Kommunikation (KNX RF) nicht mehr gegeben ist. Mit der Spezifikation des KNX Secure Application Layers (S-ALs) wird ein Sicherheitskonzept eingeführt, das die Verwendung des KNX-Standards auch in sicherheitsrelevanten Domänen ermöglicht. Die Sicherheitsmechanismen des S-ALs werden in dieser Arbeit analysiert. Außerdem wurde eine prototypische Implementierung durchgeführt.

1.2 Gliederung der Arbeit

Kapitel 2 gibt einen Einblick in die Gebäudeautomation und in die Struktur ihres Schichtenmodells. Des Weiteren wird Sicherheit in der Gebäudeautomation behandelt. Dabei werden Angriffsziele der Gebäudeautomationssysteme vorgestellt und Anforderungen der Security untersucht, wobei auch auf spezifische Anforderungen der Gebäudeautomation eingegangen wird.

Der Stand der Technik hinsichtlich Kryptographie sowie der Security-Mechanismen von gängigen, offenen Gebäudeautomationssystemen wird in Kapitel 3 behandelt. Im ersten Teil wird insbesondere auf die für die Untersuchung des S-ALs relevanten Verschlüsselungsverfahren eingegangen. Abschließend wird eine Übersicht der Sicherheitskonzepte der behandelten Gebäudeautomationssysteme gegeben.

In Kapitel 4 wird der Standard KNX beschrieben, um in Kapitel 5 den KNX S-AL analysieren zu können. Die Security-Mechanismen des S-ALs werden untersucht, indem die in Kapitel 2 aufgestellten Security-Anforderungen auf ihre Einhaltung hin überprüft werden und indem typische Angriffe auf die sichere KNX-Kommunikation analysiert werden. Außerdem wird die leicht abgeänderte AES-CCM-Verschlüsselung sowie die relativ kurze Länge des eingesetzten MACs untersucht. Darüber hinaus erfolgt eine Schätzung des Speicherbedarfs des S-ALs und ein Vergleich der Sicherheitsmechanismen hinsichtlich BACnet und ZigBee.

Um die Realisierbarkeit des S-ALs zu zeigen, wurde eine prototypische Implementierung auf einem Mikrocontroller durchgeführt. In Kapitel 6 werden die Performance und der Speicherbedarf der Implementierung untersucht. Außerdem wurden Probleme festgehalten, die bei der Umsetzung der Spezifikation auftraten.

KAPITEL 2

Gebäudeautomation

2.1 Einleitung

Gebäudeautomation bedeutet Überwachen, Messen, Steuern und Regeln unter Zuhilfenahme von Anlagen und Geräten der technischen Gebäudeausrüstung (TGA). Dabei sollen die Kosten für Betrieb und Wartung des Gebäudes gesenkt und die Energieaufnahme minimiert werden. Des Weiteren ist es Aufgabe, den Nutzern des Gebäudes einen erhöhten Komfort zu bieten. Typische Anwendungen der Gebäudeautomation sind Heizung, Lüftung und Klimatechnik (HLK) sowie Beleuchtung und Verschattung. In Zweckbauten steht neben den Betriebskosteneinsparungen auch das verbesserte Gebäudemanagement im Vordergrund. Von zentraler Stelle aus kann auf die Anlagen des Gebäudes zugegriffen werden, wodurch einerseits Änderungen im Regelbetrieb erleichtert und andererseits die rechtzeitige Behebung von Störungen, auch von einem anderen Ort aus, mit minimalem personellen Aufwand ermöglicht werden. Ein nicht zu vernachlässigender Punkt bei Zweckbauten, wie Bürogebäuden, ist außerdem der Gewinn an Komfort und daraus resultierende Effekte, wie erhöhte Motivation und Produktivität der Mitarbeiter.

Gebäudeautomation ist in Einfamilienhäusern mit nur wenigen Komponenten bis hin zu industriell genutzten Gebäuden und Bürogebäuden mit mehreren tausenden Kontrollpunkten anzutreffen. Die eingesetzten Sensoren, Aktoren, Bedienelemente und andere Einheiten sind miteinander vernetzt. Auf Grund der räumlichen Trennung und der funktionalen Anforderungen werden vermehrt verteilte Systeme eingesetzt. Im privaten Bereich und bei Smart Homes spielen die Erhöhung des Komforts sowie der Wohn- und Lebensqualität eine zentrale Rolle. Neben den typischen Anwendungen der Gebäudeautomation werden hier oft auch Haushaltsgeräte (weiße Ware) und Unterhaltungselektronik (braune Ware) integriert.

2.1.1 Schichtenmodell

Die Kommunikation in Gebäudeautomationssystemen wird in drei Ebenen unterteilt, die Feldebene, die Automationsebene und die Managementebene [29]. Abbildung 2.1 zeigt dieses 3-Schichtenmodell. In der Feldebene findet die Interaktion mit der Umwelt statt. Mit Sensoren,

wie Bewegungsmeldern, Temperatur- oder Helligkeitssensoren, werden Daten von der Umgebung gesammelt und Aktoren für beispielsweise die Beleuchtungs-, Heizungs-, Klima- und Lüftungsanlage geschalten. Die Daten der Sensoren und Aktoren werden typischerweise mit Feldbussystemen übertragen. Das Datenaufkommen ist in der Feldebene üblicherweise gering. Die hier gesammelten Daten werden an die Automationsebene weitergegeben, wo sie von DDC-Stationen (Direct Digital Control) verarbeitet werden. DDC-Stationen übernehmen die Regelung einzelner Gebäudeteile bis hin zur individuellen Regelung von Räumen. Sie steuern die Aktoren auf Grund der von der Feldebene übermittelten Daten und der von der Managementebene gelieferten Vorgaben. Die Daten der Automationsebene werden an die Managementebene übergeben (vertikale Kommunikation) oder können auch an weitere Geräte innerhalb der Automationsebene weitergegeben werden (horizontale Kommunikation). Auf der Managementebene sind sämtliche Informationen des Systems verfügbar. Die Anlage kann überwacht werden und ein manueller Eingriff in das System ist möglich. Auf dieser Ebene laufen alle Daten zusammen, weshalb Netzwerke mit hoher Bandbreite eingesetzt werden. Die Daten werden von einem Server erfasst und archiviert. Der Server dient außerdem dazu, eigenständige Systeme der Gebäudeausrüstung, wie Gefahrenmeldeanlagen oder Zugangskontrollsysteme, einzubinden. Über ein Gateway kann in der Managementebene zusätzlich die Verbindung zu WANs (Internet) hergestellt werden. Die Aufgaben der Feldebene erfordern eine verteilte Struktur und auch die Automationsebene ist typischerweise als solche aufgebaut. Die Managementebene besteht oft aus einer zentralen Einheit, kann aber auch als verteiltes System realisiert werden.

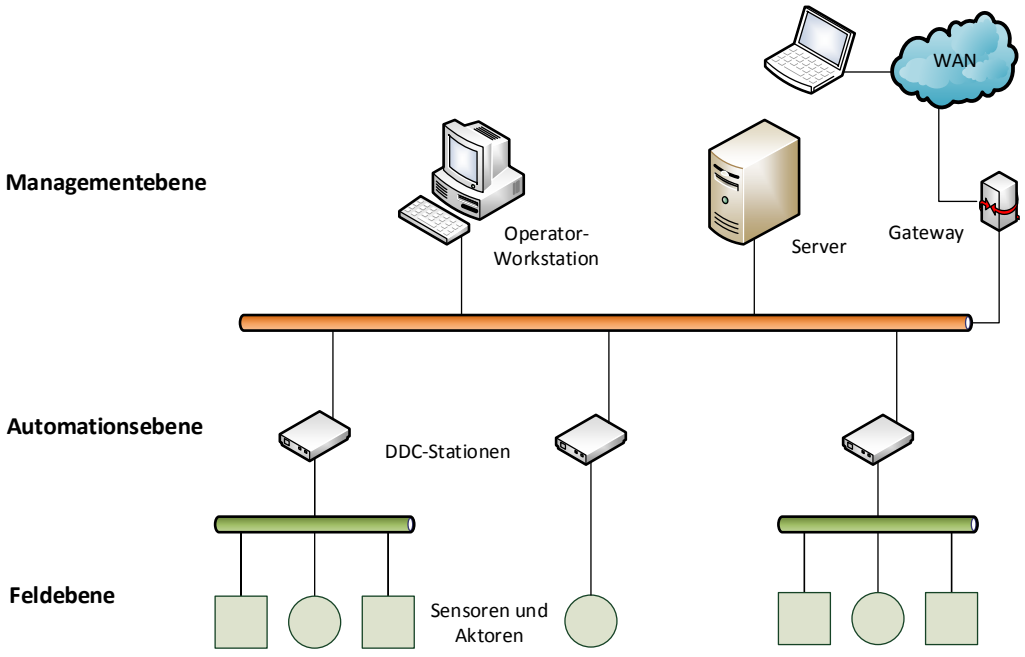

Abbildung 2.1: 3-Schichtenmodell der Gebäudeautomation [29, 59]

Diese klassische Aufteilung in Feld-, Automations- und Managementebene kann heutzutage als flachere, zweischichtige Architektur implementiert werden [19, 59]. Das ist einerseits durch intelligente Feldgeräte, die mehr Funktionalität bieten, und andererseits durch den Einzug von Informationstechnologie und ihrer Infrastruktur in die Automationsebene möglich. Die Funktionen der früheren Automationsebene wandern nun in die Feldgeräte oder werden von den Managementgeräten übernommen. Die intelligenten Feldgeräte, bestehend aus Sensoren, Aktoren und Steuergeräten, übernehmen Aufgaben der DDC-Stationen der bisherigen Automationsebene. Die DDC-Stationen wiederum werden mit besseren Netzwerkschnittstellen ausgestattet und für Funktionen der Managementebene verwendet. Das führt zu der flacheren Hierarchie, bei der intelligente Feldgeräte über ein Gateway mit einem Backbone mit einer hohen Datenrate verbunden sind, wobei sich für den Backbone das Internet Protocol (IP) als Netzwerkprotokoll durchgesetzt hat. In der Feldebene sind Kosteneffizienz, Flexibilität und Robustheit die wichtigsten Merkmale, weshalb hier nach wie vor Feldbussysteme mit einer niedrigen Datenrate von wenigen Kilobits pro Sekunde eingesetzt werden. Abbildung 2.2 veranschaulicht diese zweischichtige Architektur.

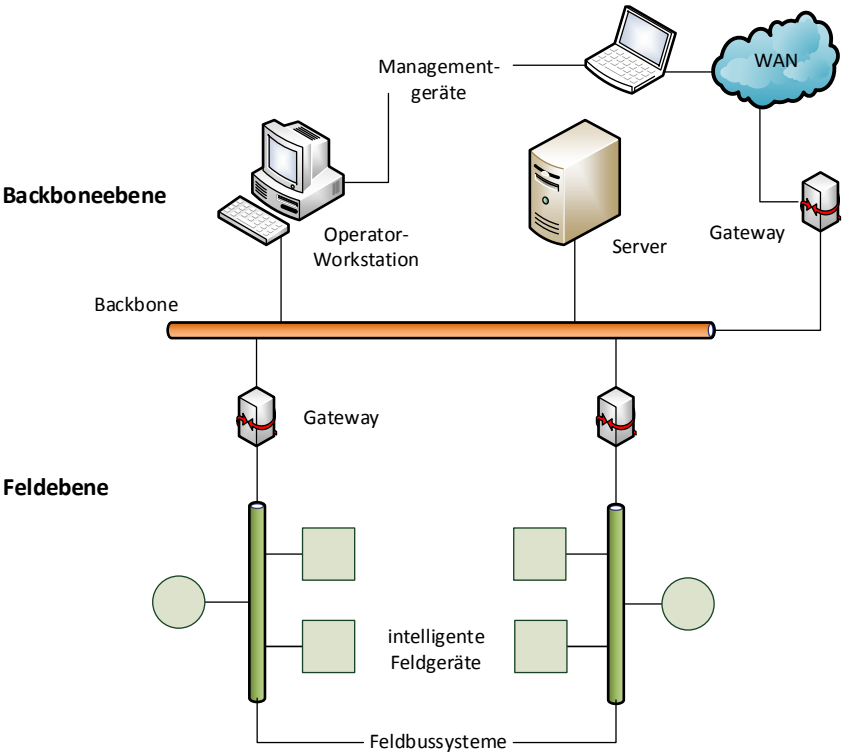

Abbildung 2.2: 2-Schichtenarchitektur der Gebäudeautomation [21, 59]

2.2 Sicherheit in der Gebäudeautomation

Die typischen Anwendungen in der Gebäudeautomation sind HLK sowie Beleuchtung und Verschattung. Dienste aus den sicherheitsrelevanten Domänen, wie Zugangskontrollsysteme, Einbruchsicherung oder Feueralarmsysteme, werden üblicherweise von eigenständigen, anwendungsspezifischen Subsystemen abgedeckt. Das geschieht nicht zuletzt wegen der mangelhaften oder gar komplett fehlenden Sicherheitsmechanismen vieler Gebäudeautomationssysteme. Eine Integration solcher Subsysteme findet, wenn überhaupt, erst auf der Managementebene statt. Zusätzliche Subsysteme bedeuten jedoch auch eine zusätzliche Infrastruktur in Form von Netzwerken und eine höhere Anzahl an Knoten. Somit ist eine Integration der sicherheitskritischen Systeme in das zentrale Gebäudeautomationssystem wünschenswert. Eine höhere Integration mindert die Kosten für Anschaffung und Wartung. Zusätzlich werden komplexere Funktionen ermöglicht, da auf Grund der Informationsfusion auf Daten verschiedener Sensoren zugegriffen werden kann.

Im deutschen Sprachgebrauch fasst der Begriff „Sicherheit“ die beiden Themen *Safety* und *Security* zusammen, die jedoch unterschiedliche Gebiete der Sicherheit sind. Safety und Security sind allerdings auch miteinander korreliert, denn ohne Security existiert keine wirkliche Safety, und Lücken in der Security gefährden die Safety. Security muss somit als notwendige Voraussetzung für Safety gesehen werden [41].

Safety beschreibt die Betriebssicherheit eines Systems, also die fehlerfreie Ausführung beziehungsweise das Erkennen eines Fehlers, um anschließend das System wieder in einen sicheren Zustand zu bringen. Safety bezieht sich speziell auf die Ablauf- und Ausfallsicherheit eines Systems zum Schutz von Mensch, Umwelt und dem System selbst. Safety-kritische Dienste, wie Feueralarme oder Alarme in Aufzügen, sind Dienste, deren Zweck es ist, die Menschen vor direkten oder indirekten unbeabsichtigten Auswirkungen auf die Gesundheit zu bewahren [40].

Security hingegen bezeichnet die Informationssicherheit eines Systems, wofür die drei Schlüsselziele Vertraulichkeit, Integrität und Verfügbarkeit eingeführt werden [46, 54]. Vertraulichkeit stellt sicher, dass private Informationen nur für autorisierte Benutzer zugänglich sind. Integrität garantiert, dass Informationen und Programme nur von autorisierten Benutzern geändert werden können und dass ein System seine bestimmungsgemäße Funktion auf unbeeinträchtigte Art und Weise ausführen kann. Verfügbarkeit gewährleistet, dass das System unverzüglich arbeitet und dass autorisierten Benutzern nicht der Zugang verwehrt wird. Diese drei Konzepte formen ein Dreieck, das in Abbildung 2.3 zu sehen ist, und verkörpern die fundamentalen Zielsetzungen der Security für Daten und Dienste.

Durch Anwendung von Gegenmaßnahmen wird das System vor beabsichtigten, schädlichen Angriffen geschützt. Gebäudeautomationssysteme und deren Netzwerke können, wie jedes andere System und Netzwerk auch, angegriffen werden. Das kann beispielsweise durch Abhören des Datentransfers oder das anschließende Einschleusen von Nachrichten geschehen. Auf diese Art und Weise könnte beispielsweise ein Zugangskontrollsystem abgehört und derart manipuliert werden, dass unerlaubter Zugang zum Gebäude erlangt werden kann. Security-kritische Dienste schützen Informationen oder Ressourcen durch Gegenmaßnahmen vor beabsichtigten, böswilligen Angriffen. Eine physische Zugangskontrolle wäre ein Beispiel für einen security-kritischen Dienst [40].

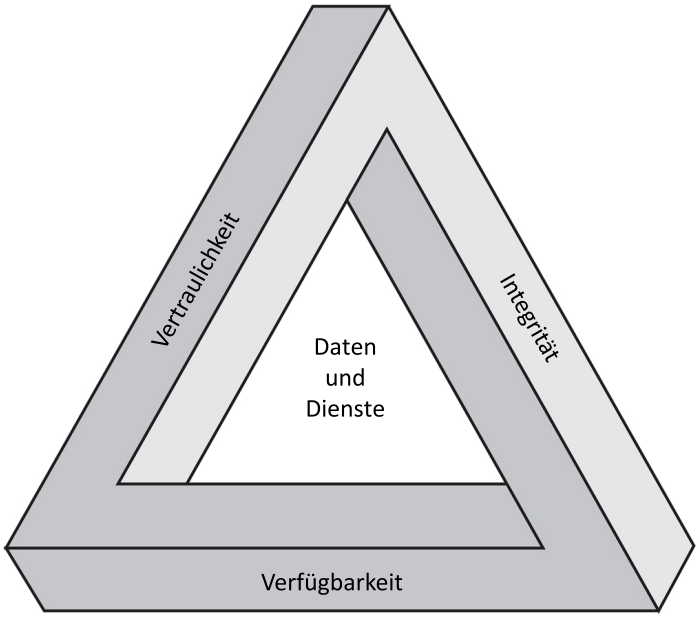

Abbildung 2.3: Dreieck der Anforderungen an die Security [54]

Bei der Entwicklung der Gebäudeautomationssysteme wurde anfangs der Sicherheitsaspekt vernachlässigt. Man hat sich auf die physische Isolation der Komponenten und der Netzwerke durch das Gebäude selbst verlassen. Diese physische Isolation ist in öffentlichen Gebäuden und besonders durch den mittlerweile vermehrt aufkommenden Einsatz an drahtloser Kommunikation in der Gebäudeautomation jedoch nicht immer gegeben, wodurch Security-Maßnahmen nötig werden. Drahtlose Technologien haben auf Grund der wegfallenden Verkabelung den Vorteil von reduzierten Installationskosten. Dadurch wird es auch möglich, Sensoren und Aktoren an Orten zu platzieren, die aus ästhetischen oder Sicherheitsgründen keine Verkabelung erlauben. In der Gebäudeautomation sind die Geräte über weite Strecken verteilt angebracht, was den Einsatz von drahtloser Kommunikation besonders attraktiv macht. Drahtlose Installationen können zudem einfacher erweitert und verändert werden, wodurch eine Anpassung an geänderte Anforderungen erleichtert wird. Des Weiteren können beim Einsatz von drahtlosen Netzwerken mobile Geräte wie Smartphones einfacher in Automationssysteme eingebunden werden [48].

2.2.1 Angriffe

Bei Angriffen auf ein System werden Vulnerabilitäten ausgenutzt, um die Sicherheit des Systems zu verletzen [51]. Vulnerabilitäten sind Fehler oder Schwachstellen im Design, der Implementierung oder der Ausführung und Verwaltung des Systems. Eine Bedrohung ist eine mögliche Gefahr, die eine Vulnerabilität ausnutzen kann [51]. Angriffen auf ein System können Gegenmaßnahmen entgegengesetzt werden. Bedrohungen werden blockiert, indem die Vulnerabilitäten kontrolliert werden. Gelingt diese Kontrolle nicht, so treten die Konsequenzen der Bedro-

hung ein. Die Kontrolle der Vulnerabilitäten ist eine Schutzmaßnahme und kann eine Aktion, Prozedur, Technik oder ein Gerät sein, das die Vulnerabilität beseitigt oder verringert. Jemand der eine Vulnerabilität ausnutzt, verübt einen Angriff auf das System [46]. Es besteht ein Unterschied zwischen einem Angriff und einer Bedrohung. Ein Angriff ist die Aktion, mit der versucht wird, die Sicherheit des Systems zu verletzen, während die Bedrohung das Potential für eine Verletzung der Sicherheit ist, die möglicherweise nie ausgenutzt wird [17]. In Abbildung 2.4 werden die Begriffe an einem Beispiel veranschaulicht.

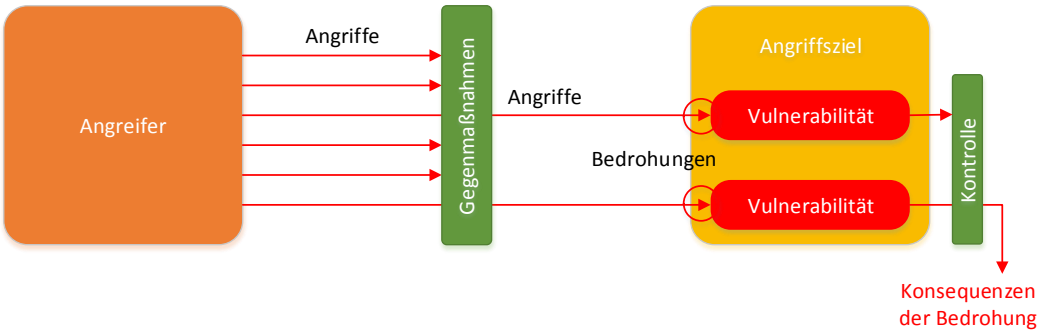

Abbildung 2.4: Angriffe, Bedrohungen und Vulnerabilitäten [17, 46, 51]

Angriffsziele

In der Gebäudeautomation können nach der zweischichtigen Architektur in Abbildung 2.2 fünf verschiedene Ziele angegriffen werden [17]:

- Der *Feldbus* und die Daten, die über diesen übertragen werden, können Ziel eines Angriffs sein.
- Der *Backbone* verbindet die einzelnen Feldbussysteme miteinander. Ein Zugang zum Backbone durch einen Angriff hat somit den Gewinn der Daten des ganzen Systems als Folge.
- *Intelligente Feldgeräte* in Form von Sensoren, Aktoren und Steuergeräten können Ziel eines Angriffs sein. Es können die Benutzeranwendungen manipuliert werden und Konfigurationsparameter sowie Kontrolldaten, wie die Werte der Aktoren, können geändert werden.
- *Gateways* verbinden die Netzwerksegmente miteinander. Über sie kann auch die Verbindung mit fremden Netzwerken, wie dem Internet, hergestellt werden. Wird ein Gateway erfolgreich angegriffen, so kann sich der Angreifer darüber nicht autorisierten Zugang zu den Daten verschaffen.
- Über *Managementgeräte*, die zur Konfiguration und Wartung benutzt werden, kann ein Angreifer Zugriff auf die Feldgeräte oder Gateways erlangen, indem die Zugangsberechtigungen des angegriffenen Managementgerätes benutzt werden.

Prinzipiell gibt es zwei verschiedene Möglichkeiten, um sich unerlaubten Zugang zu den Anwendungen eines Gebäudeautomationssystems zu verschaffen [17]. Es kann ein Angriff auf die Geräte selbst oder auf die Netzwerke erfolgen.

Angriffe auf Geräte Diese Angriffe werden weiter in drei Arten unterteilt [17, 47]:

- *Physische Angriffe* beeinträchtigen die Geräte selbst, was durch Austausch des ganzen Gerätes oder von Komponenten davon, wie dem Speicher, erfolgen kann. Es kann auch über eine lokale Schnittstelle auf den Speicher eines Gerätes zugegriffen werden, um auf diesem Weg Daten auszulesen. Es handelt sich dabei um invasive Angriffe, bei denen ein Zugriff auf das Gerät möglich sein muss, um das System zu beobachten, zu manipulieren und darin einzugreifen. Neben dem Austausch von Komponenten oder dem Zugriff auf den Speicher ist Microprobing, bei dem geladene Teilchen auf das Objekt gestrahlt werden, ein weiteres Beispiel für einen invasiven Angriff. Diese Art von Angriffen sind schwierig durchzuführen, weil einerseits Zugang zu den Komponenten und andererseits teures Equipment erforderlich sind.

- Bei *Seitenkanalangriffen* kann durch die Beobachtung des Gerätes auf Interna geschlossen werden. Beispiele hierfür sind Zeit-, Leistungs- und Fehlerverhaltensanalysen oder Analysen der elektromagnetischen Abstrahlung. Die Durchführung dieser Analysen lässt Rückschlüsse auf die vom Prozessor ausgeführten Operationen zu. Als konkretes Beispiel wird bei Zeitanalysen die Rechenzeit von kryptographischen Algorithmen gemessen, wodurch nach und nach auf den geheimen Schlüssel geschlossen werden kann.

- *Softwareangriffe* zielen darauf ab, über Kommunikationskanäle Schwachstellen in der Software einer Komponente auszunutzen. Der Zugang zur Software kann dabei über das Netzwerk oder über lokale Schnittstellen am Gerät erlangt werden. Es können beispielsweise Algorithmen inkorrekt implementiert sein oder Ausnahmebehandlungen fehlerhaft sein. Weitere denkbare Angriffe sind das Einschleusen von ausführbarem Code (*Code Injection*) oder das Ausnutzen von Konfigurations-, Wartungs- und Debugmechanismen.

Angriffe auf Netzwerke Es gibt vier Arten von Angriffen auf ein Netzwerk [17, 46]. Der Zugang zum Netzwerk kann dabei durch Zugriff auf das Übertragungsmedium stattfinden, was bei drahtlosen Netzwerken leicht zu bewerkstelligen ist, oder es wird die Netzwerkschnittstelle eines anderen Gerätes, wie einem Feldgerät oder Gateways, dazu benutzt [21]:

- *Abfangen (interception)* bedeutet, dass ein nicht autorisierter Zugriff auf die ausgetauschten Daten erfolgt. Das kann durch das „Anzapfen“ einer Leitung oder dem Mitlesen von drahtlosem Datenverkehr mit einem Sniffer geschehen.

- Durch eine *Störung (interruption)* wird ein Teil des Systems nicht mehr verfügbar oder unbrauchbar. Der Angreifer stört die Kommunikation zwischen den Teilnehmern, was durch DoS-Angriffe erreicht werden kann, indem Komponenten des Netzwerks gezielt überlastet werden.

- Bei der *Modifizierung (modification)* wird nicht nur auf Daten zugegriffen, sondern die Nachrichten werden bei der Übertragung über das Netzwerk auch verändert. Das kann durch Man-in-the-Middle-Angriffe bewerkstelligt werden kann.

- Das *Herstellen (fabrication)* von Informationen ist eine Methode, bei der alte Nachrichten wiederverwendet werden oder unechte Nachrichten in das System eingeschleust werden, um dieses zu manipulieren.

Zusammenfassung der Angriffe Abbildung 2.5 fasst die Möglichkeiten der Angriffe auf Gebäudeautomationssysteme zusammen.

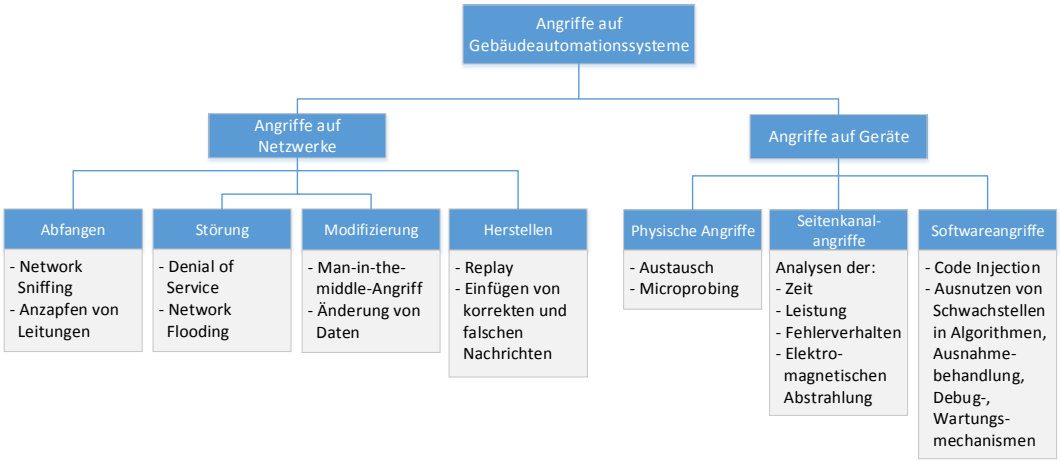

Abbildung 2.5: Angriffe auf Gebäudeautomationssysteme [17]

Diese Angriffe können zudem generell in aktive und passive Angriffe unterteilt werden [44]. In passiven Angriffen wird die Kommunikation zwischen den Teilnehmern mitgehört, dabei werden allerdings keine Datenpakete geändert und somit auch keine Auffälligkeiten für böswillige Aktivitäten geliefert. Unlesbare verschlüsselte Daten können hierbei ebenfalls brauchbare Informationen liefern, indem die Header der Pakete sowie deren Größe und andere Metainformationen analysiert werden und Informationen über Kommunikationsmuster gewonnen werden. Bei aktiven Angriffen werden die Nachrichten nicht nur mitgehört, sondern auch geändert, gelöscht oder es werden neue Nachrichten eingefügt. Es wird dabei also aktiv in die Kommunikation eingegriffen. Da bei passiven Angriffen keine Modifizierung von Daten stattfindet, sind diese Angriffe nur schwer oder im Fall von drahtloser Kommunikation, wo kein physischer Zugang zum Übertragungsmedium erforderlich ist, unmöglich zu erkennen und sie erfolgen anonym.

2.2.2 Anforderungen an eine sichere Kommunikation

Besonders drahtlose Netzwerke sind auf Grund ihrer Broadcast-Eigenschaft des Übertragungsmediums anfällig für Angriffe. Um sicherheitskritische Dienste integrieren zu können, muss

das zugrunde liegende System zuverlässig und stabil gegenüber den Bedrohungen sein. Für einen sicheren Datenaustausch müssen verschiedene Anforderungen der Security erfüllt werden [17, 18, 21, 60]:

- *Authentifizierung*: Die Teilnehmer einer sicheren Kommunikation müssen ihre Identität nachweisen. Die Authentifizierung verhindert, dass eine böswillige Entität vorgibt, eine vertrauenswürdige Einheit zu sein. Der Empfänger kann prüfen, ob die Daten tatsächlich vom behaupteten Sender stammen. Das kann durch symmetrische Kryptosysteme, in denen Sender und Empfänger einen gemeinsamen geheimen Schlüssel besitzen, oder digitale Signaturen, die einen geheimen und einen öffentlichen Schlüssel einsetzen, erreicht werden.

- *Autorisierung*: Nach der Authentifizierung der Entität wird geprüft, ob sie die nötigen Zugriffsrechte besitzt, um der Kommunikationsbeziehung beizutreten. Wird in Folge der Autorisierung festgestellt, dass die Zugriffsrechte der Entität unzureichend sind, so wird die Teilnahme an der Kommunikation verweigert.

- *Datenvertraulichkeit* stellt sicher, dass vertrauliche Informationen nicht offengelegt werden und nur autorisierte Entitäten Zugang dazu haben. Erreicht wird das durch Verschlüsselung der zu übertragenen Daten, sodass nur die jeweiligen Empfänger der Daten den Klartext erlangen können. Je nach Art der Anwendung werden sensible Daten ausgetauscht, was einen sicheren Kanal erfordert. Dieser wird auch für die Verteilung der Schlüssel benötigt.

- *Datenintegrität* gewährleistet, dass die empfangenen Nachrichten während der Übertragung nicht unerwünscht und unbemerkt verändert wurden. Das kann durch einen MAC am Ende der Nachricht erreicht werden, der über die Daten berechnet wird, für die Datenintegrität gefordert ist.

- *Datenursprungsauthentifizierung* ist eine strengere Form der Datenintegrität. Es wird überprüft, ob die Quelle der empfangenen Daten die ist, die sie behauptet zu sein [55].

- *Datenaktualität* schützt vor der Wiederverwendung von bereits gesendeten Nachrichten und garantiert, dass die Daten aktuell sind. Es kann weiters zwischen schwacher und starker Datenaktualität unterschieden werden. Schwache Datenaktualität bietet eine Halbordnung der Nachrichten, enthält aber keine weiteren Zeitinformationen, wie Informationen über die Verzögerung. Bei schwacher Datenaktualität kann der Empfänger einer Nachricht auf Grund eines Zählers feststellen, ob die empfangene Nachricht nach der letzten korrekt empfangenen Nachricht, die einen niedrigeren Zählwert hatte, gesendet wurde. [45] Starke Datenaktualität bietet eine Totalordnung der Nachrichten sowie Verzögerungsinformationen. Diese Art der Datenaktualität kann in Protokollen zur Zeitsynchronisation erlangt werden [1].

- *Datenverfügbarkeit* sorgt dafür, dass den autorisierten Entitäten der Zugang zu den Daten nicht verweigert wird. Diese Anforderung schließt beabsichtigte oder unbeabsichtigte

Versuche ein, eine nicht autorisierte Löschung von Daten durchzuführen oder auf andere Art eine Verweigerung von Diensten oder Daten herbeizuführen [55].

Spezifische Anforderungen

Es existiert bereits eine Vielzahl an etablierten Security-Mechanismen, die beispielsweise in der Informationstechnologie (IT) eingesetzt werden. Die Heim- und Gebäudeautomation stellt jedoch zusätzlich für diese Domäne spezifische Security-Anforderungen [17, 18, 21, 60]. Deshalb ist es nicht ohne weiteres möglich, die bereits existierenden Mechanismen aus anderen Domänen, wie der Informationstechnologie, zu übernehmen, da diese Security-Mechanismen auf die IT-Welt zugeschnitten sind. Die für die Heim- und Gebäudeautomation spezifischen Anforderungen sind:

- *Limitierte Ressourcen*: Die in Gebäudeautomationssystemen eingesetzten Komponenten haben aus Kosten- und Platzgründen typischerweise limitierte Ressourcen, was Speicher und Rechenleistung betrifft. Neben dem Speicher und der Rechenleistung ist auch die Energieversorgung limitiert und erfolgt meist über den Bus oder Batterie. Der Einsatz von Security-Mechanismen erfordert zusätzliche Ressourcen, vor allem kryptographische Algorithmen sind rechenintensiv. Durch die zusätzlich benötigte Rechenleistung sowie dem Overhead, der durch Security-Funktionen zu Stande kommt, und durch das sichere Speichern von Parametern (zum Beispiel Schlüssel) wird mehr Energie verbraucht. Es ist somit nötig, eine vernünftige Balance zwischen dem Einsatz von Security-Diensten und den verfügbaren Ressourcen zu finden.

- *Skalierbarkeit*: Die Skalierbarkeit der Mechanismen spielt eine wichtige Rolle, weil sich in der Gebäudeautomation üblicherweise eine Vielzahl an Komponenten im Netzwerk befinden. Das Netzwerk kann aus einigen wenigen Management-Geräten, mehreren Gateways und tausenden Sensoren, Aktoren und Steuergeräten bestehen. In der Gebäudeautomation erfolgt die Kommunikation häufig nach dem Peer-to-Peer-Schema ohne zentraler Instanz. In der Informationstechnologie hingegen ist die Anzahl der Geräte in einer Kommunikation relativ klein und es findet meist das Client/Server-Modell Anwendung.

- *Feldbussysteme ohne IP*: Die Sicherheitsmechanismen der Informationstechnologie sind auf den Einsatz in IP-Netzwerken ausgerichtet. IP kommt derzeit in der Gebäudeautomation nur in der Backboneebene zum Einsatz. In der Feldebene werden auf Grund von Robustheit, Flexibilität und Kosten nach wie vor Feldbussysteme eingesetzt.

- *QoS-Parameter*: Die benötigten QoS-Parameter eines Netzwerks im Bereich der Gebäudeautomation unterscheiden sich von jenen der IT-Welt. Das Datenvolumen in IT-Netzwerken ist üblicherweise hoch, wohingegen Kontrolldaten in der Domäne der Gebäudeautomation nur einige Bytes betragen. Weiters können die in der Gebäudeautomation übertragenen Daten weiche Echtzeitanforderungen aufweisen, was bedeutet, dass Eingaben typischerweise innerhalb der Zeitanforderungen abgearbeitet werden, eine Zeitüberschreitung jedoch keine katastrophalen Auswirkungen hat. Weiche Echtzeitanforderungen können beispielsweise durch die geforderte Reaktionszeit eines Beleuchtungssystems auftreten. Zu-

dem müssen in der Domäne der Gebäudeautomation QoS-Eigenschaften, wie Verlässlichkeit und Ordnung der Nachrichten, beachtet werden.

- *Unbeaufsichtigter Betrieb*: Durch den Einzug von Verbindungen zu außenstehenden Netzwerken, wie dem Internet über Gateways sowie dem Einsatz von IP, und insbesondere durch drahtlose Technologien ist die physische Isolation der Netzwerke in der Domäne der Gebäudeautomation nicht mehr gegeben. Geräte, wie Sensoren und Aktoren, können in Umgebungen platziert sein, die offen für Umwelteinflüsse und mögliche Angriffe sind. Durch die Fernverwaltung der Komponenten sind physische Angriffe nur schwer zu entdecken. Je länger eine Komponente unbeaufsichtigt arbeitet, desto größer ist die Wahrscheinlichkeit, dass sie kompromittiert wird. Man kann sich somit nicht nur auf die physische Sicherheit verlassen.

- *Lange Lebensdauer*: Gebäudeautomationssysteme müssen über einen Zeitraum von vielen Jahren oder Jahrzehnten funktionstüchtig bleiben. Es muss damit gerechnet werden, dass im Laufe der Zeit Vulnerabilitäten auftreten, die zu beheben sind. Somit muss die Möglichkeit bestehen, die Komponenten zu warten und zu aktualisieren, was wiederum Möglichkeiten für Angriffe bietet. Die Wartungsmechanismen müssen ebenfalls gegen nicht autorisierte Benutzung geschützt werden.

KAPITEL 3

Stand der Technik

Dieses Kapitel behandelt den Stand der Technik im Bereich der Security, der für die Analyse des KNX Secure Application Layer (S-AL) relevant ist. Zuerst wird auf die Kryptographie eingegangen und das symmetrische Verschlüsselungsverfahren Advanced Encryption Standard (AES) und seine Betriebsmodi untersucht, wobei auch deren Sicherheit behandelt wird. Daraufhin wird CCM beschrieben, ein Betriebsmodus für Blockverschlüsselungen, wie AES, der Authentifizierung und Integrität sowie Vertraulichkeit vereint. AES und CCM werden unter anderem auch im KNX S-AL eingesetzt. Im zweiten Teil des Kapitels werden die Security-Mechanismen von wichtigen Gebäudeautomationsstandards untersucht und abschließend einander gegenübergestellt.

3.1 Kryptographie

Kryptographie beschäftigt sich mit mathematischen Methoden in Zusammenhang mit den Zielen der Informationssicherheit, wie Vertraulichkeit, Datenintegrität, Authentifizierung und Datenursprungsauthentifizierung [38]. Die Informationen sollen mittels Kryptographie vor böswilligen Eingriffen geschützt werden. Beispiele für kryptographische Methoden, die für Informationssicherheit sorgen, sind Verschlüsselung, Hashfunktionen und digitale Signaturen.

Bei der Verschlüsselung wird ein lesbarer Text, genannt Klartext, unter Verwendung eines Verschlüsselungsverfahrens in einen nicht mehr verständlichen Geheimtext umgewandelt. Zur Verschlüsselung und Entschlüsselung werden ein oder mehrere Schlüssel verwendet. Prinzipiell unterscheidet man zwischen *symmetrischer* und *asymmetrischer* Verschlüsselung. Bei symmetrischen Verschlüsselungsverfahren werden die Ver- und Entschlüsselung mit demselben Schlüssel durchgeführt. Asymmetrische Verschlüsselungsverfahren bestehen erst seit Ende der 1970er Jahre. Sie benutzen einen öffentlichen Schlüssel, der für jeden zugänglich ist, und einen geheimen privaten Schlüssel. Der öffentliche Schlüssel ermöglicht es jedem, Daten für den Inhaber des privaten Schlüssels zu verschlüsseln, dessen digitale Signaturen zu prüfen oder ihn zu authentifizieren. Mit dem privaten Schlüssel können die Daten entschlüsselt werden, die mit dem

öffentlichen Schlüssel verschlüsselt wurden, und es können digitale Signaturen erzeugt werden. Ein Vorteil der symmetrischen Verfahren ist die relativ kurze Länge der Schlüssel. Die Übergabe des geheimen Schlüssels stellt jedoch ein Sicherheitsproblem dar. Verglichen mit symmetrischen Verfahren von offensichtlich gleicher Sicherheit haben asymmetrische Verschlüsselungsverfahren einen relativ hohen rechnerischen Aufwand.

Whitfield Diffie und Martin Hellman waren mit ihrem 1976 veröffentlichten Artikel „*New Directions in Cryptography*“ [9] die Wegbereiter der *Public-Key*-Kryptographie. Sie beschäftigten sich mit der Frage, ob der Austausch der Schlüssel effizienter gestaltet werden kann und präsentierten den Diffie-Hellman-Schlüsselaustausch. Mit ihm können zwei Kommunikationspartner, die über einen unsicheren Kanal jeweils eine Nachricht austauschen, einen gemeinsamen geheimen Schlüssel erzeugen. Jemand der beide Nachrichten mithört, ist jedoch nicht in der Lage diesen geheimen Schlüssel zu berechnen, was als das Diffie-Hellman-Problem bezeichnet wird. Der Diffie-Hellman-Schlüsselaustausch ist allerdings nicht sicher vor Man-in-the-Middle-Angriffen, in denen sich ein Angreifer zwischen die beiden Kommunikationspartner schaltet und Nachrichten verändern kann.

Das erwähnte Diffie-Hellman-Problem ist das Folgende: Es sei eine Primzahl p einer zyklischen Gruppe primer Ordnung $\mathbb{Z}_p^*$, der Erzeuger g dieser Gruppe und die Elemente $A = g^a \bmod p$ und $B = g^b \bmod p$ gegeben, wobei a und b eine jeweils geheimzuhaltende Zufallszahl aus der Menge $\{1, ..., p-1\}$ ist. Es gilt $g^{ab} \bmod p$ zu finden, wenn a und b unbekannt sind [38]. Die Werte g und p sowie A und B müssen nicht geheim gehalten werden und können unsicher übertragen werden. Der Schlüssel K berechnet sich nach Austausch der Werte bei den Kommunikationspartnern aus $K = B^a \bmod p$ beziehungsweise aus $K = A^b \bmod p$.

Ein sehr verbreitetes asymmetrisches kryptographisches Verfahren ist RSA, benannt nach den Nachnamen seiner Entwickler Ron Rivest, Adi Shamir und Leonard Adleman. Das Verfahren wurde 1977 veröffentlicht. Es kann sowohl zur Verschlüsselung als auch zur digitalen Signatur verwendet werden. RSA basiert auf einer sogenannten Falltürfunktion, das ist eine spezielle Einwegfunktion, die mit Hilfe einer Zusatzinformation auch rückwärts leicht zu berechnen ist. Die Zerlegung einer großen Zahl in ihre Primfaktoren ist eine Einwegfunktion, da dies ein sehr aufwändiges Verfahren ist, während die Multiplikation von Primzahlen einfach ist. Mit der in RSA öffentlichen Information N und e, wobei $N = p \cdot q$, mit $p, q \in \mathbb{P}$ und $1 < e < \varphi(e)$, ist es leicht $m^e \bmod N$ von einer zu verschlüsselnden Nachricht m zu berechnen, während es in die andere Richtung nicht möglich ist. Ist die Faktorisierung von N allerdings bekannt, ist die inverse Berechnung ebenfalls einfach durchzuführen [13].

Elliptic Curve Cryptography (ECC) sind asymmetrische Kryptosysteme, die auf Operationen auf elliptischen Kurven über endlichen Körpern basieren. Der Vorteil gegenüber herkömmlichen asymmetrischen Verfahren ist, dass im Gegensatz zum Problem der Primfaktorzerlegung von ganzen Zahlen in RSA für das Problem des diskreten Logarithmus in elliptischen Kurven (ECDLP) kein Algorithmus mit einer Laufzeit von sub-exponentieller Zeit bekannt ist [37]. Somit kommt ECC bei vergleichbarer Sicherheit mit erheblich kürzeren Schlüsseln aus, als das bei herkömmlichen asymmetrischen Kryptosystemen der Fall ist. Dadurch eignen sich Verfahren, die auf ECC basieren, besonders für eingebettete Systeme, bei denen Speicher- und Rechenkapazitäten begrenzt sind.

3.1.1 Advanced Encryption Standard (AES)

Der symmetrische Verschlüsselungsstandard Advanced Encryption Standard (AES) [39] ist eine Blockverschlüsselung und der Nachfolger des Data Encryption Standards (DES). AES basiert auf dem Rijndael-Algorithmus [8] und wurde im Oktober 2000 vom National Institute of Standards and Technology (NIST) als neuer Standard veröffentlicht. DES mit seiner Schlüssellänge von nur 56 Bits wurde als zu schwach eingestuft. Im Jahr 1999 wurde ein DES-Schlüssel mit dem DES-Cracker der Electronic Frontier Foundation (EFF) [14] in 22 Stunden und 15 Minuten gefunden. NIST hatte bei der ersten AES-Konferenz fünfzehn Verschlüsselungsalgorithmen als Kandidaten zum DES-Nachfolger zur Auswahl, von denen nach der zweiten Konferenz noch fünf übrig blieben. Das wichtigste Auswahlkriterium für den neuen Standard war die allgemeine Sicherheit des Algorithmus [7].

Das Ziel eines symmetrischen Verschlüsselungsalgorithmus, wie AES, ist es, dass die sogenannte Exhaustionsmethode, das Ausprobieren aller möglichen Schlüssel, der beste Angriff gegen den Algorithmus darstellt. Es gibt jedoch keine Möglichkeit zu beweisen, dass dies der beste Angriff auf einen Verschlüsselungsalgorithmus ist [7]. Unter der Annahme, dass die Exhaustionsmethode den besten Angriff darstellt, bestimmt die Länge des Schlüssels die Stärke des symmetrischen Algorithmus. Um einen n-Bits langen Schlüssel zu finden, müssen durchschnittlich 2^{n-1} Möglichkeiten durchsucht werden. In der Kryptographie wird versucht, Angriffe auf Verschlüsselungsalgorithmen zu finden, die weniger als die 2^{n-1} Operationen durchführen müssen. Wurde so ein Angriff gefunden, dann gilt der Algorithmus als gebrochen, auch wenn die Realisierung dieses Angriffs praktisch irrelevant ist.

Beinahe alle symmetrischen Verschlüsselungsverfahren verwenden mehrere Runden zur Durchführung der Verschlüsselung. Um einen Angriff auf das Verfahren zu finden, wird mit einer vereinfachten Version des Algorithmus mit weniger Runden gearbeitet. Anschließend wird versucht, den gefundenen Angriff auf den vollen Algorithmus auszuweiten. Die Entwickler der Verschlüsselungsverfahren schätzen die Anzahl der Runden ab, die benötigt werden, um bekannten Angriffen zu widerstehen, und fügen als Sicherheitsbuffer weitere Runden hinzu.

NIST war nicht in der Lage, eine adäquate Auswahlentscheidung basierend auf der Sicherheit der Verschlüsselungsalgorithmen zu treffen. Alle teilnehmenden Algorithmen waren so konstruiert, dass sie den bekannten Angriffsmethoden widerstehen konnten. Dennoch gibt es Unterschiede in der Gestaltung der Algorithmen betreffend deren Sicherheit. Der Teilnehmer Serpent war der Gegenpol zum späteren AES-Standard, dem Rijndael-Algorithmus. Serpent war auf maximale Sicherheit ausgelegt und benötigte 32 Runden zur Verschlüsselung. Dadurch hat Serpent einen großen Sicherheitsbuffer, denn der beste Angriff auf den Algorithmus war ein Angriff auf eine Version mit nur 12 Runden. Der Preis dafür war jedoch die Geschwindigkeit der Ausführung in Software, so ist Serpent zirka ein Drittel langsamer als Rijndael [13].

Weitere wichtige Auswahlkriterien für den AES-Standard waren die Performance sowie die lizenzfreie Implementierung und Benutzung. Die Anforderungen an die Geschwindigkeit betrafen verschiedene Plattformen wie Pentium-Prozessoren, 8-Bit-Mikrocontroller und die direkte Implementierung in Hardware, zum Beispiel auf einem FPGA. Die Wahl des Rijndael-Algorithmus als AES-Standard geschah primär auf Grund seiner guten Performance auf den verschiedenen Plattformen sowie wegen seiner einfachen Realisierbarkeit in Hardware [7]. Der Algorithmus ist außerdem der Einzige, der je nach Länge des Schlüssels eine unterschiedliche

Anzahl an Runden benötigt, wodurch die Performance von AES mit einem 128 Bits langen Schlüssel erhöht wird. Jeder Schritt der Verschlüsselung besteht aus mehreren Operationen, die auch parallel ausgeführt werden können. Der Rijndael-Algorithmus war unter den zur Wahl stehenden Algorithmen bei der Ausführung sowohl in Hardware als auch in Software unter den schnellsten Algorithmen [49].

Rijndael ist ein Algorithmus zur symmetrischen Blockverschlüsselung. Bei symmetrischen Kryptosystemen verwenden beide Teilnehmer den gleichen Schlüssel zur Ver- und Entschlüsselung. Blockverschlüsselungen können, im Gegensatz zu Stromverschlüsselungen, zu jedem Zeitpunkt nur einen Block an Daten verschlüsseln. Zur Verschlüsselung längerer Daten wird ein Betriebsmodus verwendet, der festlegt, wie die Blockverschlüsselung anzuwenden ist. Der Rijndael-Algorithmus unterstützt verschiedene voneinander unabhängige Block- und Schlüssellängen. Für AES wurde vom NIST die Blocklänge allerdings auf 128 Bits festgesetzt und die Länge der Schlüssel auf 128, 196 und 256 Bits eingeschränkt. Die Bezeichnungen der drei AES-Varianten AES-128, AES-192 und AES-256 beziehen sich jeweils auf die gewählte Schlüssellänge.

Arbeitsweise

Der DES-Standard und viele andere symmetrische Verschlüsselungsverfahren basieren auf Feistelnetzwerken, einer symmetrischen Struktur, mit der Blockverschlüsselungen realisiert werden können. Der AES-Standard beruht auf dem als Substitutions-Permutations-Netzwerk bekannten Designprinzip, das eine gute *Diffusion* und *Konfusion* bietet. Konfusion ist ein Verschlüsselungskonzept, das darauf beruht, dass die Beziehung zwischen dem Schlüssel und dem Geheimtext so komplex wie möglich gestaltet wird. Diffusion ist ein Prinzip zur Auflösung von statistischen Strukturen des Klartextes im Zuge der Verschlüsselung. Ein Substitutions-Permutations-Netzwerk besteht aus einer Anzahl von Runden gleichen Aufbaus.

AES führt seine Vorgänge auf einer zweidimensionale Datenstruktur mit vier Zeilen aus, die *State* genannt wird. Jede Zelle dieses States ist ein Byte groß. Da AES eine auf 128 Bits festgelegte Blockgröße hat, weist die Datenstruktur vier Spalten auf. Jeder Block wird nun nacheinander bestimmten Transformationen unterzogen. Die Anzahl der Runden, die zur Ausführung des Algorithmus absolviert werden müssen, hängt von der Länge des Schlüssels ab. Für AES-128 werden zehn, für AES-192 zwölf und für AES-256 werden vierzehn Runden durchgeführt, wobei sich die jeweils letzte Runde geringfügig von den anderen unterscheidet.

Ablauf

Zu Beginn des Algorithmus wird die Eingabe in in den State s kopiert, was nach dem Schema $s[r, c] = in[r+4c]$ geschieht, wobei r für eine Zeile und c für eine Spalte der zweidimensionalen Datenstruktur stehen. Der State wird im Zuge der Ausführung der Runden des Algorithmus den Transformationen unterzogen. Anschließend wird der letzte State nach dem Schema $out[r + 4c] = s[r, c]$ in die Ausgabe out kopiert. Abbildung 3.1 zeigt die Eingabe, den State und die Ausgabe.

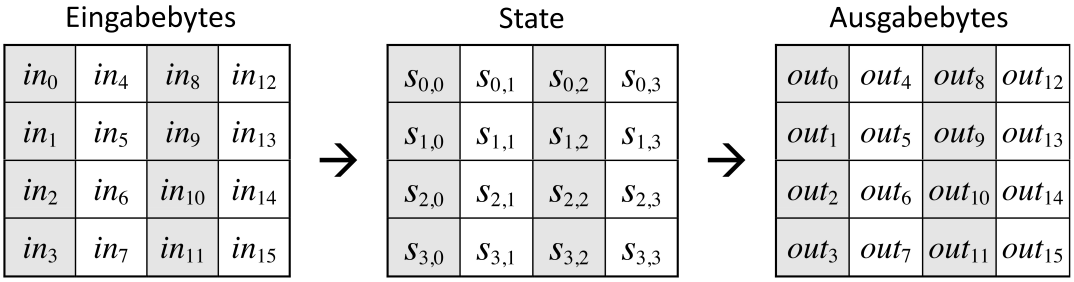

Abbildung 3.1: Eingabe und Ausgabe des States [39]

Der Ablauf des gesamten AES-Algorithmus sieht wie folgt aus:

1. Schlüsselexpansion: $R + 1$ Rundenschlüssel werden erzeugt, R ... Anzahl der Runden
2. Vorrunde
   a) `AddRoundKey(Rundenschlüssel[0])`
3. Verschlüsselungsrunden ($r = 1$ bis $R - 1$)
   a) `SubBytes()`
   b) `ShiftRows()`
   c) `MixColumns()`
   d) `AddRoundKey(Rundenschlüssel[r])`
4. Schlussrunde
   a) `SubBytes()`
   b) `ShiftRows()`
   c) `AddRoundKey(Rundenschlüssel[R])`

Abbildung 3.2 veranschaulicht den Ablauf des AES-Algorithmus und zeigt die Operationen, die in einer Runde des Algorithmus ausgeführt werden. Die einzelnen Operationen werden im Folgenden genauer erläutert.

Schlüsselexpansion In diesem Schritt werden aus dem Schlüssel die Teilschlüssel, genannt Rundenschlüssel, erzeugt, was durch eine Expansion des Schlüssels geschieht. Der Erzeugungsprozess der Rundenschlüssel ist nicht linear und nicht invertierbar. Da die Vorrunde auch als Runde gilt, müssen $R + 1$ Rundenschlüssel generiert werden.

`AddRoundKey()` In der Vorrunde und am Ende jeder Runde wird `AddRoundKey()` ausgeführt, worin der aktuelle Rundenschlüssel mit dem State kombiniert wird, was durch eine bitweise XOR-Verknüpfung erfolgt.

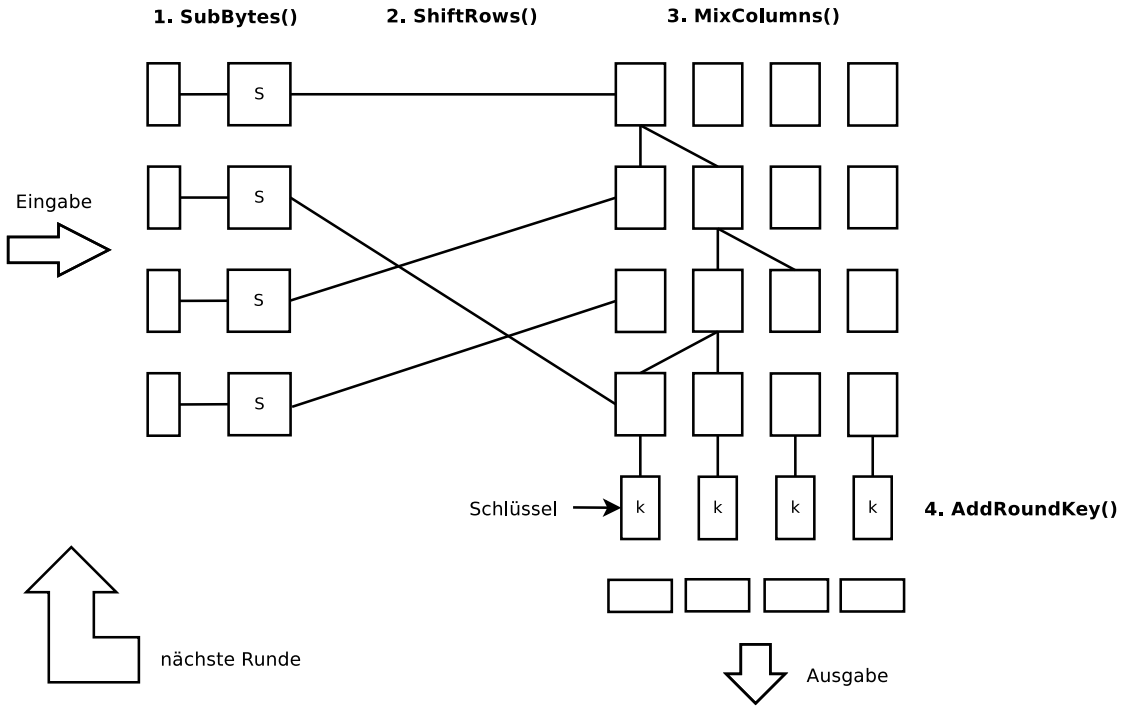

Abbildung 3.2: Struktur des AES-Algorithmus [46]

`SubBytes()` Diese Funktion ist der erste Schritt jeder Runde. Es wird darin eine nicht lineare Substitution ausgeführt. Jedes Byte des States wird ersetzt, indem eine Substitutionsbox (S-Box) benutzt wird (siehe Abbildung 3.3). Die S-Box ist eine zweidimensionale Tabelle mit 256 Bytes und wird mit Hilfe des multiplikativen Inversen des endlichen Körpers $GF(2^8)$ gebildet. Die Werte der S-Box können im Vorhinein berechnet und gespeichert werden oder sie werden dynamisch festgelegt, um Speicher zu sparen.

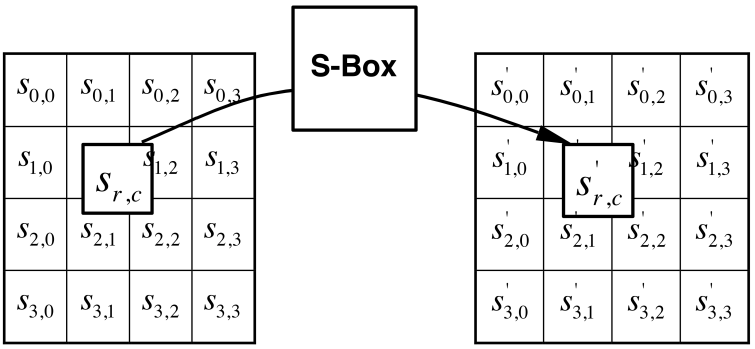

Abbildung 3.3: Anwendung der S-Box in `SubBytes()` [39]

`ShiftRows()` Die Bytes in den letzten drei Zeilen des States werden um eine bestimmte Anzahl an Spalten nach links verschoben. Die Anzahl der Verschiebungen ist abhängig von der Zeile.

`MixColumns()` Dieser Schritt sorgt, gemeinsam mit `ShiftRows()`, primär für die Diffusion im Verschlüsselungsalgorithmus, indem die Daten innerhalb der Spalten vermischt werden. Die Spalten werden als Polynome über $GF(2^8)$ betrachtet und es findet eine Matrixmultiplikation statt. Alle vier Bytes der Eingabe beeinflussen jedes Byte der Ausgabe.

Entschlüsselung Bei der Entschlüsselung werden die Operationen rückwärts durchlaufen. Die Daten werden wieder in eine zweidimensionale Datenstruktur kopiert und es werden die Rundenschlüssel erzeugt. Begonnen wird mit der Schlussrunde und die Funktionen werden in umgekehrter Reihenfolge durchlaufen. Die Funktionen zum Entschlüsseln unterscheiden sich auf Grund des Einsatzes von hauptsächlich simplen XOR-Operationen nur geringfügig von jenen zum Verschlüsseln. In `AddRoundKey()` werden sogar lediglich XOR-Verknüpfungen verwendet, wodurch diese Funktion selbst ihre eigene Inverse ist. Zum Entschlüsseln wird außerdem eine inverse S-Box eingesetzt und die Zeilenverschiebungen erfolgen in die entgegengesetzte Richtung.

Angriffe

AES ist ein Algorithmus mit einer eleganten und einfachen Struktur. Das kann einerseits als Vorteil gesehen werden, weil eine Analyse der Stärke des Algorithmus erleichtert wird. Andererseits ist die Schlichtheit von AES gleichzeitig auch ein Nachteil, da ein Brechen des Algorithmus mit neuartigen Angriffstechniken wahrscheinlicher ist [7].

Es existieren viele Angriffe auf den Algorithmus, doch alle diese bisherigen Angriffe auf AES sind von theoretischer Natur ohne praktischer Relevanz [13]. Die Designer von AES selbst haben einen Angriff auf eine vereinfachte Version des Algorithmus mit sechs Runden vorgestellt. Sie wussten also von einem Angriff auf AES, wenn der Algorithmus lediglich sechs Runden hätte. Aus diesem Grund wurde, abhängig von der Schlüssellänge, eine Anzahl von zehn bis vierzehn Runden gewählt. Die stärksten Angriffe, die während des Auswahlverfahrens des NIST auf AES gefunden wurden, waren ein Angriff auf AES-128 mit sieben (von zehn) Runden, ein Angriff auf AES-192 mit acht (von zwölf) Runden und ein Angriff auf AES-256 mit neun (von vierzehn) Runden. Der Angriff auf AES-128 benötigte 2^{101} Bytes an Speicher und 2^{120} Operationen [7].

Ein im Jahr 2010 vorgestellter Angriff auf AES-256 mit zehn (von vierzehn) Runden ist in 2^{45} Schritten erfolgreich und benötigt dazu 2^{33} Bytes an Speicher und 2^{44} Bytes an Daten [4]. Für diesen Angriff wurde die sogenannte *Related-Subkey*-Methode verwendet. Es kamen dabei zwei miteinander in Beziehung stehende Schlüssel zum Einsatz. Frühere Angriffe benötigten noch 64 Schlüssel. Dieses Angriffsmodell, mit Schlüssel die zueinander in Bezug stehen müssen, ist nicht allgemein als praktisches Modell akzeptiert und die Angriffe funktionieren bisher nur bei AES-256, der AES-Variante, die die größte Sicherheit bieten soll, und nicht bei AES-128.

Angriffe, die ebenfalls zueinander in Bezug stehende Schlüssel einsetzen, wurden auch auf den vollen Algorithmus von AES-192 und AES-256 präsentiert [5]. Diese Angriffe sind ebenso hauptsächlich von theoretischer Natur und stellen keine Gefahr für reale Anwendungen dar, die AES benutzen. Es handelt sich hierbei um Bumerang-Angriffe, die auf der differenziellen Kryptoanalyse basieren. Der Angriff auf den vollen AES-192 benötigt vier Schlüssel und 2^{176} Operationen. Der volle AES-256 kann mit vier Schlüssel in 2^{176} Schritten gebrochen werden.

Der erste Angriff auf den vollen AES-Algorithmus, ohne dabei mehrere Schlüssel einsetzen zu müssen, wurde im Jahr 2011 vorgestellt [6]. Es handelt sich dabei um einen *Biclique*-Angriff, eine neuartige Technik der Kryptoanalyse von Blockverschlüsselungen. Der Angriff berechnet den geheimen Schlüssel von AES-128 in $2^{126.1}$ Schritten, bei AES-192 werden $2^{189.7}$ Schritte benötigt und der Schlüssel von AES-256 kann in $2^{254.4}$ Schritten berechnet werden. Im Durchschnitt sind diese Angriffe zwar um den Faktor 4 schneller als das Durchsuchen aller möglichen Schlüssel, für die praktische Sicherheit sind sie dennoch nicht relevant.

Seitenkanalangriffe Bei einem Seitenkanalangriff wird die Implementierung eines Algorithmus in einem Gerät angegriffen. Es wird dabei nicht das kryptographische Verfahren selbst mittels der Exhaustionsmethode angegriffen oder theoretische Schwachstellen im Algorithmus ausgenutzt. Ein bei einer Implementierung eines Kryptosystems funktionierender Seitenkanalangriff muss nicht auf andere Implementierungen übertragbar sein. Das kryptographische Gerät wird bei seiner Ausführung beobachtet. Damit wird versucht, charakteristische Informationen zu finden, die auf den Schlüssel schließen lassen. Beispiele für Angriffe dieser Art sind Analysen der Laufzeit oder der elektromagnetischen Abstrahlung eines Gerätes.

Es wurde ein Seitenkanalangriff auf AES-128 vorgestellt, bei dem der Schlüssel beinahe in Echtzeit erhalten werden kann [24]. Dabei werden weder Informationen über den Klartext, wie seine Verteilung, noch der Geheimtext benötigt. Konkret funktioniert dieser Seitenkanalangriff gegen die AES-Implementierung in OpenSSL 0.9.8n auf einem Linux-System. Es muss dafür ein Prozess unter einem Linux-Benutzerkonto ausgeführt werden können. Weiters benötigt der Angreifer eine Testmaschine, die mit der zu angreifenden Maschine identisch ist, um Trainingsmuster für zwei künstliche neuronale Netzwerke zu generieren. In der Praxis ist die größte Einschränkung des Angriffs, dass er auf eine ganz spezielle Kombination von Hardware und Software zugeschnitten ist und bereits kleine Änderungen dieser Konfiguration umfangreiche Anpassungen des Angriffs erforderlich machen.

3.1.2 Betriebsmodi von Blockverschlüsselungen

Eine Blockverschlüsselung wie AES kann Klartext mit der Länge von genau einem Block verschlüsseln. Im Gegensatz dazu kann mit Stromverschlüsselungen jedes Klartextzeichen sofort verschlüsselt und übertragen werden. Die Kombination von Blockverschlüsselung und Betriebsmodus erlaubt es, Nachrichten zu ver- und entschlüsseln, die länger als die Blocklänge sind.

Es existieren eine Reihe an Betriebsmodi von Blockverschlüsselungen. Die klassischen Betriebsmodi sind ECB, CBC, CFB, OFB und CTR [10, 13, 38]. Diese entstanden in den 1970er Jahren bis Anfang der 1980er Jahre. CTR wurde vom NIST erst im Jahr 2001, nach der Standar-

disierung von AES, zu den Empfehlungen von Betriebsmodi für Blockverschlüsselungen [10] hinzugefügt.

Im Folgenden wird insbesondere auf die Betriebsmodi CBC und CTR eingegangen, da diese die Grundlage für den im KNX S-AL verwendeten Modus CCM bilden.

Initialisierungsvektor Bis auf ECB verwenden die Betriebsmodi zusätzlich zum Klartext einen Datenblock, der Initialisierungsvektor genannt wird. Der Initialisierungsvektor wird eingesetzt, um zu verhindern, dass gleiche Klartextblöcke bei der Verschlüsselung mit demselben Schlüssel zu den immer gleichen Blöcken an Geheimtext führen. Der Initialisierungsvektor muss unter Verwendung des gleichen Schlüssels immer einmalig sein, um nicht die Datenvertraulichkeit zu schwächen oder gänzlich zu zerstören. Üblicherweise muss der Initialisierungsvektor nicht geheim gehalten werden, wodurch auch ein zufälliger Wert verwendet werden kann. Da dem Empfänger der Nachricht der Initialisierungsvektor bekannt sein muss, wird ein zufälliger Wert typischerweise mit der Nachricht im unverschlüsselten Header mitgesendet.

ECB (Electronic Codebook)

ECB ist der einfachste der verschiedenen Betriebsmodi. Die Klartextblöcke werden unabhängig voneinander verschlüsselt, wodurch Muster im Klartext erhalten bleiben. Identische Klartextblöcke ergeben mit dem gleichen Schlüssel identische Geheimtextblöcke. Eine Umordnung der Geheimtextblöcke hat eine entsprechende Umordnung der Klartextblöcke zur Folge. Ein Fehler in einem Block beeinflusst bei ECB nur die Entschlüsselung dieses einen Blocks. Da sich bei ECB auf Grund der voneinander unabhängigen Verschlüsselung der Blöcke auch ein Fragment eines Geheimtextes zum ursprünglichen Klartext entschlüsseln lässt, ist dieser Modus besonders anfällig für Replay-Angriffe, in denen abgefangene Pakete erneut gesendet werden.

CFB (Cipher Feedback)

Gewisse Anwendungen erfordern, trotz einer Blocklänge von n Bits, die Verschlüsselung und sofortige Übertragung von r Bits großen Klartextblöcken, wobei $r < n$. Das wird mit dem Modus CFB ermöglicht.

Die Ausgabe der Blockverschlüsselung wird bei CFB mit dem Klartext bitweise XOR verknüpft, um daraus den Geheimtext zu bilden. Der so entstandene Block an Geheimtext ist die Eingabe für den nächsten zu verschlüsselnden Block. Die Eingabe der Blockverschlüsselung im ersten Schritt ist der Initialisierungsvektor, der nicht vorhersehbar sein soll.

Bitfehler in einem r Bits langen Geheimtextblock beeinflussen die Entschlüsselung dieses und der nächsten $\lceil n/r \rceil$ Geheimtextblöcke. Erst nachdem n Blöcke an Geheimtext verarbeitet wurden, befindet sich der fehlerhafte Block nicht mehr im Schieberegister.

OFB (Output Feedback)

Bisher wurde in der einen oder anderen Form der Klartext als Eingabe für die Blockverschlüsselung verwendet. Bei OFB hingegen dient die Nachricht selbst nie als Eingabe für den Verschlüsselungsalgorithmus. Stattdessen wird direkt die Ausgabe der Verschlüsselungsfunktion als

Rückführung zur Verschlüsselung des nächsten Blocks verwendet. Als Eingabe für den ersten Aufruf der Verschlüsselungsfunktion findet der Initialisierungsvektor Verwendung. Somit wird die Blockverschlüsselung benutzt, um einen Schlüsselstrom zu erzeugen. Dieser Schlüsselstrom wird mit dem Klartext XOR verknüpft und damit die Geheimtextblöcke erstellt.

Dadurch, dass die Nachricht selbst keine Eingabe der Verschlüsselung ist, wird die Blockverschlüsselung als synchrone Stromverschlüsselung genutzt, wodurch es bei der Verwendung von OFB zu keiner Fehlerfortpflanzung kommt. Der Schlüsselstrom ist unabhängig vom Klar- und Schlüsseltext und hängt damit nur vom Initialisierungsvektor ab. Der Initialisierungsvektor muss bei jeder Ausführung von OFB mit dem gleichen Schlüssel einmalig sein, andernfalls wird die Sicherheit des Systems beeinträchtigt.

CTR (Counter)

Der CTR-Modus ist eine Vereinfachung von OFB, in der anstatt der Rückführung ein einmaliger Counter, der mit dem Initialisierungsvektor gleichzusetzen ist, als Eingabe für die Blockverschlüsselung dient. Zur Verschlüsselung V wird der Counter mit einem Schlüssel S verschlüsselt. Der so erhaltene Zwischenschlüssel S_i wird mit dem Klartext K XOR verknüpft, wodurch der endgültige Geheimtext G entsteht. Der Modus CTR ist somit definiert durch:

$$S_i := V(S, \text{ Counter } i) \quad \text{für } i = 1, ..., s$$
$$G_i := K_i \oplus S_i$$

Falls im letzten Schritt nur ein Teilblock von r Bits verschlüsselt werden soll, dann werden nur die höchstwertigsten r Bits für die XOR-Verknüpfung benutzt und die restlichen Bits des Blocks verworfen. Das Schema der Verschlüsselung ist in Abbildung 3.4 zu sehen.

Der Counter wird bei der Entschlüsselung ebenfalls verschlüsselt, um wieder den gleichen Zwischenschlüssel zu erhalten. Dieser Zwischenschlüssel wird mit dem Geheimtext XOR verknüpft, um den Klartext zu erhalten. Abbildung 3.5 zeigt die Entschlüsselung, die analog zur Verschlüsselung erfolgt, lediglich Klar- und Geheimtext sind vertauscht.

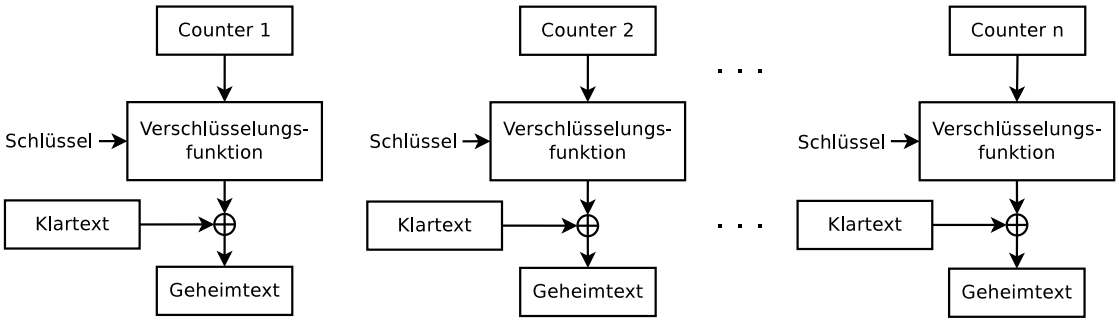

Abbildung 3.4: Verschlüsselung mit CTR [10]

Die Spezifikation des CTR-Modus verlangt, dass der Counter für jeden Klartextblock, der jemals mit dem gleichen Schlüssel verschlüsselt wird, einzigartig ist. Wird ein Counter-Block

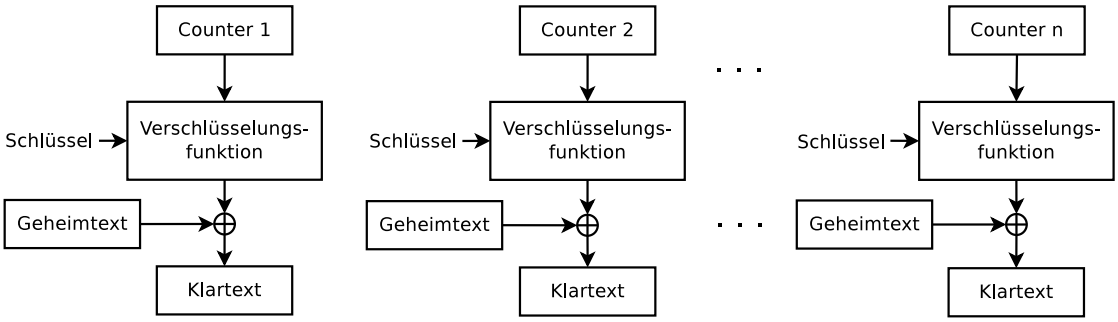

Abbildung 3.5: Entschlüsselung mit CTR [10]

mehrfach verwendet, kann die Datenvertraulichkeit gefährdet sein. Der Counter setzt sich üblicherweise aus einem einzigartigen Wert gefolgt von dem eigentlichen Counter-Wert zusammen. Wird der gleiche Counter unter dem gleichen Schlüssel für zwei verschiedene Nachrichten verwendet, dann werden diese mit dem gleichen Schlüsselstrom verschlüsselt. Angenommen zwei Klartext-Nachrichten K und K' werden mit dem gleichen Schlüsselstrom zu den Geheimtexten G und G' verschlüsselt. Ein Angreifer kann nun den Unterschied zwischen den beiden Klartexten berechnen:

$$\begin{aligned} G_i &:= K_i \oplus S_i \\ G'_i &:= K'_i \oplus S_i \\ G_i \oplus G'_i &= K_i \oplus K'_i \end{aligned}$$

Ist dem Angreifer einer der beiden Klartexte bereits bekannt, was in der Realität oft vorkommt, kann daraus leicht der andere Klartext berechnet werden [13].

Wie OFB ist auch CTR ein Modus, bei dem die Blockverschlüsselung als Stromverschlüsselung betrieben wird. Es kommt bei CTR zu keiner Fehlerfortpflanzung, Bitfehler in einem Block ergeben nur Fehler bei den entsprechenden Bits des Klartextes. Da es in CTR keine Rückführung gibt, können beliebige einzelne Blöcke entschlüsselt werden und sämtliche Ver- und Entschlüsselungsoperationen können parallel durchgeführt werden.

CBC (Cipher Block Chaining)

Die Probleme von ECB werden vermieden, indem bei der Verschlüsselung jeder Block des Klartextes mit dem vorhergehenden Geheimtextblock XOR verknüpft wird. Auf diese Art und Weise ist jeder Geheimtextblock von allen bisher verarbeiteten Klartextblöcken abhängig. Der erste Block des Klartextes wird mit dem Initialisierungsvektor verknüpft, wodurch identische Geheimtexte vermieden werden. Wird allerdings der gleiche Klartext mit dem gleichen Schlüssel und Initialisierungsvektor verschlüsselt, so ergibt das den gleichen Geheimtext. Die Struktur der Verschlüsselung wird in Abbildung 3.6 gezeigt; die Struktur der Entschlüsselung ist in Abbildung 3.7 dargestellt.

Im Gegensatz zur Verschlüsselung ist die Entschlüsselung nicht rekursiv und kann parallelisiert werden. Da im CBC-Modus die Geheimtextblöcke benutzt werden, um dem Klartext

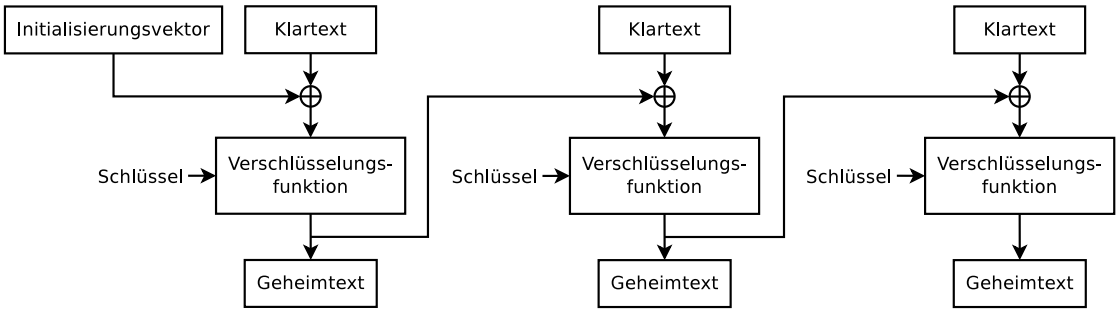

Abbildung 3.6: Verschlüsselung mit CBC [10]

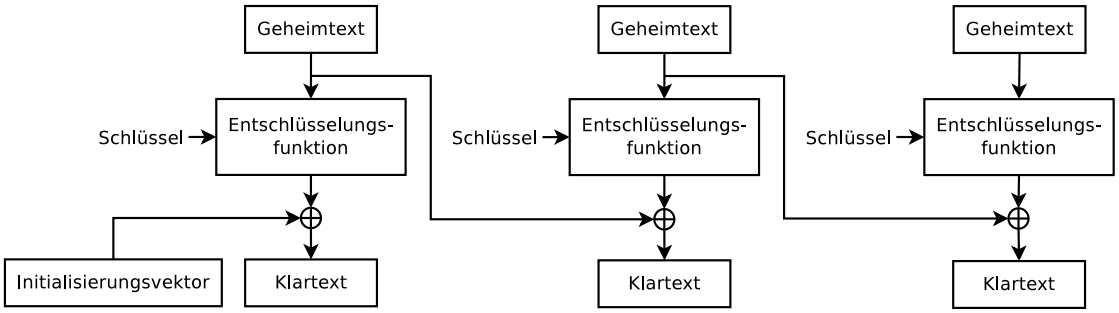

Abbildung 3.7: Entschlüsselung mit CBC [10]

innerhalb eines Blocks einen zufälligen Charakter zu verleihen, empfiehlt es sich, den Initialisierungsvektor zufällig zu wählen und dem Empfänger mitzusenden.

Fazit

ECB ist der einfachste der Betriebsmodi, was allerdings zur Folge hat, dass identische Klartextblöcke identische Geheimtextblöcke ergeben und so Muster im Klartext auch im Geheimtext erhalten bleiben. Durch diese Eigenschaft ist ECB für den Einsatz generell zu unsicher.

Die Betriebsmodi CFB, OFB und CTR haben den Vorteil, dass die Entschlüsselung analog zur Verschlüsselung abläuft und auch zur Entschlüsselung die Verschlüsselungsfunktion der Blockverschlüsselung verwendet wird. Somit muss die Funktion zur Entschlüsselung überhaupt nicht implementiert werden.

CBC ist einer der am meisten genutzten Betriebsmodi für Blockverschlüsselungen. Dieser Modus, mit dem empfohlenen zufälligen Initialisierungsvektor, hat gegenüber CTR die Nachteile, dass der Geheimtext länger ist und ein Zufallszahlengenerator benötigt wird. Der Vorteil von CBC ist seine große Robustheit. Der Modus funktioniert auch noch gut, wenn der Algorithmus missbraucht wird.

OFB und CTR sind Betriebsmodi, die Blockverschlüsselungen als Stromverschlüsselung betreiben. Der Initialisierungsvektor muss hier unbedingt für jeden zu verschlüsselnden Klartextblock unter dem gleichen Schlüssel einmalig sein. Ansonsten werden die Nachrichten mit

dem gleichen Schlüsselstrom verschlüsselt, wodurch Vertraulichkeit nicht mehr garantiert werden kann. Die Bedingung des einmaligen Initialisierungsvektors muss auch erfüllt sein, wenn das System angegriffen wird. Dies zu gewährleisten ist die Hauptquelle für Probleme und die Schwächung der Sicherheit. Insgesamt wiegen die Vorteile einer Stromverschlüsselung deren Nachteile jedoch oft auf [13].

3.1.3 CBC-MAC

CBC ist die Grundlage für CBC-MAC, wo unter Verwendung des Modus ein MAC zur Sicherung der Datenintegrität und zur Authentifizierung generiert wird. Die Daten werden mit einer Blockverschlüsselung im CBC-Modus verschlüsselt, wobei eine Kette an Blöcken gebildet wird. Jeder Block ist vom vorhergehend verschlüsselten Block abhängig und der letzte Block wird als MAC an die zu sendende Nachricht angehängt. Im ersten Schritt des Prozesses dient ein Initialisierungsvektor mit 0 als Eingabe. Die Erstellung des MAC wird in Abbildung 3.8 veranschaulicht.

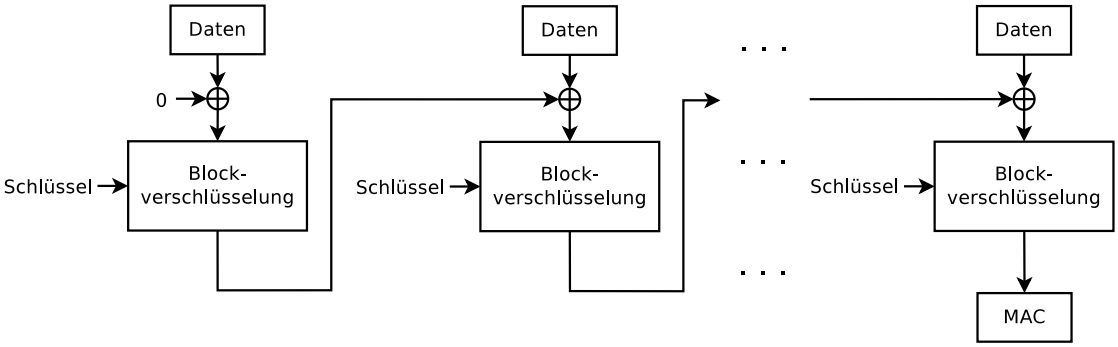

Abbildung 3.8: Struktur von CBC-MAC [38]

Falls eine mit CBC verschlüsselte Nachricht auch mit einem MAC gesichert werden soll, soll für die Generierung dieses CBC-MAC nicht der gleiche Schlüssel wie für die Verschlüsselung der Nachricht verwendet werden. Ansonsten ist der MAC äquivalent zum letzten Block des Geheimtextes, wodurch die gesamte Nachricht, ausgenommen des letzten Blocks, unentdeckt verändert werden kann.

3.1.4 CCM (Counter mit CBC-MAC)

Die klassischen Betriebsmodi von Blockverschlüsselungen gewährleisten Datenvertraulichkeit, sodass die Daten nur von autorisierten Teilnehmern gelesen werden können. Die Verschlüsselung der Daten alleine kann nicht deren Manipulation verhindern, denn die Entschlüsselungsfunktion sorgt lediglich dafür, dass der Klartext wiederhergestellt wird. Wurde der zu entschlüsselnde Geheimtext verändert, so wird dieser modifizierte Geheimtext entschlüsselt, auch wenn der daraus resultierende Klartext keinen Sinn ergibt. In fast allen Fällen ist der Schaden, der von modifizierten Nachrichten ausgeht, größer als der Schaden, wenn Klartext bekannt wird [13]. Deshalb sollte Verschlüsselung immer mit Authentifizierung und Integrität kombiniert werden.

Dazu wird zusätzlich ein Message Authentication Code (MAC) oder eine digitale Signatur benötigt. Mit CBC-MAC kann zwar auch unter Verwendung eines der klassischen Betriebsmodi für Datenintegrität gesorgt werden, für Datenvertraulichkeit gemeinsam mit Datenintegrität werden allerdings zwei verschiedene Schlüssel benötigt.

Um sowohl Vertraulichkeit als auch Authentifizierung und Integrität der Daten zu bieten, wurden MACs mit den bisherigen Betriebsmodi der Blockverschlüsselungen kombiniert [3]. Eine solche Betriebsart ist das im KNX S-AL verwendete CCM [11, 61]. Die beiden CCM-Spezifikationen [11,61] unterscheiden sich durch ihre Notation. In dieser Arbeit wird sich auf die Notation in [11] bezogen. Ursprünglich war Offset Codebook (OCB) der bekannteste Betriebsmodus für Blockverschlüsselungen, der Vertraulichkeit sowie Authentifizierung und Datenintegrität zu Verfügung stellt. OCB ist allerdings durch Patente geschützt, wodurch der kostenfreie Einsatz eingeschränkt ist. Das war der Auslöser für die Entwicklung von CCM durch Doug Whiting, Russ Housley und Niels Ferguson. CCM kombiniert den CTR-Modus für Vertraulichkeit mit CBC-MAC für Authentifizierung und Datenintegrität, wobei für CTR und CBC-MAC der gleiche Schlüssel verwendet wird. CCM ist für Blockverschlüsselungen mit einer Blocklänge von 128 Bits, wie AES, spezifiziert. Von der zugrunde liegenden Blockverschlüsselung wird nur die Verschlüsselungsfunktion und nicht die Entschlüsselung benötigt, wodurch die Codegröße klein gehalten wird. Es können mit CCM auch Nachrichten verarbeitet werden, in denen ein Teil der Nachricht lediglich authentifiziert aber nicht verschlüsselt werden soll.

Die Eingaben in CCM umfassen die Daten, die authentifiziert und verschlüsselt werden, Nutzdaten P genannt, optionale dazugehörige Daten A, die nur authentifiziert werden, sowie ein einmaliger Wert, die sogenannte Nonce N. Die drei Eingabewerte P, A und N werden in einer Sequenz an je 16 Bytes langen Datenblöcken B_i mit $i = 0, ..., r$ formatiert. Bei CCM können außerdem zwei Parameter bestimmt werden: (1) $Tlen$, die Bitlänge des MACs und (2) q, die Größe des Feldes Q, das die Bytelänge der Nutzdaten P angibt. Der Wert q gibt das Verhältnis von maximaler Nachrichtengröße und Größe der Nonce N an. Je größer q ist, desto kürzer ist die Nonce N.

Der erste Block B_0 besteht neben einem Byte mit Flags aus der Nonce N sowie der Bytelänge der Nutzdaten P, genant Q. Tabelle 3.1 zeigt den genauen Aufbau von B_0. Nach B_0 folgen optional die zu authentifizierenden Daten A verknüpft mit einer Zeichenkette, die die Länge von A angibt. Das Ergebnis wird in 16 Bytes lange Blöcke geteilt. Die Nutzdaten P werden ebenfalls in 16 Bytes lange Blöcke aufgeteilt. Daraus entsteht eine Sequenz an Blöcken B_0, B_1, ..., B_r, mit der Y_r, der CBC-MAC, berechnet wird. Abbildung 3.9 zeigt den Ablauf von CBC-MAC innerhalb von CCM. Zur weiteren Verarbeitung wird der Tag T aus den höchstwertigen $Tlen$ Bits von Y_r, der Ausgabe von CBC-MAC erstellt. Der Tag T stellt einen internen MAC-Wert des CCM-Algorithmus dar.

Byte Nummer	0	1 ... $(15 - q)$	$(16 - q)$ ... 15
Inhalt	Flags	N	Q

Tabelle 3.1: Formatierung von B_0 in CCM [11]

Zur Verschlüsselung der Nachricht wird der CTR-Modus verwendet, wozu Counter-Blöcke Ctr_j mit $j = 0, ..., m$ gebildet werden müssen. Diese bestehen aus einem Byte mit Flags,

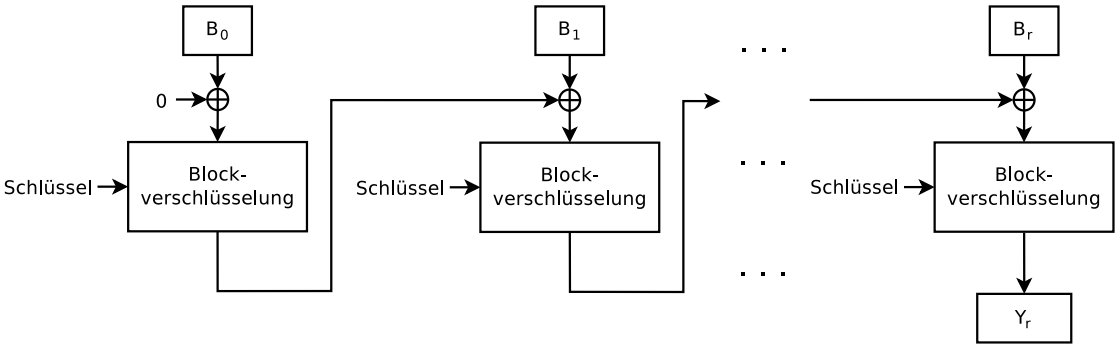

Abbildung 3.9: CBC-MAC in CCM [31, 38]

der Nonce N sowie einem Counter. Tabelle 3.2 zeigt den Aufbau der Blöcke Ctr_j. Der Block B_0 muss dabei unterschiedlich von allen Counter-Blöcken Ctr_j sein, für die CCM mit einem Schlüssel aufgerufen wird. Der MAC, der den endgültigen Wert zur Authentifizierung und Feststellung der Datenintegrität darstellt, entsteht aus der XOR-Verknüpfung von T mit den höchstwertigen $Tlen$ Bits von S_0, dem ersten Block des Schlüsselstroms S. Die verschlüsselte Nachricht wird durch XOR-Verknüpfung der Nutzdaten P mit der Bytelänge $Plen$ mit den höchstwertigen $Plen$ Bytes des Schlüsselstroms S erstellt. Der Geheimtext G ist die verschlüsselte Nachricht gefolgt vom MAC. Abbildung 3.10 veranschaulicht den Vorgang der Verschlüsselung im CTR-Modus innerhalb von CCM.

Byte Nummer	0	$1 \ldots (15-q)$	$(16-q) \ldots 15$
Inhalt	Flags	N	Counter j

Tabelle 3.2: Formatierung von Ctr_j in CCM [11]

Zur Entschlüsselung und Überprüfung des MAC werden der Schlüssel, die Nonce N, die zu authentifizierenden Daten A sowie der Geheimtext G benötigt. Wie bei der Verschlüsselung wird der Schlüsselstrom S erzeugt, um daraus durch XOR-Verknüpfung der höchstwertigen $(Glen - Tlen)$ Bits von S mit den höchstwertigen $(Glen - Tlen)$ Bits des Geheimtextes G die unverschlüsselten Nutzdaten P zu erzeugen. Der Wert T entsteht durch XOR-Verknüpfung des MAC (die niederwertigsten $Tlen$ Bits von G) mit den höchstwertigen $Tlen$ Bits von S_0. Abbildung 3.11 zeigt diesen Vorgang.

Aus den empfangenen Daten werden wiederum die Blöcke B_0, B_1, ... , B_r erzeugt. Mit der in Abbildung 3.9 gezeigten Generierung des CBC-MAC wird der Vergleichswert $T = MSB_{Tlen}(Y_r)$ generiert. Stimmen der empfangene und der generierte Wert von T überein, hat die Nachricht die Verifizierung bestanden. Diese Verifizierung innerhalb des Entschlüsselungsprozesses prüft die Korrektheit des MACs für die Nutzdaten P, die begleitenden Daten A sowie die Nonce N. Sind empfangener und berechneter MAC nicht identisch, darf bis auf die Tatsache, dass der MAC nicht korrekt ist, keine weitere Information vom Empfänger weitergegeben werden.

Um die Sicherheit aufrechtzuerhalten, dürfen für die Lebensdauer eines Schlüssels die Ver-

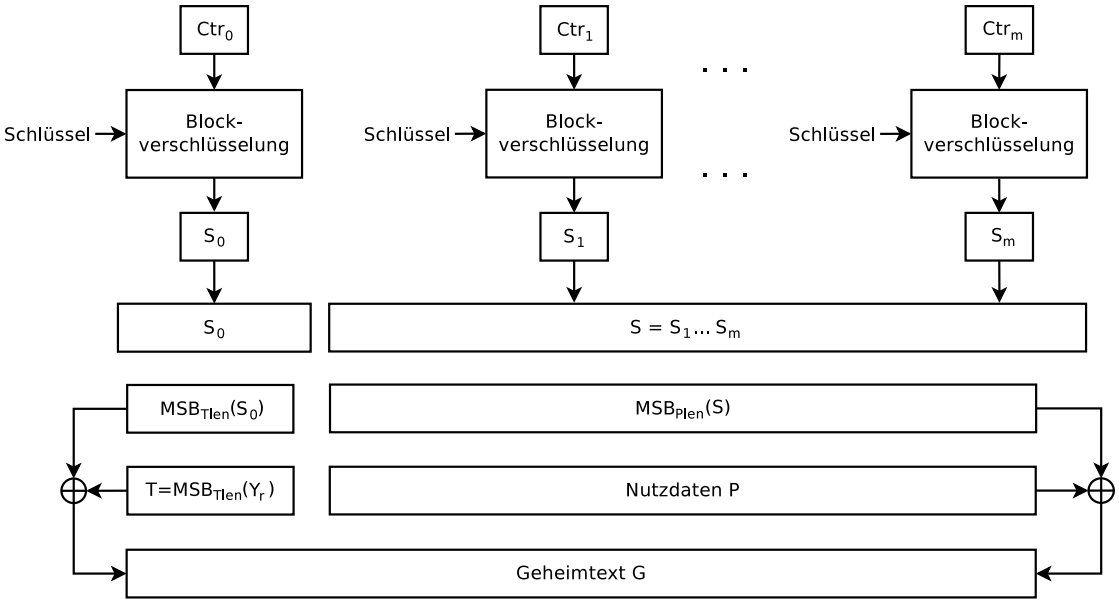

Abbildung 3.10: Verschlüsselung im CTR-Modus in CCM [31]

schlüsselungsoperationen für CBC-MAC und die der Daten zusammen höchstens 2^{61} mal aufgerufen werden. Das entspricht einer Datenmenge von beinahe 2^{64} Bytes beziehungsweise 16 Millionen Terabytes [61]. In Anwendungen, in denen dieses Limit erreicht werden könnte, muss im Sender dafür gesorgt werden, dass diese Höchstgrenze nicht überschritten wird.

CCM verwendet den CTR-Mode zur Verschlüsselung und ist somit im Grunde eine Stromverschlüsselung. Wie bei jeder Stromverschlüsselung gelten auch bei CCM die in Abschnitt 3.1.2 erläuterten Beschränkungen für den Initialisierungsvektor, um die Datenvertraulichkeit aufrechtzuerhalten. Die Wiederverwendung des gleichen Initialisierungsvektors in Kombination mit einem einzigen Schlüssel gefährdet die Datenvertraulichkeit.

Die Performance von CCM ist abhängig von der Geschwindigkeit der unterliegenden Blockverschlüsselung. CCM benötigt zwei Operationen der Blockverschlüsselung für Daten P, die zu verschlüsseln und zu authentifizieren sind. Für jeden Block an zusätzlich zu authentifizierenden Daten A wird eine Operation der Blockverschlüsselung benötigt. Der ungünstigste Fall liegt vor, wenn sowohl die Nachricht P als auch die zu authentifizierenden Daten A jeweils eine Länge von einem Byte haben. Hierbei müssen fünf Verschlüsselungsoperationen durchgeführt werden. Verglichen mit OCB benötigt CCM doppelt so viele Operationen, um eine Nachricht zu verschlüsseln und zu authentifizieren.

CCM ist von besonderer Bedeutung, da ein Beweis zur Sicherheit des Modus in Bezug auf die Sicherheit der zugrunde liegenden Blockverschlüsselung existiert [27]. Die Schlussfolgerung dieser Analyse ist, dass die Integrität und Authentifizierung in CCM auf einer Ebene mit der von anderen vorgeschlagenen Betriebsarten, wie OCB, liegen.

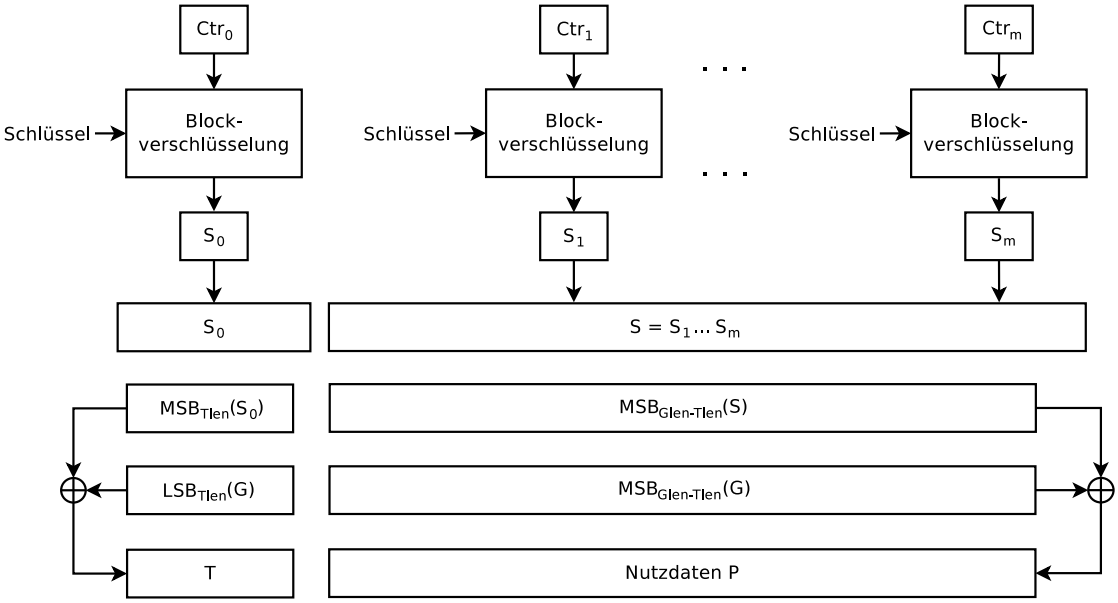

Abbildung 3.11: Entschlüsselung im CTR-Modus in CCM [31]

3.2 Security in der Gebäudeautomation

Es existiert eine Vielzahl an Technologien und Standards in der Gebäudeautomation. Die wichtigen offenen Standards, die mehrere Anwendungsbereiche abdecken, sind LonWorks, BACnet, ZigBee und KNX. In diesem Kapitel werden diese diskutiert und ihre Security-Konzepte analysiert, um sie abschließend gegenüber zu stellen. Auf diesem Weg werden die Eigenschaften sowie Vorzüge und Nachteile der implementierten Security-Mechanismen der Systeme hervorgehoben. Zusätzlich zu LonWorks, BACnet, ZigBee und KNX gibt es noch eine Reihe an Technologien, die für einen bestimmten Anwendungsbereich zugeschnitten sind und die im folgenden Absatz noch kurz vorgestellt werden. Abbildung 3.12 zeigt die Technologien und ihre Anwendungsbereiche in der Heim- und Gebäudeautomation anhand des 3-Schichtenmodells. Die folgende kurze Vorstellung beginnt von unten nach oben mit den Standards der Feldebene über die Automationsebene hin zu Technologien der Managementebene.

Ein bedeutender Vertreter in der Feldebene ist der europäische Standard Meter-Bus (M-Bus). M-Bus ist ein Kommunikationssystem zur Zählerdatenübertragung für alle Sparten sowie verschiedene Sensoren und Aktoren [43]. Es werden damit gemessene Werte übertragen, wie beispielsweise der Verbrauch von Strom in Stromzählern. M-Bus bietet allerdings keine Security-Mechanismen [17]. EnOcean und Z-Wave sind zwei drahtlose Technologien, die vorwiegend in der Feldebene eingesetzt werden. EnOcean benutzt *energy harvesting*, um die Geräte mit genügend Energie zu versorgen, um kurze Funksignale zu senden. Die Sender nutzen die Piezoelektrizität von Schaltern, Solarzellen, elektrothermische Wandler oder Bewegungsenergie mittels elektrodynamischer Energiewandler zur Energiegewinnung. In EnOcean steht somit nur eine geringe Menge an Energie zur Verfügung, es werden jedoch drei verschiedene Security-Modi

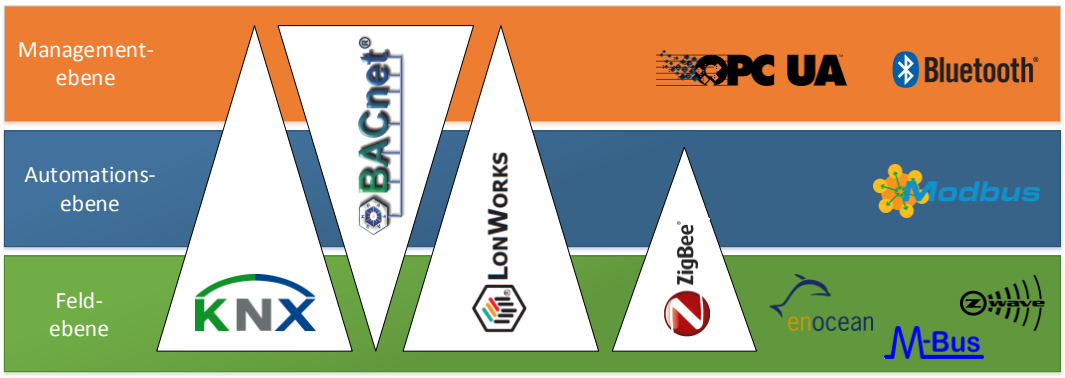

Abbildung 3.12: Standards der Heim- und Gebäudeautomation im 3-Schichtenmodell [17]

angeboten. Diese drei Modi weisen unterschiedlich starke Security-Mechanismen auf. Korrelierend mit dem Grad an Security wird auch unterschiedlich viel Energie benötigt [12]. Z-Wave wurde für die Heimautomation entwickelt und steht in direkter Konkurrenz zu ZigBee. Die Technologie hat Sicherheitsmechanismen vorzuweisen, der Standard ist jedoch nicht öffentlich verfügbar [17]. Modbus ist ein offenes Protokoll, das 1979 für die Kommunikation mit Speicherprogrammierbaren Steuerungen (SPS) veröffentlicht wurde und somit hauptsächlich in der Automationsebene Anwendung findet. In der Industrie hat sich Modbus zu einem de-facto Standard entwickelt, allerdings unterstützt das Protokoll keinerlei Security-Mechanismen. OPC Unified Architecture (OPC UA) ist eine plattformunabhängige, serviceorientierte Architektur, die für Interoperabilität in der Managementebene eingesetzt wird. In OPC UA ist Authentifizierung, Autorisierung, Verschlüsselung und Datenintegrität mittels Signaturen implementiert. Bluetooth ist ein Standard für die drahtlose Datenübertragung über kurze Distanz, der sich in der Informationstechnologie fest etabliert hat. In der Heim- und Gebäudeautomation kann Bluetooth ebenfalls eingesetzt werden, wobei die Hauptaufgaben hier im Bereich der Managementebene liegen und Bluetooth hauptsächlich zur Verbindung von mobilen Geräten wie Smartphones oder Tablets mit dem Gebäudeautomationssystem benutzt wird.

3.2.1 LonWorks

LonWorks wurde von der Echelon Corp entwickelt und besteht aus dem Kommunikationsprotokoll LonTalk sowie dem sogenannten Neuron-Chip. LonTalk ist von der IEC und der ISO als internationale Norm anerkannt. Der Neuron-Chip enthält den kompletten LonTalk-Protokoll-Stack und besteht aus drei CPUs, der Media-Access-, der Network- und der Applikations-CPU. Jeder Neuron-Chip besitzt eine eindeutige 48-Bit-Neuron-ID, um die Chips im Netzwerk zu identifizieren.

LonTalk unterstützt die Authentifizierung des Senders. Das geschieht in einem vierstufigen Challenge-Response-Verfahren, das in Abbildung 3.13 dargestellt ist. Es ist dabei nicht möglich, Kommunikationssitzungen aufzubauen und so müssen für jede sichere Anfrage vier Nachrichten

gesendet werden. Datenaktualität wird durch die 64 Bits lange Zufallszahl gewährt, indem der Sender der Anfrage die empfangene Zufallszahl in den Hashwert einbindet und somit sichergestellt ist, dass die Antwort nach der Anfrage gesendet wurde und auch tatsächlich vom Sender stammt.

Das Ablauf des Challenge-Response-Verfahrens sieht folgendermaßen aus:

1. Der Sender setzt das Authentifizierungs-Bit einer Nachricht (*Request*).
2. Der Empfänger generiert eine 64 Bits lange Zufallszahl und sendet diese zurück (die *Challenge* wird gestellt).
3. Der Sender berechnet einen 64 Bits langen Hashwert über die Nachricht und die erhaltene Zufallszahl mit einem geteilten Schlüssel. Dieser Hashwert wird an den Empfänger zurückgesendet.
4. Der Empfänger führt die gleiche Berechnung durch und vergleicht seinen berechneten Hashwert mit dem empfangenen Wert und sendet daraufhin die *Response*.

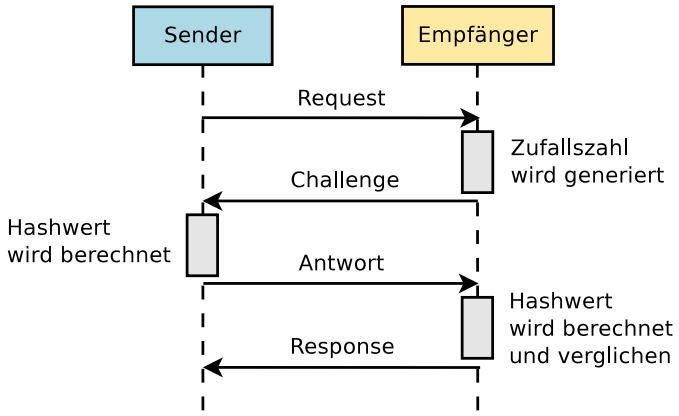

Abbildung 3.13: Authentifizierung in LonWorks [18]

LonWorks weist eine Reihe an Sicherheitslücken auf [18, 23, 50]. Details zum Zufallszahlengenerator und zum kryptographischen Algorithmus sind nicht öffentlich verfügbar und in die Firmware der Neuron-Chips integriert, wodurch sich keine Aussagen über die Stärken und Schwächen der Verfahren machen lassen. Bekannt ist die Länge von 48 Bits des Schlüssels zur Authentifizierung, was nicht mehr als sicher gilt. Der Austausch der Schlüssel muss über einen sicheren Kanal erfolgen, da LonTalk keine sichere Verteilung der Schlüssel unterstützt. Jeder Knoten besitzt nur einen Schlüssel zur Authentifizierung. Wenn eine Gruppe von Knoten miteinander kommunizieren will, teilt sich die ganze Gruppe einen Schlüssel. Aus diesem Grund ist keine Datenursprungsauthentifizierung möglich. Autorisierung wird nicht unterstützt, weil ein Gerät nur einen Schlüssel besitzen kann. Da die Datenübertragung unverschlüsselt erfolgt, kann keine Datenvertraulichkeit garantiert werden. Der Challenge-Response-Mechanismus kann nur

von einem Sender eingeleitet werden, ein Empfänger hat nicht die Möglichkeit sichere Anfragen zu fordern. Das Verfahren zur Authentifizierung ist außerdem anfällig für DoS-Angriffe, da für jede Nachricht mit gesetztem Authentifizierungs-Bit der Empfänger eine Zufallszahl generiert und einen Hashwert berechnet, was zeitaufwändig ist.

3.2.2 BACnet

Die Entwicklung von Building Automation and Control Networks (BACnet) begann im Jahr 1987 von der American Society of Heating, Refrigerating and Air-Conditioning Engineers (ASHRAE). BACnet unterstützt die Interoperabilität von Geräten verschiedener Hersteller. Seit 1995 ist BACnet ANSI/ASHRAE-Norm 135 und seit 2003 ISO-Norm 16484-5. Der Standard wird ständig erweitert, die Zusätze zum derzeit gültigen Standard werden in sogenannten Addenda festgehalten. Die aktuelle Version des BACnet-Standards ist ASHRAE 135-2012.

BACnet stellt eine Vielzahl an Security-Mechanismen zur Verfügung, die ursprünglich auf DES basierten. Dieses Security-Konzept enthielt ernsthafte Sicherheitslücken [25]. Wie in Abschnitt 3.1.1 beschrieben, ist DES kein sicherer Standard und das Protokoll zur Authentifizierung war anfällig für verschiedene Angriffe, darunter auch Replay-Attacken.

Auf Grund der gegebenen Sicherheitslücken wurde mit BACnet Addendum 135-2008g [2] ein vollständig erneuertes Security-Konzept eingeführt. Die Security-Dienste können mit allen BACnet-Übertragungsmedien und Gerätetypen verwendet werden und sind in den BACnet-Stack als Network-Layer-Nachrichten integriert, womit es keine echte Security-Schicht gibt.

Das Security-Konzept basiert auf gemeinsamen geheimen Schlüsseln, die von einem Schlüsselserver generiert und verteilt werden. Es gibt sechs Arten von Schlüssel:

- Der `General-Network-Access`-Schlüssel ist allen Teilnehmern eines Netzwerks bekannt und wird unter anderem für Broadcast-Nachrichten benutzt.
- Der `User-Authenticated`-Schlüssel wird an Geräte verteilt, die die Identität eines Benutzers authentifizieren können, oder an Geräte, die über keine Benutzerschnittstelle verfügen. Diese Geräte erfordern somit keine menschliche Interaktion und die Benutzeridentität der BACnet-Nachrichten wird im Gerät konfiguriert. Bei Nachrichten, die mit dem `User-Authenticated`-Schlüssel signiert wurden, kann davon ausgegangen werden, dass der Benutzer ordnungsgemäß authentifiziert wurde.
- Mit den `Application-Specific`-Schlüsseln kann eine Security-Abgrenzung zwischen verschiedenen Anwendungsgebieten, wie zum Beispiel Zugangskontrolle und HLK, gezogen werden. Die Verteilung von `Application-Specific`-Schlüssel ist auf Geräte begrenzt, die eine bestimmte Anwendung teilen.
- `Installation`-Schlüssel werden temporär zu Konfigurationszwecken verteilt.
- `Distribution`-Schlüssel werden verwendet, um die Schlüssel sicher zu verteilen.
- Die `Device-Master`-Schlüssel sind die am besten geschützten Schlüssel. Sie sind für jedes Gerät einzigartig und werden zur Verteilung der `Distribution`-Schlüssel eingesetzt.

Es gibt drei Arten von Nachrichten: Klartext-Nachrichten, signierte Nachrichten und verschlüsselte Nachrichten. Klartext-Nachrichten sind auf keine Art geschützt. Signierte Nachrichten bieten durch ihre Signatur Datenintegrität und durch eine Nachrichten-ID und einen Unix-Zeitstempel Datenaktualität, die sekundengenaue Zeitbestimmung ermöglicht. Da Zeitstempel benutzt werden, müssen die sicheren BACnet-Geräte ihre Uhrzeiten synchronisieren. Als Signatur kommt HMAC in Verbindung mit MD5 oder SHA-256 zum Einsatz, wobei die Länge der Signatur 128 Bits beträgt. Authentifizierung wird implizit durch die symmetrischen Algorithmen und die sogenannten Geräteinstanznummern der BACnet-Geräte gewährleistet. Bei verschlüsselten Nachrichten werden zusätzlich die Nutzdaten mit AES mit einer Schlüssellänge von 128 Bits im CBC-Modus verschlüsselt, um Datenvertraulichkeit zu erreichen.

Aus den verschiedenen Arten der Nachrichten resultieren die vier Stufen, die an Netzwerksicherheit definiert sind:

- `Plain-Non-Trusted`-Netzwerke haben keinerlei Anforderungen an die Sicherheit. Bei der Datenübertragung werden keine Security-Dienste des Protokolls angewendet und es existieren auch keine physischen Sicherheitsanforderungen.
- Bei `Plain-Trusted`-Netzwerken werden keine Sicherheitsdienste des Protokolls eingesetzt, sie sind jedoch physisch sicher.
- In `Signed-Trusted`-Netzwerken müssen alle Nachrichten zumindest signiert sein.
- In `Encrypted-Trusted`-Netzwerken sind nur verschlüsselte Nachrichten erlaubt, ansonsten werden die Nachrichten verworfen.

Es sind acht Dienste zur sicheren Kommunikation integriert. `Security-Payload` überträgt die Daten sicher zwischen den Kommunikationsteilnehmern. Eine Bestätigung einer Nachricht oder die Benachrichtigung über einen Fehler erfolgt mit `Security-Response`. Mit dem Dienst `Challenge-Request` kann die Identität eines Gerätes verifiziert werden. Die Anforderung von geheimen Schlüssel geschieht mittels `Request-Key-Update`, worauf ein Schlüsselserver mit einer `Security-Response`-Nachricht antwortet, wenn keine ausstehenden Schlüssel an das Gerät zu senden sind. Andernfalls werden mit dem `Update-Key-Set`-Dienst die Schlüssel übermittelt. Mit `Update-Distribution-Key` werden Distribution-Schlüssel an neue Geräte verteilt oder Schlüssel von bestehenden Geräten aktualisiert. Mit den Diensten `Request-Master-Key` und `Set-Master-Key` können die `Device-Master`-Schlüssel geändert werden.

BACnet bietet eine gute Basis an Sicherheitsdiensten, einige Aspekte sind allerdings unbeachtet oder offen geblieben [17]. Datenursprungsauthentifizierung kann nicht garantiert werden, wenn ein Schlüssel von mehr als zwei Geräten geteilt wird. Wenn Schlüssel, wie der `General-Network-Access`-Schlüssel, benutzt werden, die von mehreren Geräten geteilt werden, kann der Sender nicht mit Sicherheit identifiziert werden. Autorisierung muss in der Anwendung ausgeführt werden, weil die Verteilung der Schlüssel die Zugriffsrechte festlegt und diese Prozedur nicht genau im Standard definiert ist. Aus Performance-Gründen werden bei der Verteilung der Schlüssel nur symmetrische Kryptosysteme eingesetzt. Asymmetrische Kryptosysteme basierend auf Elliptic Curve Cryptography (ECC) können auch in eingebetteten

Geräten ausgeführt werden. Folglich würde kein vertrauenswürdiger Schlüsselserver mehr nötig sein. Außerdem erfordert BACnet als Kommunikationsmodell die Client/Server-Beziehung, wodurch nicht alle Arten von Anwendungen unterstützt werden.

3.2.3 ZigBee

ZigBee ist ein offener Standard für drahtlose Kommunikation, der für kostengünstige Geräte mit niedrigem Energieverbrauch ausgelegt ist. Die 2002 gegründete ZigBee-Allianz entwickelt den Standard, der auf IEEE 802.15.4 aufbaut. Die erste Spezifikation von ZigBee wurde im Jahr 2004 veröffentlicht. 2006 erschien eine neue Spezifikation, die jedoch nicht abwärtskompatibel ist. Die aktuelle Spezifikation ist ZigBee 2012.

IEEE 802.15.4 definiert die physikalische Schicht und die Medium-Access-Control-Schicht. ZigBee fügt die Netzwerk- und Anwendungsschicht sowie seine eigenen Security-Dienste hinzu. Die Security-Dienste sind in die Netzwerk- und die Anwendungsschicht integriert. Die Anwendungsschicht besteht aus einer Anwendungsunterschicht, dem ZigBee Device Object (ZDO) und den vom Hersteller definierten Anwendungsobjekten [62]. Die Anwendungsunterschicht ist die Schnittstelle zwischen der Netzwerkschicht und der Anwendungsschicht und ihre Dienste werden von dem ZDO und den Anwendungsobjekten genutzt. Das ZDO ist verantwortlich für die Initialisierung der Anwendungsunterschicht, der Netzwerkschicht und der Security-Dienste sowie für Verwaltungsaufgaben. Abbildung 3.14 zeigt den Aufbau des ZigBee-Protokoll-Stacks.

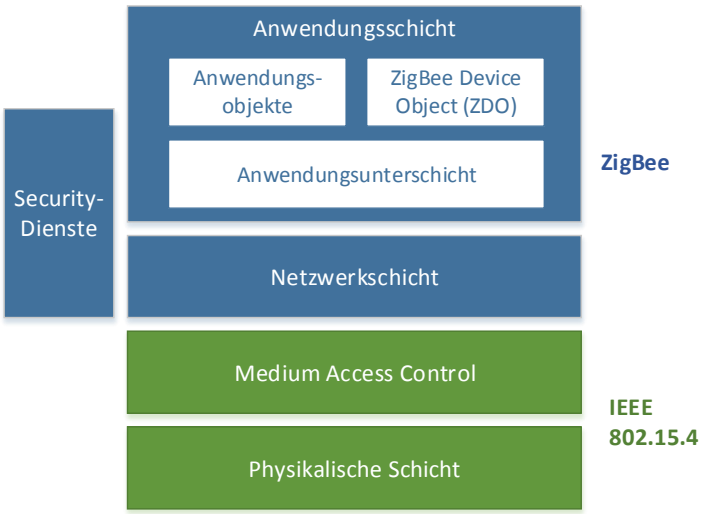

Abbildung 3.14: Architektur des ZigBee-Stacks [56]

Das Security-Konzept von ZigBee bietet Authentifizierung, Datenintegrität inklusive Datenursprungsauthentifizierung, Datenaktualität sowie die Verschlüsselung der Daten. Die zu übertragenden Nachrichten enthalten einen MAC von 32, 64 oder 128 Bits Länge. Tabelle 3.3 zeigt die verschiedenen Security-Levels von ZigBee. Die Daten werden mit AES mit CCM* mit einer

Schlüssellänge von 128 Bits verschlüsselt. CCM* stimmt mit der Spezifikation des originalen CCM überein, weist jedoch geringfügige Änderungen auf. Mit CCM* ist neben Authentifizierung sowie Authentifizierung mit Verschlüsselung auch nur Verschlüsselung der Daten möglich. Es wird dabei kein MAC an die Nachricht angefügt.

ID	Verschlüsselung	Integrität
0	Nein	Nein
1	Nein	Ja (32 Bits)
2	Nein	Ja (64 Bits)
3	Nein	Ja (128 Bits)
4	Ja	Nein
5	Ja	Ja (32 Bits)
6	Ja	Ja (64 Bits)
7	Ja	Ja (128 Bits)

Tabelle 3.3: Security-Levels in ZigBee [62]

Die Security-Architektur von ZigBee beinhaltet Security-Mechanismen in der Netzwerk- und in der Anwendungsschicht [62]. Diese Schichten sind für die sichere Übertragung ihrer jeweiligen Telegramme zuständig. Darüber ist die Anwendungsunterschicht für die Errichtung und Verwaltung der sicheren Kommunikationsbeziehungen verantwortlich. Das ZDO verwaltet die Security-Konfiguration eines Gerätes.

In jedem sicheren Netzwerk existiert ein *Trust Center*, das verschiedene für die Security relevante Aufgaben hat [36]. Es fungiert als *Trust Manager*, um Geräte zu authentifizieren, die sich zu dem Netzwerk verbinden wollen. Weiters ist das Trust Center als *Netzwerkmanager* verantwortlich für die Verwaltung und Verteilung der Schlüssel. Als *Konfigurationsmanager* sorgt es für die Security zwischen den Endgeräten.

In ZigBee gibt es drei verschiedene Schlüsselarten:

- Der *Link Key* wird von den Security-Diensten der Anwendungsschicht eingesetzt. Er sichert die Nachrichten zwischen zwei Geräten.
- Alle Geräte innerhalb eines Netzwerks benutzen den *Network Key*. Dieser Schlüssel wird von den Security-Diensten der Netzwerkschicht verwendet.
- Der *Master Key* wird nicht zur Verschlüsselung von Telegrammen eingesetzt, sondern wird als initialer Schlüssel zwischen zwei Geräten für die Prozedur der Einrichtung der Link Keys verwendet.

Es gibt verschiedene Möglichkeiten, die geheimen Schlüssel zu verbreiten [17]:

- Bei der `Pre-Installation` werden die Schlüssel in das Gerät geladen, bevor dieses sich zur Laufzeit zu einem Netzwerk verbindet. Es ist nicht exakt spezifiziert, wie genau

die `Pre-Installation` ablaufen muss. Der Vorgang kann zum Beispiel vom Hersteller der Komponente getätigt werden.

- Wenn die Methode `Key-transport` verwendet wird, sendet das Trust Center die Schlüssel mittels eines dedizierten Kommunikationsdienstes direkt an die Geräte. `Key-transport` wird eingesetzt, um den Network Key an die Geräte zu verteilen, die sich zum Netzwerk verbinden wollen, und um die Link Keys zu übermitteln.

- `Key-establishment` ist nur für Link Keys verfügbar. Im Gegensatz zu `Key-transport` sind bei `Key-establishment` beide Geräte am Erstellen der Schlüssel beteiligt, wobei der Master Key eingesetzt wird.

ZigBee bietet umfangreiche Sicherheitsdienste, es besteht jedoch noch Raum für Verbesserungen [17]. Das Trust Center stellt einen *Single Point of Failure* dar und in großen Netzwerken sind unter Umständen mehrere Hops nötig, um das Trust Center zu erreichen. Aus diesen Gründen wäre eine mehrfache Ausführung des Trust Centers wünschenswert. ZigBee verwendet ein eigenes Security-Konzept und nutzt nicht die Dienste von IEEE 802.15.4, wodurch Angriffe ermöglicht werden, die auf die Dienste der Data-Link-Ebene zielen. Da Link Keys nur zwischen zwei Geräten geteilt werden können, kann Gruppenkommunikation nur mit dem Network Key geschützt werden. Dadurch ist jedoch keine sichere Trennung der verschiedenen Multicast-Gruppen möglich und Datenursprungsauthentifizierung kann nicht garantiert werden. Autorisierung kann nur pro Gerät erfolgen und nicht pro Anwendung auf einem Gerät, weil ein Gerät nur einen Schlüssel besitzen kann. ZigBee ist anfällig für DoS-Angriffe, besonders zu dem Zeitpunkt, wenn ein Gerät dem Netzwerk beitritt und die Adresse zugewiesen wird sowie die Synchronisation erfolgt. Die Authentifizierung der Entitäten findest erst nach diesen Vorgängen statt und somit ist dieser Zeitraum nicht abgesichert.

3.2.4 Zusammenfassung

BACnet und ZigBee weisen ein gutes Sicherheitskonzept auf, während die Security-Dienste von LonWorks nur unzureichend sind. LonWorks unterstützt keine Verschlüsselung der Daten und sowohl Authentifizierung als auch Datenintegrität erfolgen anhand eines MACs der mit einer Schlüssellänge von nur 48 Bits generiert wird, was nicht mehr zeitgemäß ist. Für sicherheitskritische Anwendungen ist LonWorks somit nicht geeignet. Sowohl BACnet als auch ZigBee verwenden AES-128 zur Verschlüsselung der Daten und auch eine zuverlässige Datenintegrität ist bei beiden gegeben. Um Datenaktualität garantieren zu können, enthalten BACnet-Nachrichten nicht nur einen Zähler, sondern zusätzlich noch einen Zeitstempel, der es ermöglicht, Nachrichten nur innerhalb eines gewissen Zeitfensters anzunehmen. Daraus folgend müssen die Uhren der sicheren BACnet-Geräte synchronisiert werden. In LonWorks wird die Aktualität der Daten durch die 64 Bits lange Zufallszahl gewährt, die bei jeder sicheren Anfrage generiert wird und über die der Sender den Hashwert bilden muss, wodurch feststeht, dass die Antwort nach der Nachricht gesendet wurde. Bei der Autorisierung müssen sowohl bei BACnet als auch bei ZigBee Abstriche gemacht werden. BACnet legt die Autorisierung in die Hand der Anwendung; bei ZigBee ist sie nur für ein ganzes Gerät, nicht aber für eine einzelne Anwendung möglich. Ähnlich verhält es sich im Fall der Datenursprungsauthentifizierung, die von den Schlüsseln

abhängig ist und bei BACnet und ZigBee nur gegeben ist, wenn ein Schlüssel auf zwei Geräte beschränkt ist und dadurch eine eindeutige Identifizierung ermöglicht wird.

KNX bietet in seiner ursprünglichen Form keinerlei Sicherheitsmechanismen mit Ausnahme der Autorisierung für Managementfunktionen. Diese Autorisierung erfolgt anhand von 32 Bits langen Passwörtern, die im Klartext übertragen werden und somit nicht sicher sind. Die genaue Beschreibung der Security in KNX folgt im Abschnitt 4.6.

Tabelle 3.4 fasst die Sicherheitsmechanismen der wichtigen offenen Standards zusammen. Die Verfügbarkeit von Daten kann nicht durch kryptographische Methoden sichergestellt werden. Es sind auch physische Maßnahmen in Form von Komponenten zur Erkennung von ungewöhnlichem Datenverkehr sowie Gateways zur Trennung von Netzwerksegmenten nötig.

	LonWorks	**BACnet**	**ZigBee**	**KNX**
Authentifizierung und Datenintegrität	∼ 64 Bit MAC (48 Bit Schlüssel)	+ Geräte-ID und 128 Bit HMAC mit MD5 oder SHA-256	+ CCM* mit 32/64/128 Bit MAC	–
Autorisierung	–	∼ von der Anwendung ausgeführt	∼ pro Gerät, nicht pro Anwendung	– 32 Bit Klartext-Passwort
Datenvertraulichkeit	–	+ AES-128	+ AES-128	–
Datenursprungsauthentifizierung	–	∼ nur wenn ein Schlüssel auf zwei Geräte beschränkt ist	∼ nur wenn ein Schlüssel auf zwei Geräte beschränkt ist	–
Datenaktualität	∼ 64 Bit Zufallszahl	+ ID und Zeitstempel	+ Frame Counter	–
Datenverfügbarkeit	–	–	–	–

Tabelle 3.4: Übersicht der Sicherheitskonzepte von Gebäudeautomationssystemen [18]

KAPITEL 4

KNX

Die erste Spezifikation des Feldbussystems KNX wurde im Jahr 2002 veröffentlicht. KNX ist eine Weiterentwicklung des europäischen Installationsbusses (EIB) und entstand aus dem Zusammenschluss von EIB, BatiBus und European Home System (EHS). Es wurde die KNX Association gegründet, die für die KNX-Spezifikation [32] verantwortlich ist. Im Jahr 2003 wurde KNX als europäische Norm EN 50090 und 2006 als internationale Norm ISO/IEC 14543-3 anerkannt.

4.1 Konfiguration

Zur Konfiguration eines KNX-Netzwerks wird von der KNX Association die Engineering Tool Software (ETS) bereitgestellt. Die Software unterstützt den Anwender bei Projektplanung und Design, Inbetriebnahme, Projektdokumentation sowie bei der Diagnose und Störungsbeseitigung. Mit der ETS werden das Verhalten und die Kommunikationsbeziehungen der KNX-Geräte festgelegt. Die Hersteller der KNX-Geräte stellen die Produktdaten zur Verfügung, damit sie in die ETS importiert werden können. Mit Hilfe der Software werden auch die Adressen vergeben, die Parameter der Geräte festgelegt sowie die Programme in die Geräte geladen. Zur Diagnose kann die ETS alle am Bus übertragenen Telegramme anzeigen sowie für Testzwecke vom PC aus Telegramme senden. Die Konfiguration des KNX-Systems mit der ETS wird *System Mode (S-Mode)* genannt.

Eine andere Möglichkeit der Konfiguration ist der *Easy Mode (E-Mode)*. Die Konfiguration in diesem Modus ist einfacher und läuft nicht über einen PC ab, was sich allerdings auf die Flexibilität und Funktionalität niederschlägt. Die Konfiguration erfolgt dabei mit Hilfe einer der drei Sub-Modi:

- Im *Controller Mode* ordnet ein zentrales Steuerungsgerät jeder Komponente eine physikalische Adresse zu.
- Beim *Push Button Mode* werden die zu verbindenden Geräte manuell in einen speziellen Konfigurationsmodus gebracht.

- Der *Logical Tag Extended (LTE) Mode* unterscheidet sich von den beiden vorigen Modi dadurch, dass Verbindungen mit anderen Geräten vorgegeben sind. Mehrere Datenpunkte sind mit einer Zonenadresse verbunden und Datenpunkte innerhalb einer Zone können ihre Prozessdaten austauschen.

4.2 Topologie und Adressierung

In KNX wird eine dreistufige Hierarchie verwendet, wobei die kleinste Einheit eine Linie ist und bis zu 256 Teilnehmer enthalten kann. Bis zu 15 Linien können über Linienkoppler mit einer Hauptlinie verbunden werden, um einen Bereich zu bilden. Die Bereiche wiederum können über Bereichskoppler an die Bereichslinie angeschlossen werden. Der gesamte Adressbereich kann aus bis zu 15 verbundenen Bereichen bestehen und wird als Domäne bezeichnet. An die Bereichslinie können ebenfalls bis zu 255 Endgeräte angeschlossen sein, was nicht als eigenständiger Bereich angesehen wird. Abbildung 4.1 zeigt die beschriebene Topologie.

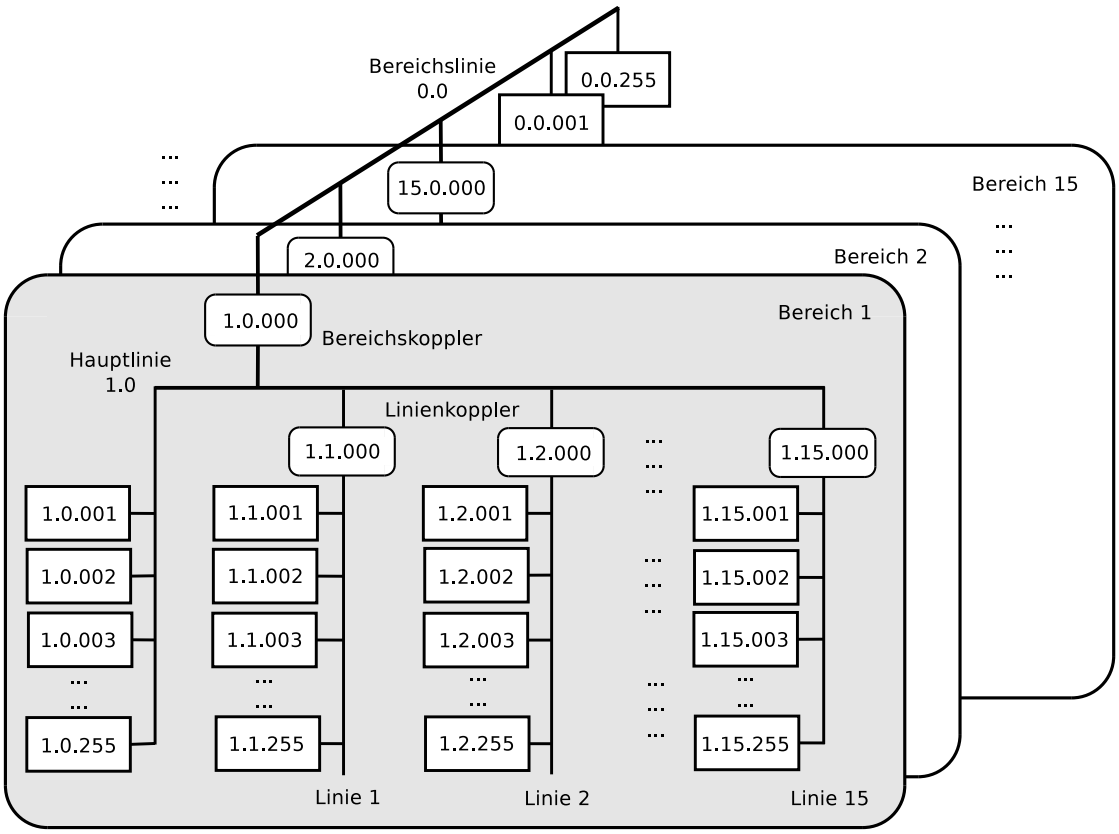

Abbildung 4.1: KNX-Topologie [32]

4.2.1 Adresstypen

In KNX gibt es zwei verschiedene Arten von Adressen, die physikalischen Adressen (auch individuelle Adressen genannt) und die Gruppenadressen. Die Kommunikation mit mehreren Teilnehmern erfolgt über die logische Adresse, die sogenannte Gruppenadresse.

Physikalische Adresse

Jedes Gerät im Netzwerk hat eine einzigartige physikalische Adresse. Die dreistufige Struktur der KNX-Topologie spiegelt sich in der physikalischen Adresse wider. Sie besteht aus zwei Bytes, wobei ein Byte die Geräteadresse enthält und das andere Byte in eine je vier Bits lange Bereichsadresse und Linienadresse aufgeteilt ist. Die Notation der Adressen erfolgt getrennt durch Punkte (*Bereichsadresse.Linienadresse.Geräteadresse*). Bereichs- und Linienadresse können auch zu einer Subnetzwerkadresse zusammengefasst werden. Die Geräteadresse ist innerhalb eines Subnetzwerks einmalig und Router haben immer die Geräteadresse null, wodurch Endgeräte die Werte 1 bis 255 zugewiesen bekommen können. Die Linienadresse ist innerhalb eines Bereiches einmalig und liegt zwischen 0 und 15. Die Bereichsadresse ist innerhalb eines Netzwerkes einmalig und liegt ebenfalls zwischen 0 und 15. Abbildung 4.2 zeigt den Aufbau der physikalischen Adresse.

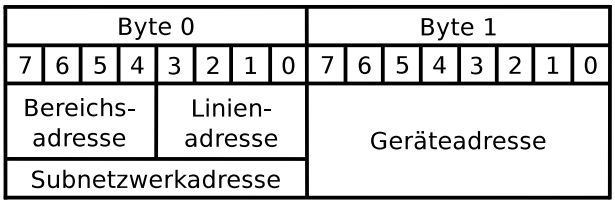

Abbildung 4.2: Struktur der physikalischen Adresse [32]

Gruppenadresse

Gruppenadressen müssen, im Gegensatz zu den physikalischen Adressen, nicht einzigartig sein. Einem Gerät können mehrere Gruppenadressen zugewiesen werden. Gruppenadressen werden global für das gesamte Netzwerk definiert. Wie die physikalischen Adressen weisen sie eine Länge von zwei Bytes auf. Zur Gruppenadresse 0x0000 ist jedes Gerät zugehörig, Nachrichten an diese Adresse entsprechen somit einem Broadcast.

In der ETS werden die Gruppenadressen, wie die physikalischen Adressen, als zwei- oder dreistufige Adressen dargestellt. Die dreistufigen Adressen sind aufgeteilt in eine Hauptgruppe mit Werten zwischen 0 und 15, eine Mittelgruppe, deren Wert zwischen 0 und 7 liegt, und die Untergruppe mit Werten von 0 bis 255. Die zweistufige Adresse besteht aus der Hauptgruppe mit einem Wert aus 0 bis 15 und der Untergruppe mit einem Wert von 0 bis 2047. Diese Adressen benutzen nur 15 der 16 zur Verfügung stehenden Bits. Zur Unterscheidung von den physikalischen Adressen sieht die Notation eine Trennung durch Schrägstriche anstatt durch Punkte vor (*Hauptgruppe/Mittelgruppe/Untergruppe* beziehungsweise *Hauptgruppe/Untergruppe*).

4.3 Schichten in KNX

Das OSI-Modell [26] standardisiert die Aufteilung eines Kommunikationssystems in Schichten. Es sind sieben Schichten definiert, wobei der Abstraktionsgrad der Funktionalität von oben nach unten hin abnimmt. KNX ist mit dem OSI-Modell übereinstimmend aufgebaut, es sind allerdings nicht alle der sieben OSI-Schichten in KNX implementiert. Der Presentation Layer, der für eine standardisierte Darstellung der Daten zuständig ist, und der Session Layer, der Dienste, wie den Aufbau von Sitzungen oder Synchronisation, anbietet, sind in KNX nicht enthalten. Das OSI-Modell und die Schichten in KNX sind in Abbildung 4.3 zu sehen.

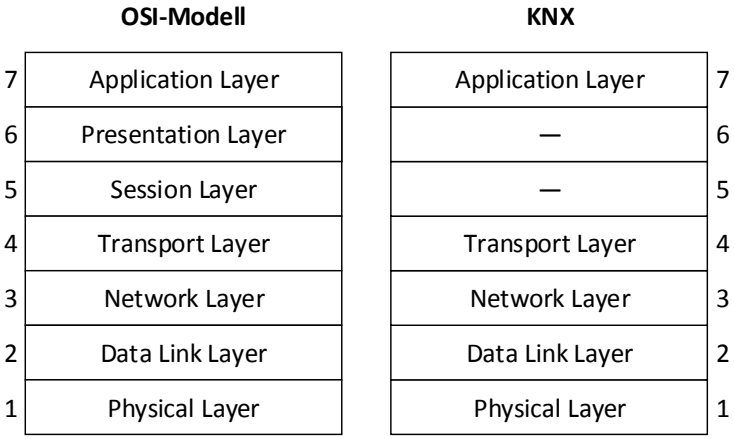

Abbildung 4.3: Schichten im OSI-Modell und in KNX

In KNX stehen mehrere physische Übertragungsmedien zur Verfügung, die auch miteinander kombiniert werden können. Das führt zu einer Aufteilung des Protokoll-Stacks in einen Teil, der vom Medium abhängig ist, bestehend aus dem Physical Layer und dem für das Medium spezifischen Teil des Data Link Layers, und einen vom Medium unabhängigen Teil, bestehend aus dem generellen Data Link Layer und der sogenannten Logical Link Control [30].

Eine Schicht kann die Dienste der darunter liegenden Schicht verwenden und stellt Dienste zur Verfügung, die vom Benutzer der jeweiligen Schicht verwendet werden. Üblicherweise ist der Benutzer einer Schicht die darüber liegende Schicht. Die Dienste werden bei KNX in die vier Dienstelemente *request (req)*, *confirm (con)*, *indication (ind)* und *response (res)* unterteilt. Es muss nicht jeder Dienst alle vier dieser Dienstelemente aufweisen.

Jede Schicht fügt den zu sendenden Daten ihre spezifischen Informationen hinzu. Somit wird eine Nachricht von jeder Schicht erweitert, um anschließend über das physische Medium übertragen zu werden. Die Kommunikation einer lokalen Schicht N mit der darunter liegenden Schicht $(N-1)$ erfolgt durch die jeweilige Service Data Unit (SDU), welche die Benutzerdaten der Schicht N darstellen. Die Schicht $(N-1)$ fügt ihre spezifischen Protokollinformationen hinzu, die Benutzerdaten inklusive dieser Informationen werden Protocol Data Unit (PDU) genannt. Die lokale Schicht N kommuniziert mit einer entfernten Schicht N über den Austausch

der PDU. Nachdem der Data Link Layer die restlichen Protokollinformationen hinzugefügt hat, wird das vollständige Telegramm, die Data Link Protocol Data Unit (LPDU), durch den Physical Layer übertragen. In Abbildung 4.4 wird die Kommunikation der Schichten in KNX dargestellt.

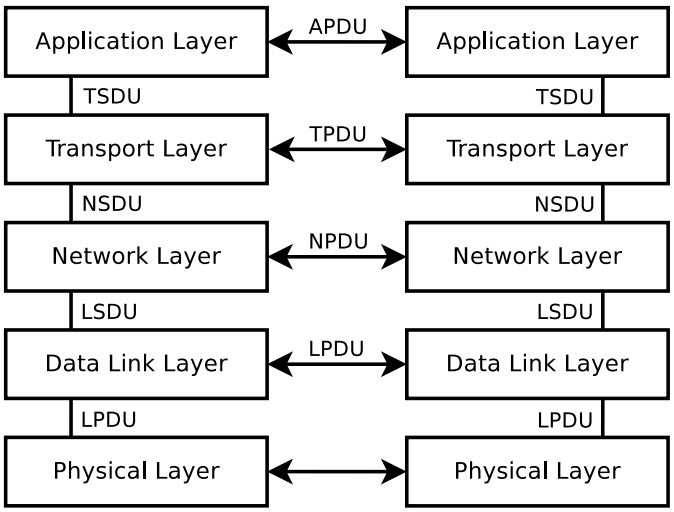

Abbildung 4.4: Interaktivität der Schichten in KNX

4.3.1 Physical Layer

Der Physical Layer ist die unterste Schicht des OSI-Modells und spezifiziert die elektrischen und physischen Eigenschaften der Datenverbindung. In KNX können verschiedene Medien zur Übertragung der Daten benutzt werden [30, 32].

- Twisted Pair 1 (TP1) wurde von EIB übernommen und ist bis dato das gebräuchlichste Übertragungsmedium. Die Teilnehmer werden über die Busleitung sowohl mit den Daten als auch mit der nötigen Betriebsspannung versorgt. Die Daten werden asynchron übermittelt und die Kommunikation erfolgt im Halbduplex-Verfahren. Die Geschwindigkeit der Datenübertragung beträgt 9600 Bits/s. Die Topologie kann frei gewählt werden. Zur Vermeidung von Kollisionen wird CSMA/CA eingesetzt.

- Bei Powerline 110 (PL110) wird die vorhandene 230-Volt-Leitung als Übertragungsmedium verwendet. Die Signale werden auf die Netzspannung aufmoduliert. Zur Datenübertragung wird eine Frequenzumtastung im Bandspreizverfahren eingesetzt. PL110 arbeitet mit einer Übertragungsgeschwindigkeit von 1200 Bits/s. Die Kommunikation erfolgt asynchron im Halbduplex-Verfahren. Die Teilnehmer greifen in Zeitschlitzen auf das Medium zu, um keine Kollisionen entstehen zu lassen (TDMA).

- KNX Radio Frequency (RF) ist die Übertragung über Funk im Frequenzband für Short Range Devices (SRD-Band), das in Europa im Bereich von 863 bis 870 MHz angesiedelt ist. Um den Übertragungsbereich zu erhöhen, können Repeater verwendet werden. Die

Datenrate von KNX RF beträgt 16.384 Bits/s. Zur Datenübertragung werden Frequency Shift Keying (FSK) und eine Manchester-Codierung eingesetzt. Um Übertragungsfehler zu erkennen und korrigieren zu können, enthalten KNX-RF-Nachrichten eine CRC-Prüfsumme mit einer Hamming-Distanz von 6 [48]. Damit sich die Adressräume nicht mit benachbarten Installationen überschneiden, wird ein erweitertes Adressformat benutzt. Je nach Kommunikationsbeziehung enthält dieses zusätzlich zur herkömmlichen Adressierung die KNX-Seriennummer oder die RF-Domänenadresse. Trotz des offenen Mediums stellt KNX RF keine Security-Mechanismen zur Verfügung.

- KNX IP erlaubt es KNX-Endgeräten, KNX-Telegramme direkt über ein IP-Netzwerk zu senden. Die KNX-Telegramme werden in den Nutzdaten von UDP-Telegrammen auf Port 3671 übertragen. Die Übertragungsgeschwindigkeit hängt vom unterliegenden IP-Netzwerk ab, ist in der Regel allerdings deutlich schneller als bei TP1 und PL110. KNXnet/IP ist schon länger als KNX IP verfügbar, jedoch nicht für Endgeräte gedacht, sondern für Geräte, die mit einer IP-Schnittstelle und einer Schnittstelle eines „traditionellen“ KNX-Übertragungsmediums ausgestattet sind (sogenannte KNXnet/IP-Server).

4.3.2 Data Link Layer

In KNX teilt sich der Data Link Layer in den für das Übertragungsmedium spezifischen Data Link Layer sowie in den vom Medium unabhängigen generellen Data Link Layer. Der generelle Data Link Layer befindet sich über dem jeweiligen spezifischen Data Link Layer. Der genaue Aufbau der Telegramme und die Zugriffssteuerung auf das Übertragungsmedium sind vom spezifischen Data Link Layer abhängig. Der generelle Data Link Layer besteht hauptsächlich aus der Schnittstelle zum Network Layer.

Der Data Link Layer hat die Aufgabe, eine zuverlässige Übertragung der Daten zu gewährleisten. Beim Senden ist der Data Link Layer dafür verantwortlich, die Daten in Blöcke aufzuteilen, den Zugang zum Übertragungsmedium zu schaffen und die Datenblöcke unter Verwendung des Physical Layers zu übertragen. Beim Empfang ist der Data Link Layer zuständig für die Überprüfung fehlerhafter Datenblöcke. Weiters wird anhand der Zieladresse entschieden, ob die Daten an die darüber liegende Schicht weitergegeben werden. Es wird mit einer positiven oder, im Falle einer fehlerhaften Übertagung, mit einer negativen Bestätigung an den Sender geantwortet. Ist ein Datenblock fehlerhaft, wird er verworfen und erneut übertragen, sodass die dem Data Link Layer übergeordnete Schicht nur korrekt empfangene Daten erhält.

Mit dem Dienst `L_Data` können die Datenblöcke übertragen werden. Es wird ein Telegramm an eine andere Instanz eines Data Link Layers im gleichen Subnetzwerk übertragen, wobei die Zieladresse eine physikalische Adresse oder eine Gruppenadresse sein kann. `L_Data.req` wird benutzt, um einen Datenblock zu senden. Mit `L_Data.ind` werden Daten empfangen. `L_Data.con` signalisiert, ob die Übertragung erfolgreich war. `L_SystemBroadcast` wird verwendet, um einen Datenblock an alle Geräte im Netzwerk zu senden. Mit `L_Poll_Data` können Daten von einem oder mehreren Teilnehmern angefordert werden.

4.3.3 Network Layer

Der Network Layer ist hauptsächlich für Knoten mit Routing-Funktionalität von Interesse. Für das Routing werden zwei verschiedene Mechanismen benutzt. Für Gruppenadressen wird ein Filteralgorithmus verwendet, um Telegramme von einem Subnetzwerk in ein anderes zu leiten. Bei Punkt-zu-Punkt-Adressierung wird die physikalische Adresse interpretiert. Es wird außerdem zwischen zwei verschiedenen Benutzern des Network Layers unterschieden. In einem Endgerät ist der Benutzer des Network Layers der darüber liegende Transport Layer; in einem Router ist der Benutzer des Network Layers der Filteralgorithmus.

Der Network Layer stellt seinem Benutzer den Unicast-Dienst `N_Data_Individual` und den Multicast-Dienst `N_Data_Group` zur Verfügung. Außerdem gibt es noch die Broadcast-Dienste `N_Data_Broadcast` und `N_Data_SystemBroadcast`, wobei ersterer Daten an alle Teilnehmer innerhalb eines Bereiches sendet und letzterer systemübergreifend Nachrichten versendet. Der Network Layer fügt dem Telegramm außerdem den `hop_count` hinzu, der von den Routern geprüft und im Zuge der Weiterleitung des Pakets erniedrigt wird.

4.3.4 Transport Layer

Der Transport Layer stellt fünf verschiedene Kommunikationsbeziehungen mit den jeweiligen Diensten zur Verfügung.

- Die Punkt-zu-Multipunkt-Verbindung (Multicast) ist verbindungslos und erlaubt die Kommunikation zwischen Geräten innerhalb einer Gruppe. Jedes Mitglied der Gruppe kann die Kommunikation beginnen und die Gruppe wird durch ihre Gruppenadresse identifiziert. Der Dienst dieses Kommunikationsmodus ist `T_Data_Group`.

- Die Kommunikationsbeziehung Punkt-zu-Domäne (Broadcast) ist verbindungslos und kann über den Dienst `T_Data_Broadcast` verwendet werden.

- Der Verbindungstyp Punkt-zu-allen-Punkten (System-Broadcast) ist verbindungslos und sein Dienst ist `T_Data_SystemBroadcast`.

- Die verbindungslose Punkt-zu-Punkt-Verbindung bietet den Dienst `T_Data_Individual`.

- Die verbindungsorientierte Punkt-zu-Punkt-Verbindung stellt die Dienste `T_Connect`, `T_Data_Connected` und `T_Disconnect` zur Verfügung. Die Benutzer dieses Kommunikationsmodus müssen vor dem Austausch der Daten zuerst eine Verbindung aufbauen. Im Fall der Überschreitung eines Zeitlimits oder bei Auftreten eines nicht behebbaren Fehlers kann die Verbindung vom Transport Layer beendet werden.

4.3.5 Application Layer

Der Application Layer ist die oberste Schicht und stellt den Anwendungen eine Reihe an Diensten zur Verfügung. Die Anwendungen der verschiedenen Geräte interagieren miteinander unter

Einsatz dieser angebotenen Dienste. Die Dienste des Application Layers können in zwei verschiedene Kategorien unterteilt werden [30]. Die erste Kategorie sind die Dienste, die eingesetzt werden, um Prozessdaten auszutauschen. Die wichtigsten Dienste zum Austausch von Prozessdaten sind diejenigen zur Gruppenkommunikation, nämlich `A_GroupValue_Read` und `A_GroupValue_Write`, die auf dem Transport-Layer-Dienst `T_Data_Group` basieren. In die zweite Kategorie fallen die Dienste für Konfigurations- und Wartungszwecke. Diese beinhalten Dienste, um Benutzeranwendungen zu laden, physikalische Adressen und Gruppenadressen zuzuweisen oder um Diagnoseinformationen abzurufen.

Abbildung 4.5 zeigt den Ablauf beim Aufruf von Diensten des Application Layers. Der Dienst des Application Layers wird dem entsprechenden Dienst des darunter liegenden Transport Layers zugeordnet. Der Transport Layer gibt dem Application Layer mit `T_Service.con` lokal eine Bestätigung, die vom Application Layer auf die lokale Bestätigung `A_Service.Lcon` abgebildet und an den Benutzer des Application Layers weitergegeben wird. Das Dienstelement `.req` wird nach der Übertragung über das Medium zum entfernten Transport Layer als `.ind` an den entfernten Benutzer weitergereicht. Die Antwort des entfernten Benutzers mittels `A_Service.res` wird wieder auf ein `T_Service.req` abgebildet. Die Bestätigung des entfernten Transport Layers wird vom Application Layer als `.Rcon` an den entfernten Benutzer des Application Layers übergeben. Die vom entfernten Benutzer übertragene Bestätigung wird an den lokalen Benutzer als `.Acon` übergeben.

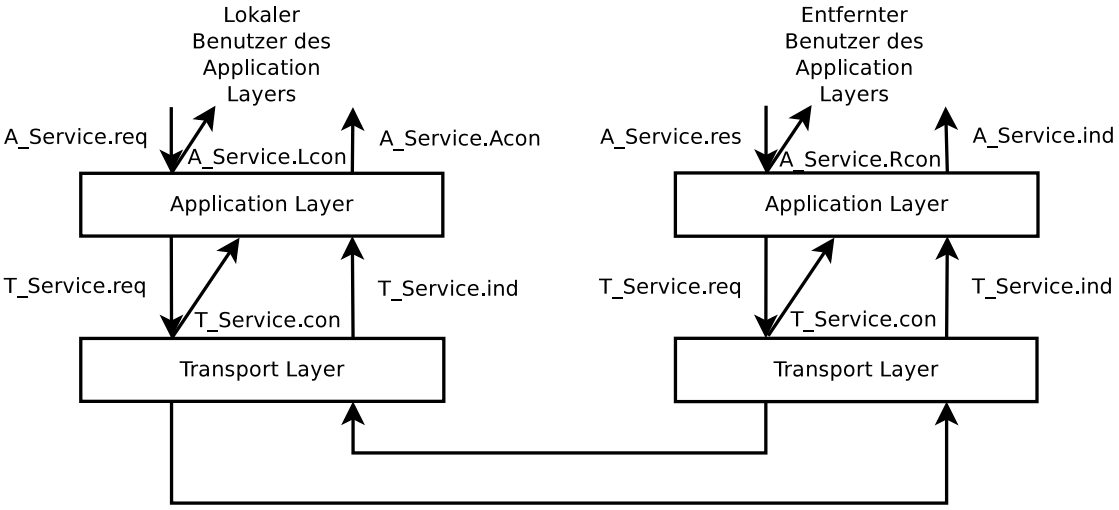

Abbildung 4.5: Interaktivität des Application Layers [32]

Application Interface Layer

Der Application Interface Layer (AIL) ist die Schicht zwischen dem Application Layer und der eigentlichen Anwendung. Seine Aufgabe ist es, der Anwendung die Kommunikation zu abstrahieren, indem eine objektbasierte Kommunikationsschnittstelle zur Verfügung gestellt wird. Der AIL enthält Gruppenobjekte und Schnittstellenobjekte, wobei die Gruppenobjekte über

Multicast-Dienste und die Schnittstellenobjekte über Punkt-zu-Punkt- und Broadcast-Verbindungen angesprochen werden können. Gruppenobjekte können Referenzen auf Schnittstellenobjekte sein. Die Schnittstellenobjekte werden in Systemschnittstellenobjekte und Anwendungsschnittstellenobjekte unterteilt. Die Systemschnittstellenobjekte sind zuständig für Netzwerk- und Gerätemanagement. Die Anwendungsschnittstellenobjekte werden in der Benutzeranwendung definiert. Abbildung 4.6 zeigt den Aufbau des AIL.

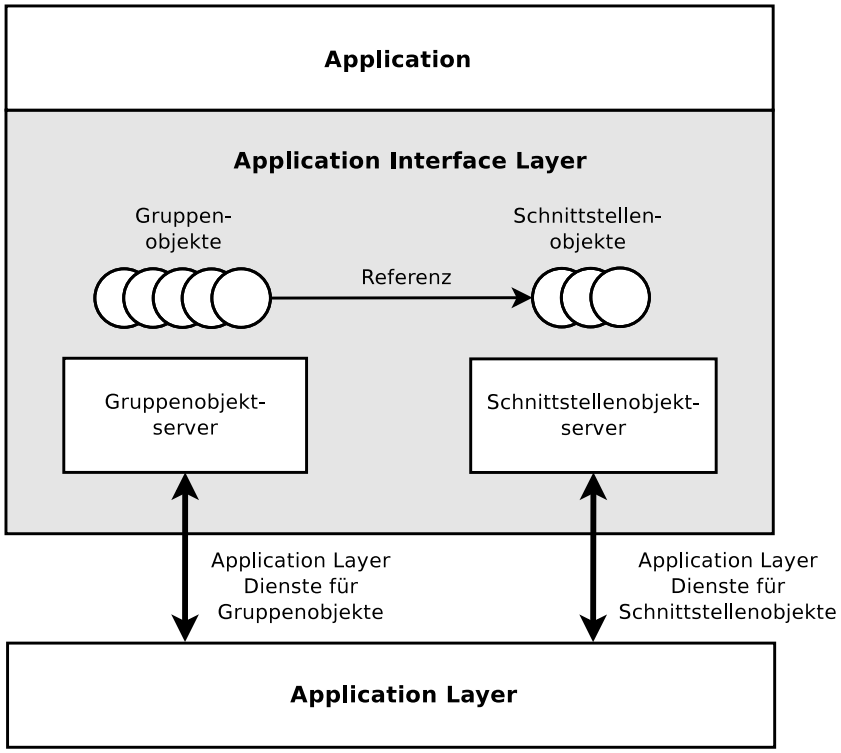

Abbildung 4.6: Application Interface Layer [32]

4.4 Aufbau eines Telegramms

Je nach Übertragungsmedium sind verschiedene Formate der zu übertragenden Telegramme definiert. Das am häufigsten anzutreffende Format ist das TP1-Telegramm. Es wird zwischen dem Standard-Telegramm und dem Extended-Telegramm unterschieden. Beim Standard-Telegramm beträgt die Länge der APDU höchstens 15 Bytes. Für die Übertragung einer APDU mit einer Länge von bis zu 254 Bytes wurde das Extended-Telegramm definiert. Der Aufbau des Extended-Telegramms unterscheidet sich nur geringfügig vom Standard-Telegramm. Es besitzt ein zusätzliches Kontrollfeld am Beginn. Zur Angabe der Länge des Telegramms stehen acht statt vier Bits zur Verfügung.

Die Telegramme beginnen mit dem Kontrollbyte, in dem die Priorität angegeben ist, wobei zwischen Standard- und Extended-Telegramm unterschieden wird. Die Quelladresse identifiziert

den Sender und ist eine physikalische Adresse, während die Zieladresse auch aus einer Gruppenadresse bestehen kann. Der Typ der Zieladresse wird im Adresstyp-Bit angegeben. Nach Angabe der Länge der Daten folgt die Transport Layer Protocol Control Information (TPCI) mit einer Länge von sechs Bits. Sie beschreibt die Kommunikation auf dem Transport Layer und identifiziert den verwendeten Dienst, um zum Beispiel mittels `T_Connect` eine verbindungsorientierte Punkt-zu-Punkt-Kommunikation aufzubauen. Die Application Layer Protocol Control Information (APCI) gibt den verwendeten Dienst des Application Layers an, wie zum Beispiel `A_GroupValue_Write`, um Daten an mehrere Teilnehmer zu senden. Nach der APCI folgen die Nutzdaten selbst. Beendet werden die Telegramme mit einer Prüfsumme, die Datenkonsistenz und eine zuverlässige Übertragung gewährleistet. Abbildung 4.7 zeigt die Struktur des Standard-Telegramms und Abbildung 4.8 die des Extended-Telegramms. Die Abbildungen zeigen außerdem, von welcher Schicht welche Daten hinzufügt werden.

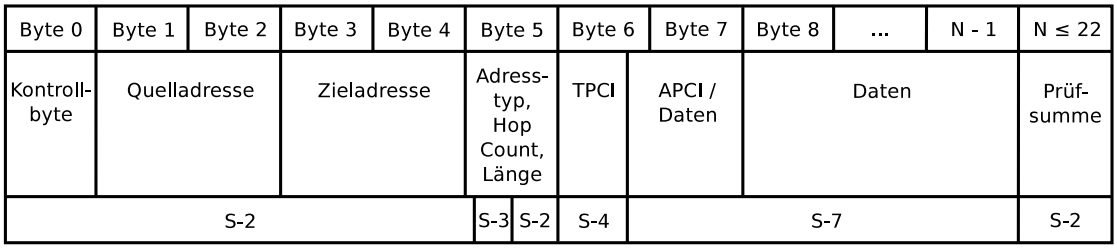

Abbildung 4.7: Struktur des KNX-Standard-Telegramm [32]

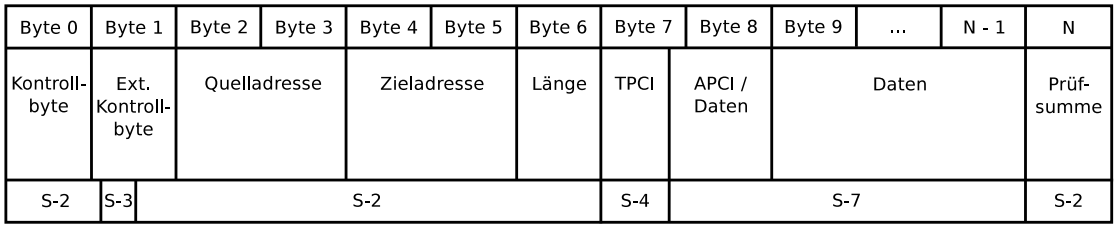

Abbildung 4.8: Struktur des KNX-Extended-Telegramm [32]

Die Telegramme, die über KNX RF übertragen werden, haben ein anderes Format. Es wird hier nicht zwischen Standard- und Extended-Telegramm unterschieden. KNX-RF-Telegramme bestehen aus einer Präambel, mehreren Datenblöcken, gefolgt von je einer zwei Bytes langen CRC sowie einer Postambel. Prä- und Postambel sind von der physikalischen Schicht abhängig, von denen es innerhalb von KNX RF mit KNX RF Ready, KNX RF Multi, KNX RF BiBat und KNX RF BiBat 2 vier verschiedene Ausprägungen gibt.

Der erste Datenblock eines Telegramms hat eine Länge von zehn Bytes und enthält die Länge, die Seriennummer beziehungsweise die Domänenadresse des Senders und andere spezifische Informationen. Die darauf folgenden Datenblöcke sind jeweils 16 Bytes lange, mit Ausnahme des abschließenden Datenblocks, der auch kürzer sein kann. Der zweite Datenblock enthält in den ersten acht Bytes das Kontrollbyte, Quell- und Zieladresse, LPCI, TPCI, eine Sequenznum-

mer und die APCI. Der Rest der Bytes enthält die eigentlichen Nutzdaten. Abbildung 4.9 zeigt den Aufbau eines KNX-RF-Telegramms. Nachrichten, die zwischen KNX-RF und TP1 ausgetauscht werden, werden übersetzt und an das jeweilige Format angepasst.

	10 Byte	2 Byte	16 Byte	2 Byte	...	2 Byte	
Präambel	Datenblock 1	CRC	Datenblock 2	CRC	...	CRC	Postambel

Abbildung 4.9: Struktur des KNX-RF-Frame [32]

4.5 Interworking-Modell

KNX ist ein offener Standard, in dem Geräte unterschiedlicher Hersteller miteinander kommunizieren, was eine Spezifikation der Formate der zu übertragenden Daten erfordert. Ein wesentliches Konzept hierfür sind die Datenpunkte, die Eingaben, Ausgaben, Diagnosedaten, Parameter und andere Werte darstellen können. Sie repräsentieren Prozess- und Kontrollvariablen im System. Um die Zusammenarbeit zu ermöglichen, müssen standardisierte Datenpunkttypen implementiert werden, die auch zu Funktionsblöcken gruppiert werden können. KNX definiert unter anderem Datenpunkttypen für HLK-Anwendungen. Die erforderliche logische Verbindung der Datenpunkte wird *Binding* genannt. Wenn die Anwendung eines Gerätes, wie einem Sensor, den Wert eines zu sendenden Datenpunkts schreibt, dann wird eine Nachricht mit dem neuen Wert gesendet. Die adressierten Datenpunkte der entfernten Geräte empfangen diese Nachricht und geben den neuen Wert an ihre Anwendung weiter. Auf diesem Weg wird eine verteilte Anwendung gebildet.

Das Binding wird in drei verschiedene Kategorien unterteilt: frei, strukturiert und markiert. Bei freiem und strukturiertem Binding trägt der numerische Wert der Adresse keine Semantik der Anwendung. Die einzige Annahme, die gemacht wird ist, dass alle Datenpunkte, die miteinander kommunizieren wollen, die gleiche Adresse zugewiesen haben. Im Gegensatz dazu beinhaltet die Adresse bei markiertem Binding eine semantische Kennung, die auf die Datenpunkte des Kommunikationspartners deutet. Freies Binding ist im S-Mode von zentraler Bedeutung. Der Controller- und Push Button Mode folgen dem Konzept des strukturierten Bindings. Markiertes Binding wird von mehreren Modi eingesetzt. Im LTE-Mode geht markiertes Binding mit der Zoneneinteilung einher.

4.6 Security

Im KNX-Standard ist lediglich eine Zugangskontrolle basierend auf 32 Bits langen Passwörtern, die Schlüssel genannt werden, integriert. Es können bis zu 255 verschiedene Passwörter für bis zu 256 verschiedene Berechtigungsstufen definiert werden. Berechtigungsstufe 0 ist die höchste Stufe mit den maximalen Zugriffsrechten, Berechtigungsstufe $(N - 1)$ die niedrigste und Berechtigungsstufe N hat keine Beschränkungen. Die genaue Anzahl an Berechtigungsstufen N, die ein Gerät unterstützt, ist vom Hersteller abhängig. Für jede der Berechtigungsstufen,

ausgenommen der Letzten, kann ein Passwort festgelegt werden. Tabelle 4.1 verdeutlicht dieses Schema.

Passwort	**Berechtigungsstufe**
Passwort für Stufe 0	0
Passwort für Stufe 1	1
...	...
Passwort für Stufe N-1	N-1
Kein Passwort	N (freier Zugang)

Tabelle 4.1: Passwörter und Berechtigungsstufen [32]

Mit dem Dienst `A_Authorize_Request` kann der Zugriff mit dem Passwort angefordert werden. Der Empfänger durchsucht seine Tabelle an Passwörtern, um zu prüfen, für welche Berechtigungsstufe dieses gültig ist. Die Antwort erfolgt mit einer `A_Authorize_Response`-Nachricht, die die aktuelle Berechtigungsstufe enthält. Diese Berechtigungsstufe ist so lange gültig, bis die Verbindung beendet oder ein neues Passwort übermittelt wird.

Mit dem Dienst `A_Key_Write` können Passwörter geändert oder gelöscht werden. Dazu wird das neue Passwort gemeinsam mit der betreffenden Berechtigungsstufe an den Kommunikationspartner übermittelt. Damit eine Änderung des Passwortes möglich ist, muss die derzeitige Berechtigungsstufe kleiner oder gleich der in den `A_Key_Write`-Daten übermittelten sein. Das bedeutet, dass zuvor mittels `A_Connect` eine Verbindung aufgebaut und eine `A_Authorize_Request`-Nachricht gesendet werden muss [16]. Nach erfolgreicher Änderung des Passwortes wird mit einer `A_Key_Write`-Nachricht geantwortet. Der Inhalt dieser Nachricht ist die Berechtigungsstufe, für die das Passwort geändert wurde. Um ein Passwort zu löschen, muss 0xFFFFFFFF als neues Passwort angegeben werden. In Abbildung 4.10 ist der Vorgang der Autorisierung und die Änderung eines Passwortes dargestellt.

Bei KNX wurde der Sicherheitsaspekt fast vollständig außer acht gelassen, womit es eine Reihe an Angriffspunkten gibt [23]. Der Standard kann weder Datenintegrität, Datenaktualität noch Datenvertraulichkeit garantieren. Die Passwörter zur Zugangskontrolle werden in Klartext übertragen. Die Zugangskontrolle kann nur für Managementfunktionen und nicht für den Austausch von Prozessdaten eingesetzt werden. In KNX gibt es keine Möglichkeit, die Passwörter zu verwalten und zu verteilen, sie müssen vom Anwender manuell konfiguriert werden. Des Weiteren unterstützt das KNX-Protokoll keine parallelen Verbindungen. Das bedeutet, wenn eine Komponente eine Verbindung zu einer anderen hergestellt hat, dann werden andere Verbindungsanfragen ignoriert, was es anfällig für DoS-Angriffe macht.

Auf Grund der fehlenden Sicherheitsmechanismen wurde mit EIBsec [20, 33] eine Erweiterung zu KNX entwickelt, die zur Standardtechnologie kompatibel ist. EIBsec bietet Authentifizierung aller Kommunikationsteilnehmer sowie einen sicheren Kanal zur Datenübertragung, der Datenvertraulichkeit, Integrität und Aktualität garantiert. Die Daten werden mit AES-128 verschlüsselt. Es werden sowohl Gruppenkommunikation als auch Managementdienste gesichert. Die geheimen Schlüssel werden über ein Schlüsselmanagement generiert und an die Teilnehmer

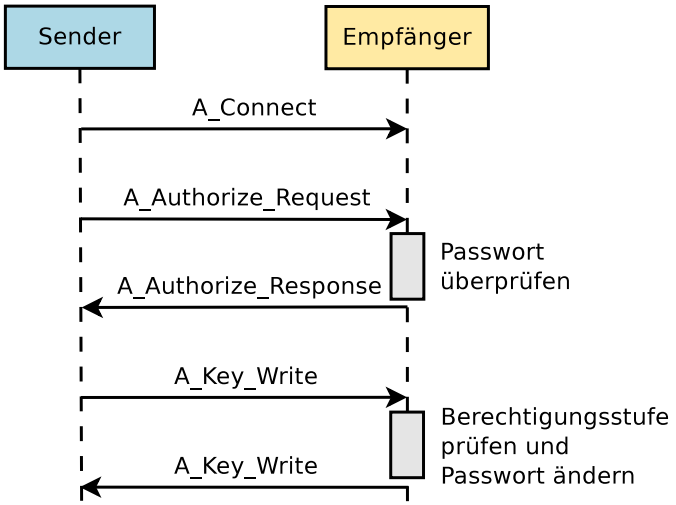

Abbildung 4.10: Autorisierung in KNX

sicher übermittelt. Dabei wird auch die Zurücknahme von unsicheren Schlüsseln unterstützt, wodurch die Begrenzung der Lebensdauer der Schlüssel ermöglicht wird. Die Schlüsselverwaltung wird durch mehrere Schlüsselserver realisiert, wobei jeder dieser Schlüsselserver für eine bestimmte Teilmenge der geheimen Schlüssel zuständig ist. Die Kommunikation mit den Schlüsselservern wird durch initiale geheime Schlüssel gesichert.

KAPITEL 5

KNX Datensicherheit

5.1 Einleitung

Das Sicherheitskonzept von KNX fußt auf zwei Säulen [34]. Einerseits ist *KNX IP Secure* zuständig für die Sicherung von KNXnet/IP-Verbindungen. Es wird dabei auf die speziellen Anforderungen der Kommunikation über IP eingegangen. Dieser eigene Standard ist wegen weiterer Angriffspotentiale bei IP-Verbindungen nötig. Außerdem ist es so möglich, bestehende Subsysteme unverändert zu lassen und lediglich den gefährdeten IP-Backbone abzusichern. Andererseits bietet *KNX Data Security* die Authentifizierung und Verschlüsselung von Nachrichten sowie Autorisierung und Datenaktualität. In dieser Arbeit wird KNX Data Security behandelt und analysiert.

Um Security in der KNX-Kommunikation garantieren zu können, wird der KNX Secure Application Layer (S-AL) [31] in den KNX-Protokoll-Stack integriert. Durch die eingeführten Sicherheitsmechanismen wird eine Verwendung des KNX-Standards auch in sicherheitsrelevanten Domänen ermöglicht. Die Anwendungen werden sowohl ausschließlich sichere Kommunikation als auch sichere und unsichere Verbindungen einsetzen können. Bei sicherer Datenübertragung kann weiters zwischen Authentifizierung und Authentifizierung in Verbindung mit Vertraulichkeit gewählt werden. Die Notwendigkeit von sicherer Kommunikation ist dabei eine Anforderung des Empfängers. Die Security-Dienste können mit allen von KNX unterstützten Übertragungsmedien eingesetzt werden. Insbesondere in Hinsicht auf das drahtlose Medium KNX RF ist eine sichere Kommunikation eine erforderliche Erweiterung. Die Datensicherheit in KNX bezieht sich vor allem auf Endgeräte im S-Mode. Die Ressourcen, Verwaltung und Konfigurationsprozeduren für E-Mode-Geräte sind nicht spezifiziert.

KNX Data Security wird vom Application Layer (AL), dem Secure Application Layer (S-AL) und dem Application Interface Layer (AIL) gehandhabt und mit Hilfe des *Security Interface Objects* verwaltet, in dem und auch die benötigten Werte, wie die Schlüssel, in den jeweiligen Eigenschaften gespeichert werden. Der Dienst `S-A_Data` des S-ALs ist in Senderichtung zuständig für das Sichern des Plain Application Layers (P-ALs). In Empfangsrichtung ist der Dienst für die Überprüfung der Security einer empfangenen PDU zuständig. P-AL und S-AL

sind zwar beides Teile des ALs, sie beeinflussen sich aber nicht gegenseitig. Abbildung 5.1 zeigt die Position des S-ALs innerhalb des KNX-Stacks.

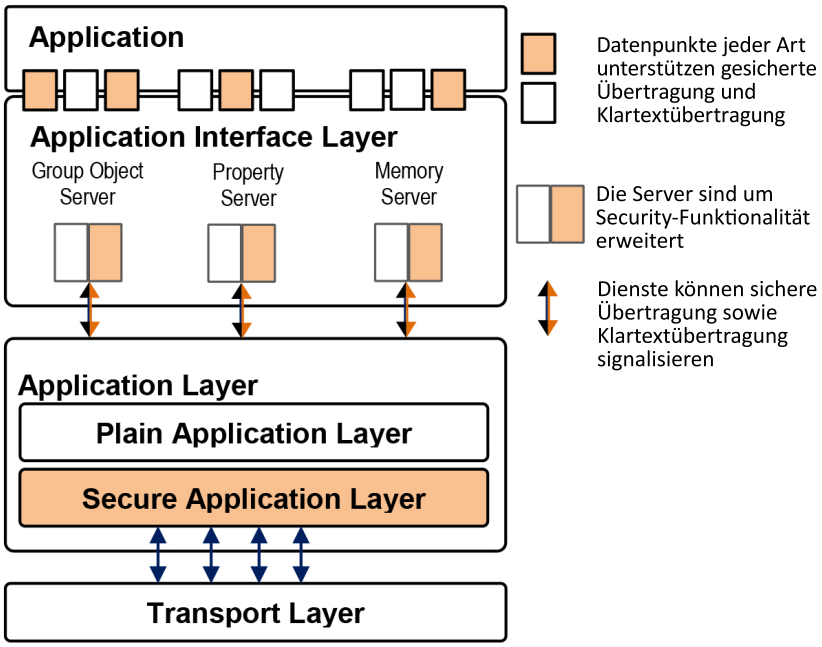

Abbildung 5.1: Datensicherheit im KNX-Stack [31]

5.1.1 Format der sicheren Daten

Eine sichere Nachricht wird durch eine zehn Bit lange APCI, die sogenannte Secure APCI, gekennzeichnet. Auf die Secure APCI folgt das acht Bit lange Security Control Field (SCF), mit dem unter anderem gewählt werden kann, ob nur Authentifizierung oder Authentifizierung und Vertraulichkeit eingesetzt werden soll. Die Secure APCI und das SCF bilden den Secure Header, worauf das sichere KNX-Telegramm die sechs Bytes lange Sequenznummer enthält. Danach folgen die, je nach Security-Modus, verschlüsselten oder unverschlüsselten Nutzdaten. Abschließend enthält das Telegramm den vier Bytes langen MAC.

Für die sicheren Daten werden für die Secure APCI, SCF, Sequenznummer und den MAC insgesamt zwölf Bytes benötigt. Ein TP1-Standardtelegramm bietet 15 ganze Bytes an Nutzdaten. Dem Endanwender stehen bei einem sicheren KNX-Standardtelegramm bei TP1 somit noch drei Bytes zur Verfügung. Das ist für einfache Anwendungen der Heim- und Gebäudeautomation, wie dem Aktivieren oder Deaktivieren von Aktoren, ausreichend. Abbildung 5.2 zeigt den Aufbau einer sicheren Nachricht auf dem Übertragungsmedium TP1.

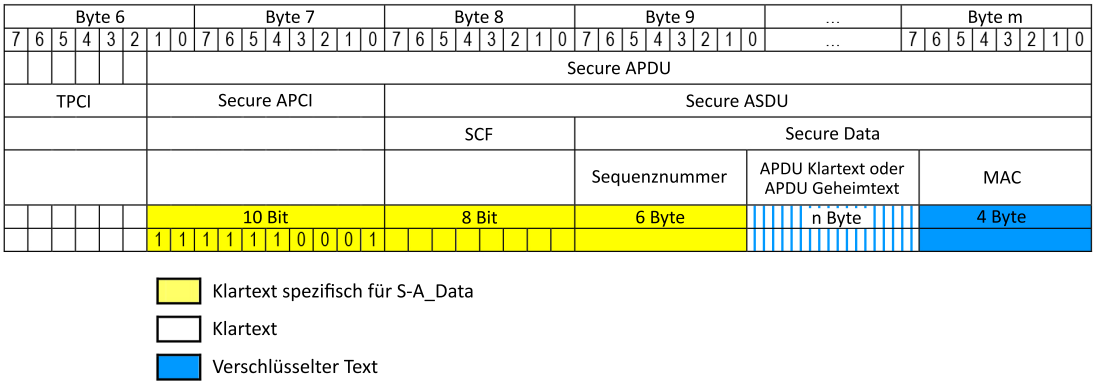

Abbildung 5.2: Format der sicheren Daten (bei TP1) [31]

5.1.2 Schlüssel

Bei der Datensicherheit in KNX werden verschiedene Schlüssel benutzt. Zur erstmaligen Konfiguration und zum initialen Schlüsselaustausch wird der *Factory Default Setup Key* (FDSK) verwendet, der mit dem Gerät ausgeliefert wird. Im Rahmen dieser erstmaligen Konfiguration wird der FDSK mit dem sogenannten *Security Tool Key* ersetzt, der die gleiche Funktion wie der FDSK erfüllt. Wurde der Security Tool Key im Security Interface Object des Gerätes festgelegt, dann wird ab diesem Zeitpunkt dieser Schlüssel und nicht mehr der FDSK verwendet. Durch den Einsatz des FDSKs kann ein Man-in-the-Middle-Angriff beim initialen Schlüsselaustausch ausgeschlossen werden. Der Zugriff auf den FDSK soll von außen weder über das KNX-Übertragungsmedium noch über eine lokale Schnittstelle beim Gerät möglich sein. Die ETS verwendet den Security Tool Key, um auf die verschiedenen Eigenschaften des Security Interface Objects zuzugreifen und für Konfigurationszwecke zu modifizieren. Die Kommunikation unter Verwendung des Security Tool Keys ist nur mit Authentifizierung mit Vertraulichkeit möglich.

Der oder die Schlüssel, die für die einzelnen in der *Security Link Table* definierten sicheren Verbindungen verwendet werden, sind in der *Security Key Table* des Security Interface Objects gespeichert. Auf die Security Key Table kann von außen ausschließlich mit der ETS zugegriffen werden. Die Schlüssel werden von der ETS zufällig generiert. Die Security Link Table enthält für jede ihrer sicheren Verbindungen einen Verweis auf den Schlüssel der Security Key Table, der für diese Verbindung benutzt wird. Es wird pro Sender zumindest ein Schlüssel verlangt, wobei für alle Gruppenadressen eines Senders ein einziger Schlüssel verwendet werden kann.

Die ETS soll alle Schlüssel, den FDSK, den Security Tool Key und die Laufzeitschlüssel, eines jeden Gerätes in ihrer Datenbank speichern. Damit nicht auf unerwünschte Art und Weise außerhalb der ETS auf diese Schlüssel zugegriffen werden kann, werden sie verschlüsselt in der Datenbank der ETS gespeichert. Dadurch ist es nicht möglich, durch Angriffe auf die ETS selbst an die Schlüssel der Geräte zu gelangen.

5.2 Analyse der Security-Mechanismen

Im Folgenden werden die in der KNX-Datensicherheit enthaltenen Security-Mechanismen analysiert. Als erstes werden die Anforderungen einer sicheren Kommunikation auf ihre Einhaltung im S-AL hin überprüft und die Eigenschaften erläutert, um später einen Vergleich zu den Standards BACnet und ZigBee zu ermöglichen. Anschließend wird auf das für KNX leicht abgewandelte CCM und auf die, im Vergleich zu ZigBee und BACnet, kurze Länge des MACs in sicheren KNX-Telegrammen eingegangen. Weiters wird anhand der in der Spezifikation angegebenen Größen der Eigenschaften des Security Interface Objects der Speicherbedarf des S-ALs abgeschätzt.

5.2.1 Erfüllung der Anforderungen

Die in Abschnitt 2.2.2 aufgestellten Anforderungen der Security für einen sicheren Austausch von Daten werden im Folgenden in Bezug auf die Datensicherheit in KNX auf ihre Einhaltung hin untersucht.

Authentifizierung und Datenintegrität

Durch einen MAC wird Gewissheit über den Ursprung der Daten und damit ihre Integrität erlangt [38]. Bei der Datensicherheit in KNX wird der MAC mit AES im Betriebsmodus CCM gebildet und ist 32 Bits lang. Sender und Empfänger müssen einen gemeinsamen geheimen Schlüssel besitzen, mit dem der MAC über die Secure APCI, SCF, Sequenznummer und über die Nutzdaten generiert wird. Durch den MAC können sowohl Übertragungsfehler als auch gewollte Manipulationen erkannt werden. Zur Verifizierung einer Nachricht wird der MAC beim Empfänger von den Daten getrennt. Aus den erhaltenen Daten berechnet der Empfänger mit seinem geheimen Schlüssel ebenfalls einen MAC und vergleicht diesen mit dem empfangenen Wert. Dadurch wird sowohl festgestellt, ob die Daten von der Quelle, die den geheimen Schlüssel besitzt, stammen und somit authentisch sind, als auch, ob Datenintegrität besteht und die Nachricht bei der Übertragung nicht durch einen Angreifer oder durch einen Übertragungsfehler geändert wurde. Schlägt diese Überprüfung des empfangenen MACs fehl, dann wird die Nachricht ignoriert. Abbildung 5.3 zeigt die Stationen, die der MAC von der Erstellung beim Sender bis zum Vergleich beim Empfänger durchläuft.

Autorisierung

Nach einer erfolgreichen Authentifizierung wird im Rahmen der Autorisierung festgestellt, ob für eine bestimmte Aktion die benötigten Rechte vorliegen. Für jede sichere Verbindung können bei der Datensicherheit in KNX Berechtigungen vergeben werden, die festlegen, auf welche Datenpunkte und Dienste des Empfängers der Kommunikationspartner zugreifen kann. Da es eine Vielzahl an Kombinationen von sicheren Verbindungen und sicheren Datenpunkten oder Diensten gibt, können die Verbindungen mit den gleichen Berechtigungen zu sogenannten Rollen zusammengefasst werden.

Die Überprüfung der Berechtigungen liegt im Aufgabenbereich des AIL. Ein Sender kennt seine Berechtigungen in einem Empfänger nicht, das liegt in der Konfiguration des Empfängers.

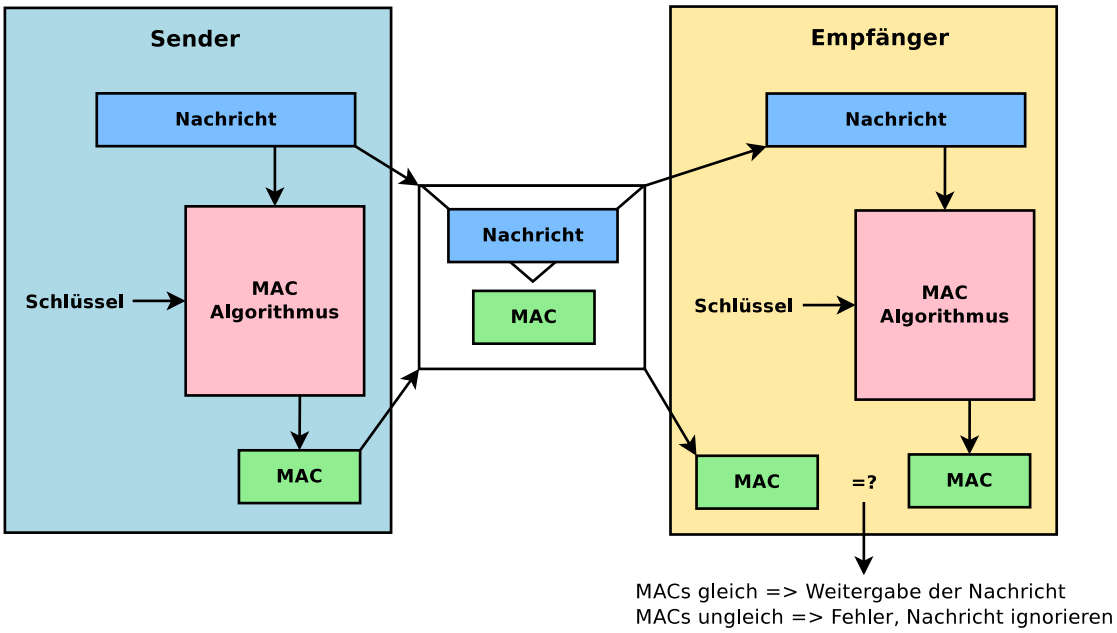

Abbildung 5.3: Generierung, Übertragung und Verifizierung des MACs

Die Berechtigungen und Rollen sind in der Anwendung festgelegt, es ist aber der Benutzer der ETS, der die Rollen der Sender den Empfängern zuteilt. Für einen Datenpunkt oder Dienst können mehrere Rollen mit unterschiedlichen Berechtigungen definiert sein.

Die Ressourcen für die Rollen sind nicht standardisiert und von der Implementierung abhängig. Die Berechtigungen werden in den Anwendungen definiert, es soll jedoch Datenpunkte und Dienste geben, auf die, unabhängig vom aktivierten Security-Modus, immer beziehungsweise niemals mittels Kommunikation im Klartext zugegriffen werden kann. Für die Umsetzung dieser Anforderung bestehen die *Security White Lists* sowie die *Security Black Lists*. Security White Lists geben die Datenpunkte und Dienste an, die in ungesicherter Kommunikation immer akzeptiert werden sollen. Security Black Lists wiederum geben an, welche Datenpunkte und Dienste nur mittels sicherer Kommunikation mit Authentifizierung und Vertraulichkeit akzeptiert werden sollen. Wie auch die restlichen Berechtigungen sind die Security White Lists und Security Black Lists keine standardisierten Ressourcen und müssen bei den von der Implementierung abhängigen Berechtigungen mit berücksichtigt werden.

Datenursprungsauthentifizierung

Mechanismen zur Datenursprungsauthentifizierung, die auf geteilten geheimen Schlüssel basieren, so wie es bei der Datensicherheit in KNX beim MAC der Fall ist, erlauben keine Unterscheidung zwischen den Beteiligten, die den gemeinsamen Schlüssel besitzen [38]. Jeder Teilnehmer, der im Besitz des geteilten Schlüssels ist, kann mit diesem eine Nachricht erzeugen. Der oder die Empfänger können von solch einer Nachricht nicht mit Gewissheit feststellen, ob sie von einem bestimmten Teilnehmer mit dem geteilten Schlüssel stammt. Wird ein Schlüssel zwischen mehr

als zwei Teilnehmern geteilt, dann kann somit keine Datenursprungsauthentifizierung garantiert werden.

Jedes KNX-Gerät kann für seine sicheren Punkt-zu-Punkt-Verbindungen und für seine sicheren Gruppenkommunikationen jeweils einen eigenen Schlüssel besitzen. Laut Spezifikation der Datensicherheit in KNX ist allerdings lediglich ein Schlüssel pro Sender erforderlich und ein einziger Schlüssel kann für alle Gruppenadressen des Senders benutzt werden. Für die Punkt-zu-Punkt-Verbindungen ist Datenursprungsauthentifizierung bei entsprechender Konfiguration somit prinzipiell möglich, wenn jede sichere Verbindung ihren eigenen geheimen Schlüssel zugewiesen bekommt. Eine Beschränkung stellt die Anzahl der Geräte und der Speicher dieser dar. Umso mehr Punkt-zu-Punkt-Verbindungen mit einmaligem geheimen Schlüssel eingesetzt werden, umso öfter kann eine Datenursprungsauthentifizierung stattfinden, doch umso mehr Speicher werden für die Daten dieser sicheren Verbindungen und die zugehörigen Schlüssel benötigt.

Eine Überprüfung im Gerät, ob ein Schlüssel tatsächlich nur für eine einzige Punkt-zu-Punkt-Verbindung und nicht auch noch für andere sichere Verbindungen genutzt wird, müsste manuell von der Anwendung erfolgen, indem die Spalte *Key Index* der Security Link Table auf mehrfache Einträge hin durchsucht wird. Da bei Gruppenkommunikationen mehrere Teilnehmer im Besitz eines Schlüssels sind, ist in diesem Kommunikationsmodus keine Datenursprungsauthentifizierung gegeben.

Datenaktualität

Durch die Sequenznummer wird Datenaktualität garantiert und Replay-Angriffe werden verhindert. Die Sequenznummer ist ein streng monoton steigender Zähler, der vom Sender für jedes gesendete sichere Telegramm erhöht wird. Ein Sender verfügt über eine Sequenznummer für alle seine sicheren Verbindungen, es gibt keine verschiedenen Sequenznummern pro Verbindung oder Datenpunkt. Die Sequenznummer ist sechs Bytes lange was, ausgehend von KNX RF als Übertragungsmedium und einer durchschnittlichen Länge eines KNX-Telegramms von 23 Bytes, ausreichend ist, damit es über 100.000 Jahre dauert, bis es zu einem Überlauf des Wertes kommt [31].

Gespeichert wird die Sequenznummer im Sender in der Eigenschaft *Sequence Number Sending* des Security Interface Objects. Der Empfänger speichert in der Eigenschaft *Security Individual Address Table* für jede physikalische Adresse seiner sicheren Verbindungen die letzte gültige Sequenznummer zu Vergleichszwecken. Wird eine Sequenznummer empfangen, dann vergleicht der Empfänger diesen Wert mit jenem, den er als letzte gültige Sequenznummer für diese Absenderadresse gespeichert hat. Die empfangene Nachricht wird nur akzeptiert, wenn ihre Sequenznummer höher als die letzte gültige gespeicherte Sequenznummer ist. Zusätzlich zur Sequenznummer ist bei KNX keine Zeitinformation enthalten, wodurch die Gültigkeitsdauer von Nachrichten nicht begrenzt werden kann und eine empfangene Nachricht immer gültig ist, solange ihre Sequenznummer höher als die der vorangegangen Nachricht dieser physikalischen Adresse als Absender ist.

Datenvertraulichkeit

Die zu sendenden Daten können verschlüsselt werden, was mit AES im Betriebsmodus CCM mit einer Schlüssellänge von 128 Bits geschieht. Datenvertraulichkeit wird immer in Zusammenhang mit Authentifizierung angewendet. Wird dieser Security-Modus gewählt, dann werden die Nutzdaten der Anwendungsschicht verschlüsselt und der MAC über Secure APCI, SCF, Sequenznummer und die Nutzdaten gebildet. Die Daten der Schichten unterhalb der Anwendungsschicht werden immer unverschlüsselt übertragen.

Datenverfügbarkeit

Datenverfügbarkeit bedeutet, dass die Dienste und Daten des Systems den autorisierten Geräten zur Verfügung stehen. Bestimmte Bedrohungen, wie DoS-Angriffe, die auf die Verfügbarkeit eines Systems abzielen, können nicht durch kryptographische Methoden, wie die Security-Mechanismen des S-ALs verhindert werden und erfordern zusätzliche physikalische Maßnahmen, worauf später in Abschnitt 5.3.4 eingegangen wird.

5.2.2 CCM in KNX

Für die Berechnung des MAC werden im CCM-Algorithmus eine Sequenz an Blöcken B_i verwendet, wobei B_0 den ersten dieser Blöcke darstellt. Zur Verschlüsselung der Daten wird der CTR-Modus eingesetzt, für den die Counter-Blöcke Ctr_j benötigt werden. Wie in Abschnitt 3.1.4 erläutert, besteht im CCM der Block B_0 aus einer Nonce N, der Bytelänge Q der Nutzdaten P und einem Byte mit Flags. Die CTR-Blöcke Ctr_j bestehen aus der Nonce N, einem Zähler j und ebenfalls einem Byte mit verschiedenen Flags. Für die sichere KNX-Kommunikation haben die Blöcke B_0 und Ctr_j eine abgeänderte Struktur, in der für KNX spezifische Werte enthalten sind. Dadurch, dass diese spezifischen Werte somit direkt in den CCM- beziehungsweise CTR-Prozess eingebunden sind, können sie bei der Übertragung nicht geändert werden, ohne dass diese Aktion unentdeckt bleibt.

Der für CCM in der Datensicherheit in KNX abgeänderte Block B_0 enthält unter anderem die Sequenznummer sowie die Quell- und Zieladresse. Tabelle 5.1 zeigt noch einmal den Aufbau von B_0 in CCM, so wie ursprünglich spezifiziert. Tabelle 5.2 zeigt im Vergleich dazu die genaue Formatierung von B_0 in KNX. QA steht für die Quelladresse, ZA für die Zieladresse, FT zeigt an, ob es sich um ein Standard- oder Extended-Telegramm handelt, AT ist der Adresstyp des KNX-Telegramms und Q gibt die Bytelänge der Nutzdaten an.

Byte Nummer	0	$1 \ldots (15-q)$	$(16-q) \ldots 15$
Inhalt	Flags	N	Q

Tabelle 5.1: Formatierung von B_0 in CCM [11]

Ebenfalls abgeändert ist das Format der Counter-Blöcke Ctr_j, die in der für KNX modifizierten Variante neben dem eigentlichen Zähler j ebenfalls aus der Sequenznummer sowie der Quell- und Zieladresse bestehen. Tabelle 5.3 zeigt die Struktur von Ctr_j in CCM. Tabelle 5.4

Byte Nummer	15	...	10	9	8	7	6	5	4	3	2	1	0
Inhalt	SeqNr			QA		ZA		FT	AT	TPCI	0x00	0x00	Q

Tabelle 5.2: Formatierung von B_0 von CCM in KNX [31]

zeigt zum Vergleich den genauen Aufbau von Ctr_j in der für KNX abgeänderten Variante. Die Bytelänge der Nutzdaten Q des Blocks B_0 sowie der Counter-Wert j der Blöcke Ctr_j sind nur ein Byte lang, was allerdings auf Grund der maximalen Länge der Nutzdaten von 242 Bytes des KNX-Extended-Telegramms (siehe Abschnitt 4.4) ausreichend ist.

Byte Nummer	0	1 ... $(15-q)$	$(16-q)$ ... 15
Inhalt	Flags	N	Counter j

Tabelle 5.3: Formatierung von Ctr_j in CCM [11]

Byte Nummer	15	...	10	9	8	7	6	5	4	3	2	1	0
Inhalt	SeqNr			QA		ZA		0x00	0x00	0x00	0x00	0x01	j

Tabelle 5.4: Formatierung von Ctr_j von CCM in KNX [31]

Da CCM im S-AL geringfügig abgeändert wurde, indem die Blöcke B_0 und Ctr_j für KNX spezifische Werte enthalten, muss sichergestellt werden, dass die Sicherheit durch diese Modifizierung des Verfahrens nicht beeinträchtigt wird. Der in Abschnitt 3.1.4 erwähnte Beweis für die Sicherheit von CCM [27] soll auch für die in KNX verwendete Variante von CCM gelten. In dem Beweis wird eine Codier-Funktion β eingeführt, die eine gültige Eingabe bestehend aus der Nonce N, den zusätzlichen zu authentifizierenden Daten genannt Header H und der Nachricht M verarbeitet. Die Funktion β leitet aus der Eingabe (N, H, M) eine Zeichenkette

$$B = B_0.B_1.\cdots.B_r$$

an CBC-MAC-Blöcken $B_0, ..., B_r$ ab. Für β muss die Bedingung erfüllt sein, dass N durch den ersten Block B_0 von $\beta(N, H, M)$ eindeutig bestimmt ist.

Für die Verschlüsselung im Modus CTR wird im Beweis ein CTR-Blockgenerator π mit den vier Argumenten j, N, H und $|M|$, der Länge von M, verwendet. Die CTR-Blöcke Ctr_j mit $j = 0,\ ...,\ m$ sind definiert als

$$Ctr_j = \pi(j, N, H, |M|).$$

Die Nonce N und der Zähler j sind dabei durch den CTR-Block $\pi(j, N, H, |M|)$ eindeutig bestimmt. Für j gilt $0 \leq j \leq m$, wobei $m = 255$ im Fall des S-ALs. Die Anforderung an die Nonce N lautet, dass sie sich unter einem Schlüssel sowohl bei der Verwendung in den Blöcken B_0 als auch bei Ctr_j jeweils nicht wiederholt.

Zusammenfassend müssen für die Gültigkeit des Beweises der Sicherheit in CCM folgende Bedingungen erfüllt sein:

- Für die Funktion β muss N durch B_0 eindeutig bestimmt sein.
- Für den CTR-Blockgenerator π müssen N und j durch Ctr_j eindeutig bestimmt sein.
- N darf sich unter einem Schlüssel nicht wiederholen.
- B_0 muss von den Blöcken Ctr_j verschieden sein.

Die Bedingung an β, dass N durch B_0 sowie die Bedingung, dass N und j durch die CTR-Blöcke eindeutig bestimmt sind, treffen auch in der Version von CCM mit den spezifischen Werten zu. Die Nonce N enthält die Sequenznummer, die Quelladresse und die Zieladresse und wird somit durch B_0 beziehungsweise Ctr_j eindeutig bestimmt, wie auch in den Tabellen 5.2 und 5.4 dargestellt ist. Weiters darf sich die Nonce N für einen Schlüssel nicht wiederholen, um die Sicherheit in CCM nicht zu gefährden, was dadurch erfüllt ist, dass sich N aus der Sequenznummer in Kombination mit der Quelladresse, einer eindeutigen physikalischen Adresse, zusammensetzt. Wie in Abschnitt 5.2.1 erläutert, ist die Sequenznummer eines Senders ein einmaliger Wert, der sich für alle sicheren Kommunikationen für jede zu sendende Nachricht erhöht. Daraus folgt, dass die Sequenznummer in Kombination mit der Quelladresse einen eindeutigen Wert darstellt, womit eine sich nicht wiederholende Nonce N garantiert ist.

Damit sichergestellt ist, dass B_0 und die CTR-Blöcke Ctr_j jeweils nicht gleich sind, enthalten die Bytes Nummer 1 der Blöcke voneinander verschiedene Werte, wie in den Tabellen 5.2 und 5.4 zu erkennen ist. Somit sind die Anforderungen an den Beweis der Sicherheit von CCM auch für die in KNX verwendete Variante von CCM mit den geänderten Formaten der Blöcke B_0 und Ctr_j gegeben und der Beweis ist auch für CCM in KNX gültig.

Die in Abschnitt 3.1.4 erwähnte maximale Anzahl der Aufrufe der Verschlüsselungsoperationen unter einem Schlüssel wird ebenfalls erfüllt. Die Verschlüsselungsoperationen für CBC-MAC und der Daten sollen demnach zusammen höchstens 2^{61} Mal aufgerufen werden, was einer Datenmenge von 2^{64} Bytes beziehungsweise 16 Millionen Terabytes entspricht. Bei dem drahtlosen Übertragungsmedium KNX RF mit einer Übertragungsgeschwindigkeit von 16.384 Bits/s werden über 285 Millionen Jahre benötigt, um eine Datenmenge von 2^{64} Bytes zu versenden, wodurch es nicht möglich ist, diese Grenze zu überschreiten und es folglich keine Beschränkung in der Anwendung implementiert werden muss. Beim langsameren KNX TP1 werden mit über 487 Millionen Jahre noch einmal rund 200 Millionen Jahre länger benötigt.

5.2.3 Länge des MACs

Mit dem MAC wird die Integrität der Nachricht festgestellt, wofür es allerdings keine absolute Garantie gibt. Ein Angreifer kann mit einer gewissen Wahrscheinlichkeit einen gültigen Geheimtext erstellen, der die Verifizierung im Rahmen des Entschlüsselungsprozesses besteht [11]. Die Bitlänge $Tlen$ des MACs ist ein wichtiger Parameter für die Sicherheit. Bei einem MAC mit einer Länge von $Tlen$ Bits gibt es 2^{Tlen} verschiedene mögliche MACs. Ein Angreifer kann alle Möglichkeiten ausprobieren, um einen gültigen MAC für eine Nachricht zu finden. Nach

durchschnittlich 2^{Tlen-1} Versuchen kann ein Angreifer einen gültigen MAC für ein bestimmtes Telegramm finden, das die Verifizierung besteht. Die Gültigkeit eines dieser geratenen MACs kann nicht lokal, sondern nur durch das Senden an einen Empfänger im Netzwerk getestet werden. Es müssen somit tatsächlich durchschnittlich 2^{Tlen-1} Telegramme gesendet werden, um einen gültigen MAC für ein bestimmtes Telegramm finden zu können.

Die Länge des MAC kann bei CCM ein beliebiges Vielfaches von 16 zwischen 32 und 128 Bits betragen. Durch einen längeren MAC kann eine zuverlässigere Authentifizierung garantiert werden, gleichzeitig wird auch mehr Bandbreite und Speicherplatz für den Geheimtext benötigt. Die Länge des MAC sollte laut NIST [11] ohne einer sorgfältigen Analyse der Risiken, die durch das Akzeptieren einer eigentlich nicht authentischen Nachricht entstehen, nicht weniger als 64 Bits betragen. Ein MAC, der kürzer als 64 Bits ist, soll nur genutzt werden, wenn die Anzahl der ungültigen Verifizierungen, die im Zuge der Entschlüsselungen durchgeführt werden, über alle Implementierungen unter jedem beliebigen Schlüssel beschränkt sind.

Die Länge des MACs von 32 Bits in der sicheren KNX-Kommunikation hat den Vorteil von geringerem Overhead. Alternativ oder ergänzend zu einer Limitierung der fehlgeschlagenen Überprüfungen des MACs unter allen Implementierungen unter einem beliebigen Schlüssel kann auch eine kurze Sitzungsdauer oder allgemeiner eine geringe Bandbreite des Übertragungsmediums eine Vielzahl an Versuchen, einen gültigen Geheimtext zu finden, einschränken [11]. In konventionellen Netzwerken würde ein MAC von 32 Bits Länge für keine ausreichende Sicherstellung der Authentifizierung sorgen. In Sensornetzen kann ein MAC mit 32 Bits Länge auf Grund der geringeren Übertragungsgeschwindigkeit jedoch ein ausreichendes Maß an Security zur Verfügung stellen [28].

Die Übertragungsgeschwindigkeit von KNX RF beträgt 16.384 Bits/s. Ein KNX-Telegramm, das die Informationen enthält, die zur sicheren Datenübertragung benötigt werden (SCF, Sequenznummer und MAC), plus einem Byte an Nutzdaten, besteht beim Format des Übertragungsmediums RF aus insgesamt 36 Bytes. Ein Byte an Nutzdaten wurde hier gewählt, weil das die kürzeste Nachricht und somit den ungünstigsten Fall darstellt. Anhand dieser Werte lässt sich die maximale Anzahl an zu übertragenden KNX-RF-Telegrammen berechnen, was einen Wert von 56,89 möglichen Nachrichten pro Sekunde ergibt, die versendet werden können, um einen gültigen MAC für ein bestimmtes Telegramm zu finden. Um die durchschnittlichen 2^{31} Möglichkeiten auszuprobieren, müssten bei KNX RF 36,4 Monate lang konstant Nachrichten gesendet werden. Bei TP1, mit der niedrigeren Übertragungsrate von 9600 Bits/s, aber auch der kürzeren Paketlänge von 20 Bytes bei einer sicheren Nachricht mit einem Byte an Nutzdaten, würde ein Angreifer immerhin noch 34,52 Monate benötigen, um 2^{31} Möglichkeiten zu testen. Das Senden von solch einer Menge an Nachrichten wäre ein effektiver DoS-Angriff, da das jeweilige Übertragungsmedium für die Dauer dieses Zeitraums belegt wäre.

Das Erkennen eines Angriffs, bei dem versucht wird, einen gültigen MAC zu finden, ist dennoch wünschenswert. Als simple heuristische Methode können die Knoten melden, wenn die fehlgeschlagenen Überprüfungen von MACs einen gewissen Wert überschreiten [28]. In der Datensicherheit in KNX ist das Zählen von fehlgeschlagenen Überprüfungen der MACs allerdings nicht eindeutig möglich. Auftretende Security-Fehler werden zwar mit den Zählern SFCC und SFCL registriert, sie erlauben für diesen Zweck jedoch keine zureichend genaue Unterscheidung der Fehlerursache. Der SFCC erhöht sich für jeden Fehler in jedem beliebigen

für die Datensicherheit zuständigen Teil eines Gerätes. Der optionale und für weniger simple Geräte vorgesehene SFCL besitzt separate Zähler für jede einzelne der Verbindungen des Gerätes, wobei der jeweilige Zähler für Fehler bei der Authentifizierung sowie Vertraulichkeit im S-AL beziehungsweise für Fehler bei den Rollen und Berechtigungen im AIL erhöht wird. Eine Fehlerbehandlung, wenn ein bestimmter Höchstwert eines Zählers erreicht wurde, ist in der Spezifikation des S-ALs nicht vorgesehen und müsste in der Anwendung umgesetzt werden. Auf Wunsch kann nach einem aufgetretenen Fehler sowie bei Erreichen des Höchstwertes des SFCCs und SFCLs ein Security Report gesendet werden, woraufhin mit der ETS die Zähler für Diagnosezwecke ausgelesen werden können.

Wird in der Datensicherheit in KNX eine nicht authentische Nachricht von einem Gerät dennoch angenommen, dann wird die Nachricht vom S-AL an den P-AL weitergegeben. Das geschieht auch im Fall, wenn für eine Nachricht mit willkürlichem Geheimtext ein gültiger MAC gefunden wird und der daraus entschlüsselte Klartext keine Bedeutung hat. Wie mit einem unbrauchbaren Klartext umgegangen wird, liegt im Aufgabenbereich der jeweiligen Anwendung.

Zusammenfassend bietet aufgrund der beschränkten Übertragungsgeschwindigkeit der Medien KNX RF sowie TP1 ein MAC mit einer Länge von 32 Bits ausreichende Sicherheit vor eigentlich nicht authentischen Nachrichten. Bei Übertragungsmedien mit höheren Kapazitäten ist, je nach Anforderungen an die Security, unter Umständen ein längerer MAC gefordert. Anforderungen nach einer Beschränkung der fehlgeschlagenen Verifizierungen von MACs mit anschließender Fehlerbehandlung oder zumindest Meldung ist nicht direkt mit den Mitteln des S-ALs umsetzbar.

5.2.4 Schätzung des Speicherbedarfs

Die Eigenschaften des Security Interface Objects, in dem die für die sicheren Verbindungen nötigen Informationen gespeichert werden, haben festgelegte Größen, wodurch der Speicherbedarf abgeschätzt werden kann. Ausgenommen davon sind jedoch die Tabellen, in denen die Berechtigungen gespeichert werden, da diese nicht spezifiziert, sondern von der jeweiligen Implementierung abhängig sind. Besonders speicherintensive Eigenschaften des Security Interface Objects sind die Security Link Table mit 14 Bytes pro Eintrag sowie die 16 Bytes langen Schlüssel in der Security Key Table. Tabelle 5.5 zeigt die genauen Werte.

In Tabelle 5.6 ist der Speicherbedarf für unterschiedlich viele Verbindungen angegeben. Die Werte sind anhand der in der Spezifikation angegebenen Größen der Eigenschaften des Security Interface Objects berechnet, die in Tabelle 5.5 zu sehen sind. Die Anzahl der Verbindungen bezieht sich dabei auf die Anzahl der Einträge in der Security Link Table. Es wird dabei von einem Gruppenobjekt (es wird ein Byte für jedes Gruppenobjekt benötigt) und von jeweils halb so vielen physikalischen Adressen in der Security Individual Address Table wie Einträgen in der Security Link Table ausgegangen.

Die Speicherung der Berechtigungen und Rollen sind von der Implementierung abhängig. Somit ist es nicht möglich, genaue Angaben über den Speicherbedarf zu machen. Im Anhang C der Spezifikation werden jedoch Möglichkeiten der Umsetzung vorgestellt. Tabelle 5.7 zeigt den Speicherbedarf der simplen Berechtigungstabelle, in der jedes Gruppenobjekt und jeder Dienst seine eigene Berechtigungstabelle hat, wobei für jede Rolle 4 Bits benötigt werden. Die

Eigenschaft	Größe
Security Link Table	14 Bytes pro Link-Index
Security Key Table	16 Bytes pro Schlüssel
Security Individual Address Table	8 Bytes pro physikalischer Adresse
Security Failure Counters on Links	4 Bytes pro Link-Index
Security Tool Key	16 Bytes
Security Report Control	1 Byte
Factory Default Setup Key	16 Bytes
Sequence Number Sending	6 Bytes
Role Table	3 Bytes pro Link-Index
Group Object Security Flags	1 Byte pro Gruppenbjekt
Security Failure Common Counter	2 Bytes

Tabelle 5.5: Speicherbedarf der Eigenschaften des Security Interface Objects

Verbindungen	Schlüssel	Größe
1	1	87 Bytes
4	2	174 Bytes
10	10	452 Bytes
30	10	952 Bytes
100	50	3342 Bytes
1000	50	25 842 Bytes
1000	500	33 042 Bytes

Tabelle 5.6: Unterschiedlicher Speicherbedarf bei unterschiedlichen Konfigurationen

Anzahl der Rollen vervielfachen die Anzahl der benötigten Bits somit und sorgen dadurch für den größten Anstieg an benötigtem Speicher.

Sensoren oder Aktoren, die lediglich Mess- oder Stellwerte an eine Station senden beziehungsweise davon empfangen, benötigen nur eine oder wenige sichere Verbindungen für wenige Gruppenobjekte, so dass der Speicherverbrauch für die sicheren Verbindungen sowie die Berechtigungen im unteren Bereich liegen. Wie in Tabelle 5.6 zu sehen ist, werden für 4 Verbindungen mit 2 Schlüssel lediglich 175 Bytes benötigt. Eine simple Berechtigungstabelle mit einem Gruppenobjekt sowie 3 Diensten und 5 Rollen verlangt 10 Bytes an Speicher, wie aus Tabelle 5.7 zu entnehmen ist. Typischerweise wird ein Mikrocontroller verwendet, zum Beispiel ein ATmega 16, der 16 Kilobytes an Flash-Speicher und 512 Bytes an EEPROM bietet, was ausreichend ist, um die Konfigurationen von wenigen sicheren Verbindungen und Berechtigungen

Gruppen-objekte	Dienste	Größe pro Rolle	Rollen	Gesamtgröße
1	3	2 Bytes	5	10 Bytes
			25	50 Bytes
10	10	10 Bytes	5	50 Bytes
			25	250 Bytes
50	16	33 Bytes	8	240 Bytes
			30	900 Bytes
70	30	50 Bytes	10	500 Bytes
			35	1750 Bytes

Tabelle 5.7: Speicherbedarf simple Berechtigungstabelle

zu speichern.

5.3 Angriffe

In diesem Abschnitt werden Angriffe auf die sichere KNX-Kommunikation analysiert und eventuell auftretende Probleme aufgezeigt.

5.3.1 Mithören der Kommunikation

Ein nicht autorisierter Teilnehmer verschafft sich bei diesem Angriff Zugang zum System, um Informationen mitzulesen und gegebenenfalls auch aufzuzeichnen. Beispiele dafür sind das „Anzapfen“ von Leitungen beziehungsweise – im Fall von drahtloser Datenübertragung – das einfache Mithören des Funkverkehrs, um die Daten zu lesen.

Network-Sniffing ist ein typischer Angriff dieser Art, in dem der Datenverkehr in einem Netzwerk mit Hilfe einer Sniffing-Software von einer außenstehenden Partei mitgelesen wird. Es handelt sich dabei um einen passiven Angriff auf das Netzwerk, wodurch er schwer zu entdecken ist. Bei drahtloser Übertragung muss dazu kein physischer Zugang zu den Leitungen des Netzwerks bestehen, was den Angriff komplett unauffällig macht.

Erfolgt die KNX-Kommunikation ohne Sicherheitsmaßnahmen oder nur mit aktivierter Authentifizierung, dann werden die Nutzdaten im Klartext übertragen, wodurch ein Angreifer den Inhalt der Datenübertragung mitlesen kann. In der Gebäudeautomation könnten so beispielsweise Daten von Zugangskontrollsystemen ausgelesen werden. Sensible Daten sollen deshalb mit aktivierter Verschlüsselung übertragen werden. Ein relevanter Teil der Sicherheit des Systems wird somit bei der Konfiguration festgelegt, da hierbei in der Security Link Table die sicheren Verbindungen definiert werden und im Feld *Security Parameters for Links* die Einstellung festlegt wird, ob Authentifizierung oder Authentifizierung mit Vertraulichkeit gewünscht ist.

Bei verschlüsselter Übertragung sind nur die Nutzdaten eines Telegramms verschlüsselt, also die Daten der Anwendungsschicht. Der Header eines Telegramms, die Metadaten, bleiben unverschlüsselt, sie können somit dennoch mitgelesen werden. Dadurch kann ein Angreifer Informationen über Quell- und Zieladresse, die Sequenznummer und andere Metadaten mitlesen. Das Wissen über diese Metadaten erlaubt zwar keinen direkten Angriff, ermöglicht es aber Informationen über die sicheren Verbindungen zu sammeln, um in weiterer Folge Angriffe effektiver gestalten zu können. Sichere Nachrichten werden nur von Geräten angenommen, mit denen eine sichere Kommunikationsbeziehung besteht. Möchte ein Angreifer an ein Gerät eine gefälschte sichere Nachricht senden, um zum Beispiel durch Ausprobieren einen MAC herauszufinden, so ist es für den Angreifer von Vorteil zu wissen, welche sicheren Kommunikationsbeziehungen bestehen. Sichere Nachrichten von Adressen, die nicht in der Security Link Table eines Empfängers enthalten sind, werden von Beginn an ignoriert.

5.3.2 Replay-Angriff

Beim Replay-Angriff werden vom Angreifer Datenübertragungen im Netzwerk aufgezeichnet, wie zuvor in Punkt 5.3.1 beschrieben, und zu einem späteren Zeitpunkt erneut im Netzwerk gesendet. Auf diesem Weg kann der Angreifer nicht autorisierten Zugang erhalten. In der Gebäudeautomation könnten dadurch Aktionen, wie zum Beispiel das erneute Öffnen einer Tür, erreicht werden. Verschlüsselung und Authentifizierung alleine schützen nicht vor dem Angriff, da die verschlüsselten und authentifizierten Daten einfach erneut verwendet werden können. Im Fall von Verschlüsselung kann der Angreifer jedoch den Inhalt der übertragenen Nachrichten nicht lesen, aber dennoch „blind“ den Replay-Angriff durchführen und die aufgezeichnete Nachricht erneut absenden.

Die im S-AL getroffene Gegenmaßnahme ist die Integration einer Sequenznummer in die Nachrichten, wodurch Datenaktualität garantiert wird (siehe den Punkt „Datenaktualität” in Abschnitt 5.2.1). Hört ein Angreifer den Datenverkehr mit und sendet eine mitgelesene Nachricht zu einem späteren Zeitpunkt erneut, dann hat diese Nachricht folglich die gleiche Sequenznummer. Die letzte gültige gespeicherte Sequenznummer beim Empfänger für diesen Sender ist deshalb mindestens gleich groß wie die in der Nachricht enthaltene, weshalb die vom Angreifer erneut gesendete Nachricht ignoriert wird. Abbildung 5.4 zeigt ein konkretes Beispiel solch eines Angriffs, bei dem eine Nachricht mit der Sequenznummer 4 mitgelesen und anschließend erneut gesendet wird. Die erneut gesendete Nachricht wird vom Empfänger verworfen, da dieser nach Erhalt der originalen Nachricht die für Sender A gespeicherte Sequenznummer auf den Wert 4 erhöht hat und die erneut gesendete Nachricht ebenfalls diesen Wert als Sequenznummer enthält. Der Angreifer kann die Sequenznummer der mitgelesenen Nachricht allerdings nicht erhöhen, bevor er die Nachricht erneut sendet, weil der in der Nachricht enthaltene MAC auch von der Sequenznummer abhängig ist. Eine Änderung der Sequenznummer würde zu einem Fehler bei der Überprüfung des MACs führen, woraufhin die Nachricht ebenfalls verworfen werden würde.

Um einen Replay-Angriff erfolgreich auszuführen, müsste eine Nachricht gelesen werden, die den Empfänger nicht erreicht. Das kann der Fall sein, wenn der Empfänger zum Zeitpunkt des Sendens der Nachricht nicht mit dem Netzwerk verbunden ist, sei es durch einen Fehler im Netzwerk oder einen Angriff, in dem der Knoten gezielt vom Netzwerk getrennt wird. Wenn nun

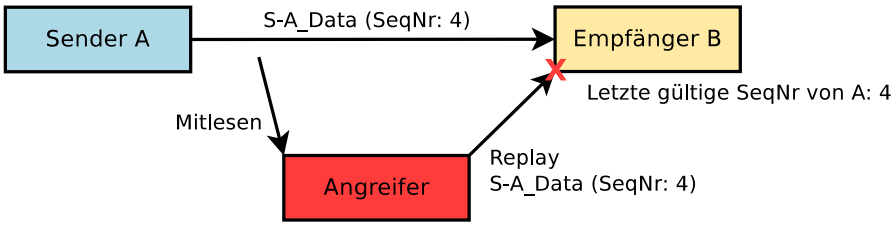

Abbildung 5.4: Beispiel eines Replay-Angriffs

die mitgelesene Nachricht vom Angreifer erneut gesendet wird, bevor der Empfänger wieder eine Nachricht vom Absender der ursprünglichen Nachricht erhält, so ist der Angriff erfolgreich, da die letzte gültige gespeicherte Sequenznummer in der Zwischenzeit nicht erhöht wurde. Abbildung 5.5 zeigt solch ein Beispiel, in dem eine Nachricht mit der Sequenznummer 3 gesendet wird. Die darauf folgende Nachricht mit der Sequenznummer 4 wird vom Angreifer mitgelesen, erreicht ihr Ziel jedoch nicht. Die anschließend vom Angreifer erneut gesendete Nachricht mit der Sequenznummer 4 wird vom Empfänger angenommen, weil die letzte gültige Nachricht von Sender A den Wert 3 als Sequenznummer hatte.

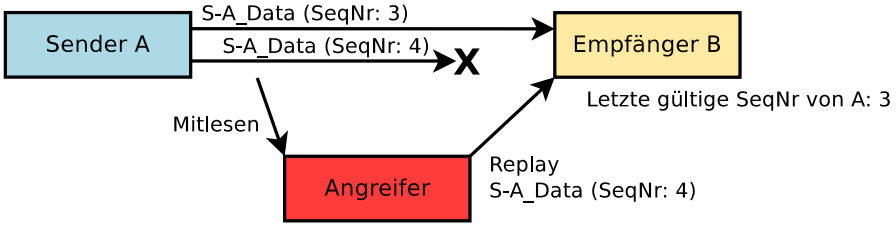

Abbildung 5.5: Replay-Angriff in dem die Originalnachricht das Ziel nicht erreicht

Diesem Fall könnte man mit einem Zeitstempel in der Nachricht verbunden mit einer maximalen Gültigkeitsdauer entgegenwirken, was jedoch eine Uhrzeit in den sicheren KNX-Geräten und in weiterer Folge, wie in BACnet, eine Synchronisierung der Uhrzeiten erfordern würde. Möglich wäre auch die Synchronisierung der Sequenznummern mit dem Dienst `S-A_Sync`, nachdem ein Gerät nach einer Trennung oder einer Abschaltung wieder mit dem Netzwerk verbunden wird. In der Spezifikation des S-ALs wird eine Synchronisierung nach einer Trennung vom Netzwerk nicht erwähnt und müsste demnach von der Anwendung ausgeführt werden. Weiters ist laut Spezifikation eine Synchronisierung der Sequenznummern zwischen zwei Endgeräten anfangs für Geräte im E-Mode vorhergesehen. Diese Methode würde außerdem nur Fälle beinhalten, in der der Knoten von seiner Trennung vom Netzwerk Kenntnis nimmt.

5.3.3 Senden von Nachrichten mit hoher Sequenznummer

Falsche Nachrichten in das System einzuschleusen, wird durch den MAC verhindert, wodurch ohne geheimen Schlüssel keine Nachricht erzeugt werden kann, die vom Empfänger als authentisch angenommen wird. Falls der Angreifer allerdings durch beispielsweise einen erfolgreichen

physischen Angriff in Besitz des Schlüssels gekommen ist oder es ihm gelingt durch Ausprobieren einen gültigen MAC für ein Telegramm auf einer vom Empfänger akzeptierten sicheren Verbindung zu finden (siehe Abschnitt 5.2.3), dann wird die Nachricht vom Empfänger angenommen und vom S-AL an den P-AL weitergegeben. In weiterer Folge wird beim Empfänger die in der Nachricht enthaltene Sequenznummer in der Security Individual Address Table als letzte gültige Sequenznummer für diesen Absender übernommen. Ist diese Sequenznummer deutlich höher als die letzte gültige Sequenznummer oder der maximal mögliche Wert der Sequenznummer, dann werden alle weiteren sicheren Nachrichten mit der zugehörigen physikalischen Adresse als Absender mit niedrigerer Sequenznummer nicht mehr akzeptiert. Abbildung 5.6 zeigt einen derartigen Fall, in dem Sender A eine Nachricht mit der Sequenznummer 4 gesendet wird. Der Angreifer, der im Besitz des geheimen Schlüssels ist oder auf andere Art und Weise erreichen kann, dass seine Nachricht als authentisch akzeptiert wird und sich als Sender A ausgibt, sendet eine Nachricht mit einer deutlich höheren Sequenznummer als die der letzten Nachricht an diesen Empfänger. In Folge dessen wird diese Sequenznummer vom Empfänger B als letzte gültige Sequenznummer für Sender A übernommen und gespeichert. Anschließend gesendete Nachrichten von Sender A werden daraufhin vom Empfänger B nicht mehr akzeptiert, da seine letzte gültige gespeicherte Sequenznummer für Sender A ein höherer Wert ist. In der Datensicherheit in KNX wird eine hohe Steigerung bei der Sequenznummer nicht weiter überprüft. Die Sequenznummer wird angenommen, solange ihr Wert höher als der letzte gültige Wert ist.

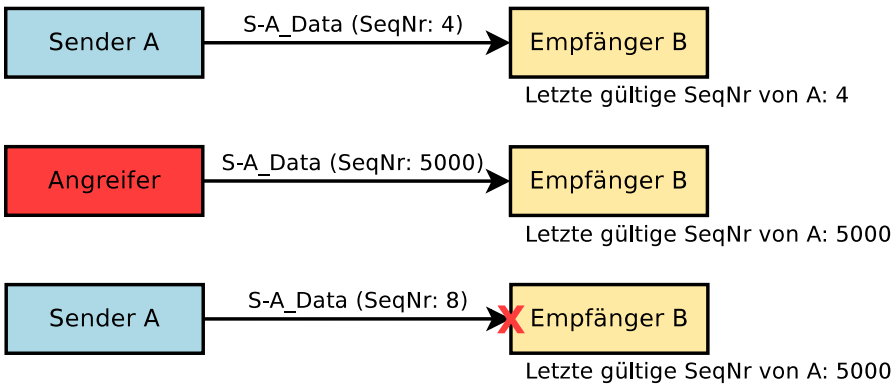

Abbildung 5.6: Nachricht mit hoher Sequenznummer

Ist eine empfangene Sequenznummer um einen hohen Wert größer als die letzte gespeicherte Sequenznummer dieses Senders, dann wäre es möglich, eine Überprüfung der Sequenznummer durchzuführen, bevor dieser Wert übernommen wird. Mit dem Dienst `S-A_Sync` kann die Sequenznummer erfragt werden, die ein entfernter Kommunikationsteilnehmer als nächstes verwenden wird. Bei einem hohen Anstieg der Sequenznummer kann der Empfänger beim Sender als Verifizierung somit dessen Sequenznummer abfragen und einen vorangegangen Angriff identifizieren. Dieser Vorgang müsste jedoch von der Anwendung ausgeführt werden und die genaue Definition des erlaubten Anstiegs der Sequenznummer wäre ebenfalls von der Anwendung abhängig. Die Synchronisierung der Sequenznummer zwischen zwei Endgeräten mit dem

Dienst S-A_Sync ist laut Spezifikation anfangs allerdings erst für E-Mode-Geräte vorgesehen.

5.3.4 Denial of Service (DoS)

Die Absicht von DoS-Angriffen ist die Verfügbarkeit von Diensten oder Daten zu stören [22]. Diese Art von Angriffen kann nicht durch kryptographische Methoden unterbunden werden und somit nicht durch den S-AL selbst abgewendet werden. Es sind dazu zusätzliche physische Maßnahmen nötig.

Um die Kommunikation in einem Netzwerk zu unterbrechen, versucht der Angreifer Überlastungen herbeizuführen. DoS-Angriffe sind auch für Automationssysteme eine ernsthafte Bedrohung [22]. Besonders in eingebetteten Systemen sind diese Angriffe nur schwer zu handhaben, da den Angreifern deutlich mehr Rechenkapazitäten zur Verfügung stehen als das bei Geräten wie Mikrocontrollern der Fall ist. Bei verteilten DoS-Angriffen werden die Kapazitäten der Angreifer noch einmal erhöht, indem sie gemeinsam ein System attackieren und somit noch mehr Ressourcen eingesetzt werden können.

DoS-Angriffe können in zwei Kategorien eingeteilt werden:

- Angriffe, die auf einen Knoten abzielen. Dabei wird versucht, den Knoten durch Überlastung seiner Systemressourcen funktionsuntüchtig zu machen. Typischerweise werden hierbei Knoten angegriffen, die für den Systemerhalt bedeutend sind, wie Firewalls oder Schlüsselserver, Geräte mit Automationsfunktionen, wie Regler oder Operator-Workstations sowie Router.

- Angriffe, die das Netzwerk direkt betreffen. Das Ziel des Angreifers hierbei ist die Bandbreite des Netzwerks derart zu überlasten, dass die Aufrechterhaltung der Kommunikation im betroffenen Netzwerksegment nicht mehr möglich ist. Ein typischer DoS-Angriff in IP-Netzwerken ist das SYN-Flooding, wobei der Verbindungsaufbau des TCP-Protokolls verwendet wird, um Ressourcen am Ziel des Angriffs zu belegen.

In KNX eignen sich die verbindungsorientierten Punkt-zu-Punkt-Kommunikationen, wie zum Beispiel der Dienst A_Memory_Read, zum Lesen eines Speicherbereichs für einen DoS-Angriff (siehe Abbildung 5.7). Es werden bei verbindungsorientierten Kommunikationsbeziehungen keine parallelen Verbindungen unterstützt [23], die Zustandsmaschine wurde nur für jeweils eine Verbindung gestaltet [32]. Werden allerdings sichere KNX-Verbindungen eingesetzt, ist ein Verbindungsaufbau nur mit Kenntnis des Schlüssels möglich. Ein DoS-Angriff bei sicherer KNX-Kommunikation kann durchgeführt werden, indem sichere Nachrichten gesendet werden, deren MAC beim Empfänger auf die Korrektheit hin überprüft wird. Unabhängig vom Resultat dieser Verifizierung werden von den (an Ressourcen armen) Automationsgeräten dafür Rechenkapazität und Zeit benötigt. Vom Angreifer muss dazu in jedem Fall Kenntnis über die sicheren Kommunikationsbeziehungen des Ziels bestehen, da Nachrichten, deren Absender nicht in der Security Individual Address Table des Empfängers vorhanden ist, ignoriert werden. Es bietet sich an, dazu S-A_Sync_Req-Nachrichten zu senden (siehe Abbildung 5.8), welche die Aufgaben haben, die Sequenznummern zu synchronisieren. Diese Nachricht selbst enthält keine Sequenznummer, die vom Empfänger überprüft wird, wodurch in jedem Fall eine Verifizierung

des MACs stattfindet. Dieser MAC kann für einen DoS-Angriff ein frei erfundener Wert sein, die Nachricht muss die Überprüfung des MACs nicht bestehen. Wird für einen DoS-Angriff in einem Netzwerk mit sicheren KNX-Verbindungen der Dienst `S-A_Data` benutzt, dann müssen die gesendeten Nachrichten eine gültige Sequenznummer aufweisen, damit diese soweit akzeptiert werden, dass eine Überprüfung des MACs stattfindet. Der MAC kann auch hier wieder ein beliebiger Wert sein, da die gesendeten Nachrichten von den Knoten nicht angenommen werden müssen.

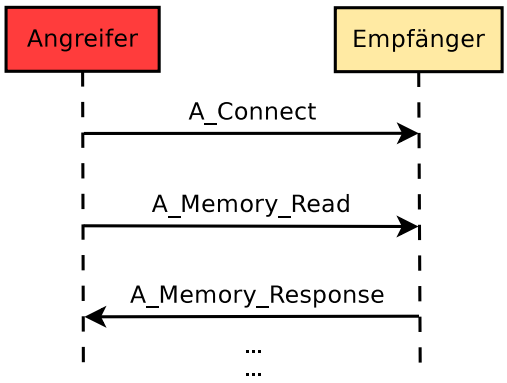

Abbildung 5.7: DoS-Angriff mit verbindungsorientierter Kommunikationsbeziehung in KNX

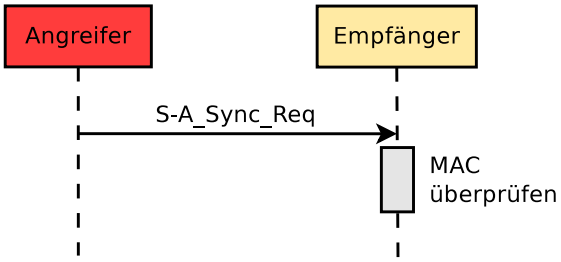

Abbildung 5.8: DoS-Angriff bei sicherer KNX-Kommunikation

DoS-Angriffe können auf zwei Arten gehandhabt werden, (1) durch DoS-Prävention und (2) durch DoS-Erkennung. Ziel der DoS-Prävention ist es, den Zugang zu den Systemressourcen derart zu beschränken, dass ein Angreifer keine Möglichkeit hat, einen erfolgreichen DoS-Angriff durchzuführen. Eine Möglichkeit das zu erreichen, ist durch die physische Sicherheit der Geräte, indem der Zugang zum Übertragungsmedium sowie zu den Geräten, die mit einer Schnittstelle dazu ausgestattet sind, limitiert wird. Diese Maßnahme ist jedoch bei offenen drahtlosen Netzwerken nicht umzusetzen. Wenn die Methoden zur Prävention nicht durchführbar sind, dann sollen DoS-Angriffe zumindest erkannt werden können. Dazu wird versucht, ungewöhnliches Systemverhalten zu erkennen, um in weiterer Folge Gegenmaßnahmen ergreifen zu können.

Ein Konzept um DoS-Angriffe zu handhaben, ist eine Kombination aus Prävention und Er-

kennung. Systemkritische Geräte, wie Operator-Workstations, Regler und Gateways, werden durch einen Mechanismus zur Prävention geschützt. Die in [22] vorgestellte DoS-Prävention besteht aus einem Puzzle, das ein Client, der eine Verbindung aufbauen möchte, zuerst lösen muss. Ziel dieses Puzzles ist es, einen DoS-Angriff für den Angreifer genauso rechenintensiv zu gestalten wie für das Ziel des Angriffs. Bei Netzwerken und Geräten, in denen zu wenige Ressourcen für die Prävention von DoS-Angriffen zur Verfügung stehen, wie zum Beispiel bei eingebetteten Geräten in der Feldebene, sollen die Angriffe zumindest erkannt werden können. Dazu wird ein Intrusion Detection System (IDS) verwendet, das ungewöhnliches Verhalten eines Gerätes oder ungewöhnlichen Datenverkehr im Netzwerk erkennt. Ein IDS besteht aus vier Komponenten [35], jeweils eine zum Sammeln der Daten, zur Datenverarbeitung und zur Speicherung der Daten sowie eine Komponente, die für das Einleiten der Aktion nach der Erkennung einer ungewöhnlichen Aktivität zuständig ist.

Abbildung 5.9 zeigt anhand der 2-Schichtenarchitektur der Gebäudeautomation die Erkennung und Prävention von DoS-Angriffen. Die Operator-Workstation und das Gateway zu einem entfernten Netzwerk, wie dem Internet, verwenden einen Mechanismus zur DoS-Prävention, während beim Feldbus beziehungsweise beim drahtlosen Sensornetz, in denen keine Prävention möglich sind, DoS-Erkennung in den Kopplern zum darüber liegenden Backbone eingesetzt wird.

Wurde ein DoS-Angriff vom IDS erkannt, müssen die aus dem Angriff resultierenden Konsequenzen auf einem Minimum gehalten werden. Dazu muss die Quelle des Angriffs vom restlichen Netzwerk isoliert werden, sodass sich der Angriff nicht weiter verbreiten kann. Je nach Topologie kann der Angreifer nur durch eine Trennung des kompletten betroffenen Netzwerksegments isoliert werden. In einer drahtgebundenen Stern-Topolgie kann der Angreifer leicht isoliert werden, ohne dass die Kommunikation der anderen Knoten beeinträchtigt wird. In drahtgebundender Bus-Topologie oder in freier Topologie kann die Isolation nur durch die Trennung des kompletten betroffenen Segments erfolgen, wodurch auch die übrigen Knoten des Segments betroffen sind. Abbildung 5.10 zeigt ein Beispiel, in dem vom kompromittierten Knoten x ausgehend eine Überflutung des Netzwerks mit Nachrichten stattfindet. Das IDS in Gateway 1 erkennt dieses Verhalten im optimalen Fall und isoliert das Netzwerksegment 2, indem keine Nachrichten mehr an das Netzwerksegment 1 weitergeleitet werden. Somit kann das restliche Netzwerk normal weiterarbeiten, allerdings wird das gesamte Netzwerksegment 1 isoliert und nicht nur der für die Störung verantwortliche Knoten x.

Damit nicht ganze Netzwerksegmente getrennt werden müssen, können die einzelnen Segmente in kleinere virtuelle Netzwerksegmente unterteilt werden. Diese Trennung wird durch eine virtuelle Brücke realisiert und ist für die Kommunikationsteilnehmer transparent. Durch die Trennung eines Netzwerksegments in kleinere virtuelle Netzwerksegmente sind bei der Isolation eines einzelnen Knoten nicht mehr unbedingt alle anderen Knoten des Netzwerksegments betroffen. Muss der Knoten x in Abbildung 5.11 isoliert werden, so kann die virtuelle Brücke 1 lediglich das virtuelle Netzwerksegment 2.2 trennen, wodurch nicht auch alle anderen Knoten von der Trennung betroffen sind. Ist jedoch eine Trennung eines Knoten aus dem Netzwerksegment 2.1 erforderlich, dann ist es nach wie vor nötig, das komplette Netzwerksegment 2 zu isolieren.

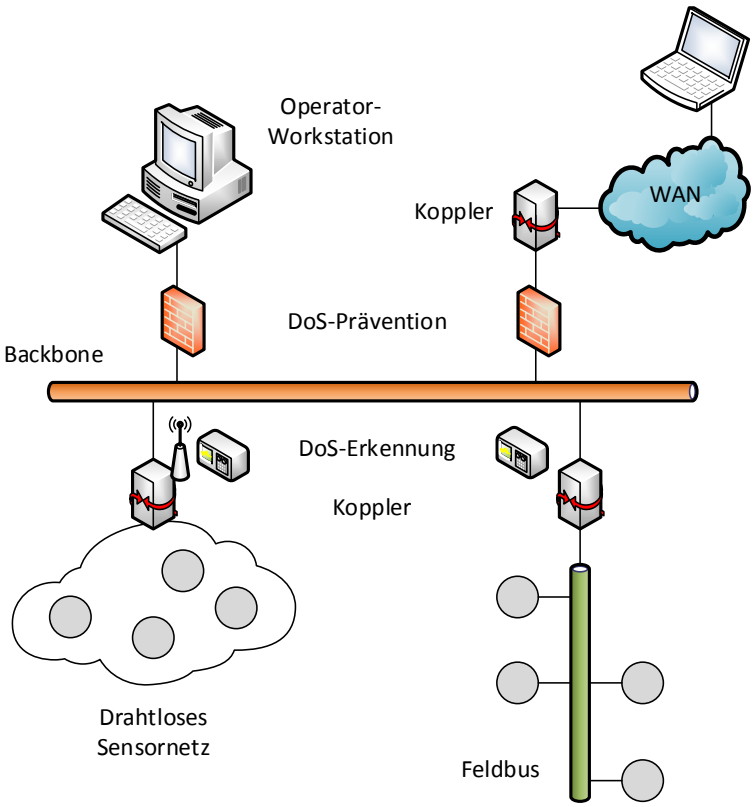

Abbildung 5.9: DoS-Erkennung und Prävention in der Gebäudeautomation [22]

5.3.5 Auswirkungen bei Erlangung von Schlüsseln

Gelingt es einem Angreifer in Besitz eines Schlüssels zu gelangen, hat das je nach Art des Schlüssels unterschiedliche Auswirkungen. Der S-AL erlaubt zwar eine sehr feine Vergabe der Berechtigungen für jeden Dienst und jeden Datenpunkt von jeder einzelnen Verbindung, die Vergabe dieser Berechtigungen ist jedoch ausschließlich vom Schlüssel abhängig. Anhand des Schlüssels wird der Kommunikationspartner authentifiziert und in weiterer Folge die Berechtigungen für diese Verbindung zugewiesen. Da die Autorisierung somit einzig und allein vom Schlüssel abhängig ist, kann ein Angreifer, der in Besitz eines Schlüssels gelangt ist, nicht nur korrekte sichere Nachrichten senden und entschlüsseln, sondern sich auch als dieser Knoten authentifizieren und seine Berechtigungen bei dem oder den (abhängig davon, ob es sich um Gruppenkommunikation handelt) Kommunikationspartnern erhalten.

In Besitz eines Schlüssels kann der Angreifer zum Beispiel durch einen physischen Angriff auf ein Gerät gelangen, worauf im folgenden Punkt 5.4.1 noch genauer eingegangen wird. Falls der Angreifer Zugang zur ETS auf einer Operator-Workstation erlangen kann, dann ist es ihm möglich, dort alle verschiedenen Schlüssel aller Geräte auszulesen.

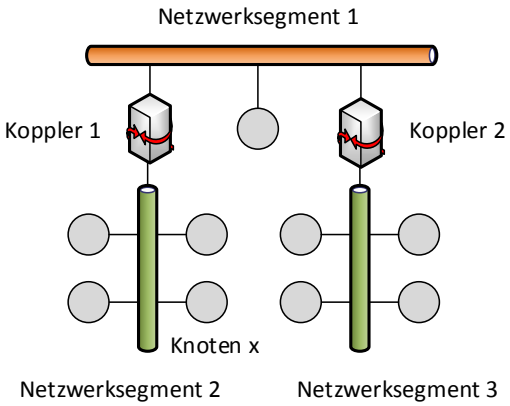

Abbildung 5.10: Isolation eines Gerätes als DoS-Gegenmaßnahme [22]

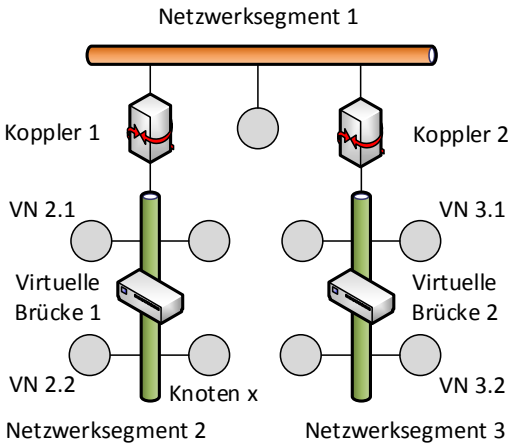

Abbildung 5.11: Isolation eines Gerätes mit virtuellen Brücken als DoS-Gegenmaßnahme [22]

Security Tool Key

Gelangt ein Angreifer in den Besitz des Security Tool Keys, dann kann er sich bei dem Gerät als ETS ausgeben und in weiterer Folge auf alle Eigenschaften des Security Interface Objects zugreifen, für die die Rolle ETS benötigt wird. Der Angreifer wäre in diesem Fall auch dazu berechtigt, Management- und Konfigurationsaufgaben durchzuführen. Das betrifft alle für die Security relevanten Daten, beispielsweise die Security Link Table und die Security Key Table. Dadurch können zum Beispiel sichere Verbindungen gelöscht oder neu konfiguriert werden, sodass sichere Nachrichten von bestimmten Verbindungen nicht mehr angenommen werden.

Die Schlüssel (inklusive Security Tool Key) und Sequenznummern könnten geändert oder der Security-Modus deaktiviert werden.

Die Eigenschaft *Sequence Number Sending* eines Knotens kann mit dem Security Tool Key in der Rolle ETS beschrieben werden und somit auch auf einen niedrigeren Wert als bisher gesetzt werden. Daraus resultierend werden Nachrichten beim Empfänger verworfen werden, weil dieser bereits eine höhere Sequenznummer für diese Absenderadresse gespeichert hat (siehe Abbildung 5.12). Weiters ist es möglich, die Sequence Number Sending auf den maximalen Wert zu setzen, sodass es beim Senden der nächsten Nachricht zu einem Überlauf des Zählers kommt. In weiterer Folge werden keine Nachrichten mehr akzeptiert, da die Sequenznummer niedriger als die beim Empfänger gespeicherte maximale Sequenznummer ist (siehe Abbildung 5.13).

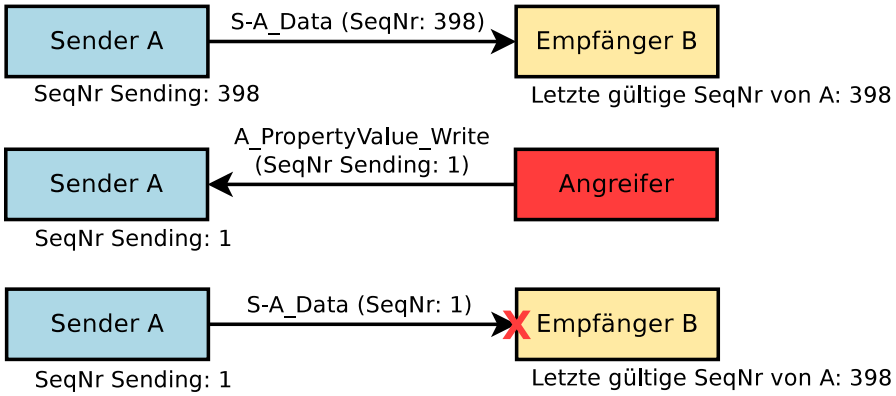

Abbildung 5.12: Erniedrigung der Sequenznummer eines Senders

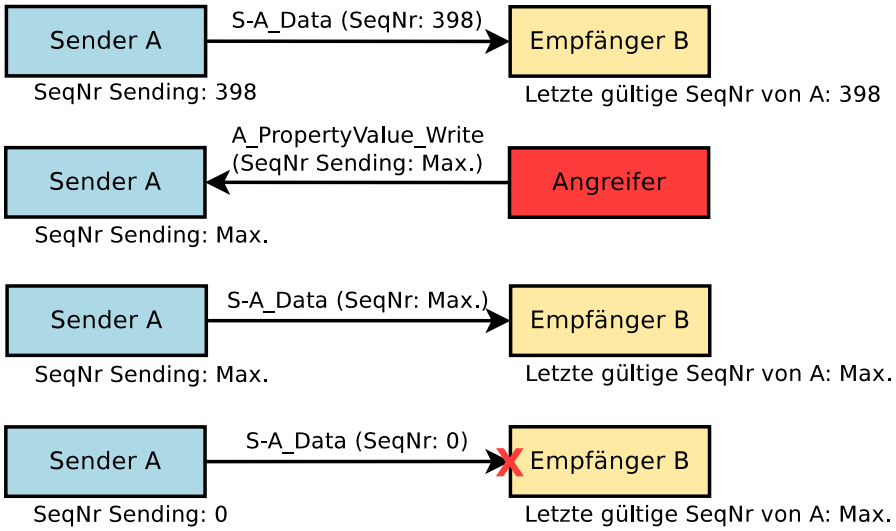

Abbildung 5.13: Erhöhung der Sequenznummer eines Senders auf den Maximalwert

Die Sequenznummer hat eine Länge von 48 Bits, sodass sie bei maximaler Durchsatzra-

te erst nach 100 000 Jahren überläuft. Im Gegensatz zu den Zählern SFCC und SFCL ist ein Überlauf jedoch möglich. Für SFCC sowie SFCL ist in der Spezifikation festgelegt, dass ein Überlauf dieser Werte nicht stattfindet. Sie sollen auf ihren Maximalwerten verbleiben. Durch eine Manipulation der Sequence Number Sending ist es möglich, dass es zu einem Überlauf kommt. Dadurch werden, wie gezeigt, die Nachrichten beim Empfänger verworfen. Weiters werden nach einem Überlauf Sequenznummern, die auch Teil des Initialisierungsvektors des CCM-Algorithmus sind, erneut verwendet, sodass die Datenvertraulichkeit von AES gefährdet ist, wie in Punkt 3.1.2 gezeigt wurde. Eine Gegenmaßnahme wäre, den Überlauf der Sequenznummer zu unterbinden, wie bei SFCC und SFCL.

Weiters können auch die Sequenznummern der Empfänger in der Eigenschaft *Security Individual Address Table* manipuliert werden. Wenn diese Sequenznummern auf einen hohen oder den maximalen Wert gesetzt werden, dann werden von diesem Empfänger die Nachrichten aller betroffenen Kommunikationspartner ebenfalls verworfen, weil ihre Sequenznummer niedriger ist als die gespeicherte manipulierte Sequenznummer der Security Individual Address Table.

Ein Angreifer in Besitz des Security Tool Keys eines Knotens ist außerdem im Stande, die sichere Datenkommunikation der ETS mit diesem Knoten mitzulesen. Dadurch ist es möglich, neue Schlüssel, die von der ETS an den Knoten zur Aktualisierung übertragen werden, zu erlangen. In weiterer Folge kann dadurch auch die Kommunikation mitgelesen werden, die mit den aktualisierten Schlüsseln gesichert ist. Kann ein Angreifer einen Security Tool Key erlangen, hat das somit verheerende Folgen, da alle Konfigurationsaufgaben damit erledigt werden.

Schlüssel der Security Key Table

Mit dem Wissen über einen oder mehrere Schlüssel der Security Key Table können vom Angreifer die Kommunikationen mitgelesen werden, die mit den jeweiligen Schlüsseln gesichert sind. Weiters können mit den Schlüsseln sichere Nachrichten erzeugt werden und ein Angreifer kann sich als einer dieser Knoten authentifizieren. Gelangt ein Angreifer in den Besitz eines Schlüssels, der zur Gruppenkommunikation eingesetzt wird, bedeutet das nicht zwangsläufig, dass damit alle Nachrichten von jedem Gruppenmitglied entschlüsselt werden können. Außerdem kann sich ein Angreifer damit nicht in allen Fällen bei jedem Knoten der Gruppe als ein beliebiger anderer Knoten dieser Gruppe ausgeben. Der S-AL ermöglicht es, für Sende- und Empfangsrichtung sowie für jedes einzelne Gruppenobjekt separate Schlüssel zur Gruppenkommunikation zu verwenden. In Folge dessen kann sich ein Angreifer nicht zwingend als ein anderer Knoten der Gruppenkommunikation ausgeben, wenn er nur in Besitz von einem der Schlüssel dieser Verbindungsoptionen ist, falls für jede Verbindungsrichtung und eventuell auch noch verschiedene Gruppenobjekte ein eigener Schlüssel konfiguriert wurde.

Mit dem Dienst `S-A_ASync` kann ein Angreifer, der in Besitz eines Schlüssels gekommen ist, die Sequenznummer eines Knotens erfragen, mit dem eine sichere Kommunikationsbeziehung besteht. Das ermöglicht es dem Angreifer in weiterer Folge sichere Nachrichten mit korrekter Sequenznummer an diesen Knoten zu senden.

5.3.6 Vergleich

Mit dem S-AL wird in KNX ein umfassendes Security-Konzept eingeführt, wodurch das Protokoll auch für sicherheitskritische Anwendungen eingesetzt werden kann. Mit AES mit einer Schlüssellänge von 128 Bits wird ein aktueller Verschlüsselungsstandard benutzt, der eine gute Performance aufweist und sich dadurch besonders für eingebettete Systeme eignet. Als Betriebsmodus für AES wird, wie in ZigBee, CCM verwendet, wodurch Authentifizierung sowie Vertraulichkeit ermöglicht wird. CCM wurde für KNX leicht abgeändert, was jedoch, wie in Punkt 5.2.2 gezeigt wurde, keinen Einfluss auf die Sicherheit des Betriebsmodus haben sollte. Im Gegensatz zu ZigBee und BACnet hat der MAC zur Authentifizierung und zur Sicherung der Datenintegrität in KNX lediglich eine Länge von 32 Bits, was bei den in KNX vorwiegend benutzten Übertragungsmedien dennoch ausreichenden Schutz vor fälschlicher Authentifizierung von Telegrammen bietet, wie in Abschnitt 5.2.3 gezeigt wurde. Die Möglichkeiten zur Autorisierung sind beim S-AL wiederum sehr vielfältig ausgefallen. Es können für jeden Datenpunkt und jeden Dienst pro Verbindung individuelle Berechtigungen vergeben werden, während in BACnet die Autorisierung von der Zuordnung der Schlüssel zu den Geräten abhängig ist und von der Anwendung ausgeführt werden muss. In ZigBee ist die Autorisierung lediglich pro Gerät möglich, weil verschiedene geheime Schlüssel für unterschiedliche Anwendungen auf einem Gerät nicht möglich sind. Die Datenursprungsauthentifizierung ist abhängig von den gemeinsamen Schlüsseln und kann bei allen Protokollen auf Grund der benutzten symmetrischen Verfahren nur garantiert werden, wenn ein Schlüssel lediglich von zwei Geräten benutzt wird. Bei der Datenaktualität hat BACnet gegenüber ZigBee und dem KNX S-AL den Vorteil, dass die Nachrichten in BACnet einen Zeitstempel enthalten, der es erlaubt, die Datenaktualität nicht nur auf Grund der Reihenfolge der Nachrichten zu bewerten. Die Zeitinformation ermöglicht es, Nachrichten eine bestimmte Lebensdauer zuzuschreiben. Die Datenverfügbarkeit kann, wie bereits in den Kapiteln 3.2.4 und 5.3.4 geschildert, nicht vom Protokoll selbst ohne physische Maßnahmen in Form von Komponenten zur Erkennung von ungewöhnlichem Datenverkehr sowie Gateways zur Trennung von Netzwerksegmenten sichergestellt werden. Tabelle 5.8 zeigt abschließend die Security-Konzepte von BACnet, ZigBee und KNX S-AL im direkten Vergleich.

5.4 Diskussion der Spezifikation

In diesem Abschnitt wird die Spezifikation des KNX S-ALs [31] auf ihre Klarheit, Vollständigkeit und auf Fehler hin untersucht. Fehlerfreiheit und Unmissverständlichkeit einer Spezifikation sind ein wichtiges Kriterium für die Sicherheit, da sie als Grundlage für die Implementierungen des S-ALs dient.

5.4.1 Verschlüsselte Speicherbereiche

Wie in Abschnitt 2.2.1 erläutert, können die Geräte selbst durch physische Maßnahmen angegriffen werden. Über eine Schnittstelle, wie JTAG, die zum Test, zur Fehlersuche und zur Programmierung dient, kann der S-AL umgangen werden, indem direkt auf den Speicher des Gerätes zugegriffen wird. Die Security-Maßnahmen des S-ALs zur sicheren Datenübertragung sind in solch einem Fall wirkungslos. Besonders bei drahtlosen Geräten, die möglicherweise

	BACnet	ZigBee	KNX S-AL
Authentifizierung und Datenintegrität	+ Geräte-ID und 128 Bit HMAC mit MD5 oder SHA-256	+ CCM* mit 32/64/128 Bit MAC	+ CCM mit 32 Bit MAC
Autorisierung	∼ von der Anwendung ausgeführt	∼ pro Gerät, nicht pro Anwendung	+ für alle Daten und Dienste möglich
Datenursprungs-authentifizierung	∼ nur wenn ein Schlüssel auf zwei Geräte beschränkt ist	∼ nur wenn ein Schlüssel auf zwei Geräte beschränkt ist	∼ nur wenn ein Schlüssel auf zwei Geräte beschränkt ist
Datenaktualität	+ ID und Zeitstempel	+ Frame Counter	+ Sequenznummer
Daten-vertraulichkeit	+ AES-128	+ AES-128	+ AES-128
Datenverfügbarkeit	–	–	–

Tabelle 5.8: KNX S-AL Sicherheitsmechanismen im Vergleich

abseits im Gebäude oder im Freien verteilt sein können, stellt das eine Bedrohung dar, weil der direkte Zugriff auf den Speicher bei unbeaufsichtigten Geräten nur schwer verhindert und unbemerkt durchgeführt werden kann. Es kann jedoch unterbunden werden, dass der Angreifer auf diesem Weg an sensible Daten des Security Interface Objects, insbesondere die verschiedenen gespeicherten Schlüssel, die Informationen der sicheren Verbindungen in der Security Link Table oder die Sequenznummern in der Security Individual Address Table gelangt. Diese Daten müssen in einem verschlüsselten Speicherbereich des Gerätes abgelegt werden, um eine Verwendung der für die Security relevanten Daten auch bei erfolgtem Auslesen des Speichers nicht zu ermöglichen. Diese Maßnahme ist allerdings von der Implementierung abhängig und ist bei den jeweiligen Eigenschaften des Security Interface Objects in der Spezifikation nicht zwingend vorgesehen, sondern wird lediglich empfohlen. Wird dieser Empfehlung nicht nachgekommen und diese Daten werden nicht in verschlüsselten Speicherbereichen abgelegt, schwächt das die Sicherheit des S-ALs bedeutend, da dieser umgangen werden kann. Das verschlüsselte Speichern der Werte sollte aus diesen Gründen in der Spezifikation zwingend vorgeschrieben werden und nicht nur als optionale Maßnahme empfohlen werden.

5.4.2 Fehlende Testvektoren für CCM

Wie in Punkt 5.2.2 beschrieben, wird im S-AL eine leicht abgeänderte Variante von CCM mit spezifischen Werten verwendet. In Folge dessen verändern sich im Vergleich mit der ursprünglichen Version von CCM auch die Geheimtexte bei der Verschlüsselung. Die Testvektoren für CCM mit AES [11,61] sind für die abgeänderte Variante von CCM, die in KNX verwendet wird, nicht mehr gültig. Im Gegensatz zur modifizierten Version von CCM in ZigBee werden in der

Spezifikation der Datensicherheit in KNX allerdings keine eigenen Testvektoren für die Variante von CCM in KNX zur Verfügung gestellt.

Durch die fehlenden Testvektoren kann bei der Implementierung des S-ALs die abgeänderte Variante von CCM nicht auf die korrekte Umsetzung hin überprüft werden. Das stellt eine Gefährdung für die Korrektheit der Endprodukte dar, weil eine einfache Möglichkeit zum Testen des Algorithmus fehlt. Eine fehlerhafte Umsetzung von CCM hätte zur Folge, dass die Geräte nicht mit anderen kommunizieren können, weil empfangene Nachrichten bei der Verifizierung verworfen werden würden, oder dass Klartexte nicht korrekt zu Geheimtexten verschlüsselt werden und vice versa.

5.4.3 Widersprüche und Unklarheiten

Dieser Abschnitt zeigt Widersprüche sowie Unklarheiten auf, die in der Spezifikation vorkommen, was in weiterer Folge dazu führen kann, dass der KNX S-AL nicht korrekt implementiert wird.

Schlüsselaustausch

Für den Security Tool Key ist in der Spezifikation der Datensicherheit in KNX beschrieben, wie die Änderung eines bereits konfigurierten Schlüssels abzulaufen hat. Mit der ETS können die Werte des Security Interface Objects gelesen und geschrieben werden, was auch auf den Security Tool Key zutrifft. In Punkt 2.3.2.10.4 der Spezifikation ist für den Security Tool Key festgelegt, dass der Management Server (MaS) (das Gerät selbst) nach der Konfiguration des Schlüssels die `A_PropertyValue_Write`-PDU mit der `A_PropertyValue_Response`-PDU bestätigt und dazu den neu geschriebenen Security Tool Key benutzt und nicht mehr den FDSK: *„The A_PropertyValue_Response-PDU to confirm the A_PropertyValue_Write-PDU shall immediately be protected using the newly written Security Tool Key.“* [31] Im darauf folgenden Punkt 2.3.2.10.5 der Spezifikation wird jedoch geschrieben, dass der MaS bei der Zuweisung oder einer Änderung des Security Tool Keys die Modifizierung mit dem vorhergehenden Security Tool Key bestätigen soll: *„The MaS shall properly handle the change or the activation of the Security Tool Key: if the Security Tool Key is assigned (after the FDSK is used) or modified (after another Security Tool Key value is used), then the MaS will confirm that modification with the preceding Security Tool Key, but from then on only accept the newly Configured Tool Key.“* [31] Dabei handelt es sich einerseits um einen Widerspruch zu der in Punkt 2.3.2.10.4 gemachten Aussage, dass für die Bestätigung der Zuweisung des Security Tool Keys mit `A_PropertyValue_Response` bereits das neu geschriebene Security Tool Key benutzt werden soll, und andererseits ist es nicht möglich, nach einer Änderung des Security Tool Keys den vorhergehenden Wert zu benutzen, da die Eigenschaft des Security Tool Keys in einem einzelnen Element gespeichert ist und der Wert bei einer Änderung überschrieben wird und keine vorangegangenen Werte des Security Tool Keys gespeichert werden.

SFCC und SFCL

In der Spezifikation kommt es hinsichtlich der Zähler SFCC beziehungsweise SFCL für auftretende Fehler bei der Security zu einem Widerspruch. In Abschnitt 2.2.1.3.1, in der die Handhabung der Sequenznummer behandelt wird, steht, dass der SFCC (und falls implementiert der SFCL) um den Wert eins erhöht werden soll, wenn die Sequenznummer der empfangenen Nachricht niedriger als die letzte gültige Sequenznummer in der Security Individual Address Table ist: *„In case the received Sequence Number is lower than the "Last Valid SeqNr" then the S-AL shall do the following. [...] It shall increment the Security Failure Common Counter by 1. If the Security Failure Counters on Links is implemented, then in the entry corresponding with the link on which the error happens, the S-AL Failure Counter shall also be incremented by 1."* [31] In der Verwendung des SFCLs durch den MaS (das Gerät selbst) in Abschnitt 2.3.2.9.4 steht jedoch ausdrücklich, dass der SFCL für Security-Fehler, die nicht die Überprüfung der Sequenznummer betreffen, erhöht wird: *„The MaS (device) shall increment the SFCL-entry of a Link if a secure message is received with a security failure other than the SeqNr checking."* [31] Der SFCL wird laut Punkt 2.3.2.9.4 für auftretende Fehler bei der Authentifizierung und Vertraulichkeit erhöht. Der SFCC wird laut Abschnitt 2.3.2.17.1 ebenfalls für auftretende Fehler bei der Authentifizierung und Vertraulichkeit erhöht sowie für Fehler von Rollen und Berechtigungen im AIL. Demnach würden die Zähler SFCC und SFCL für auftretende Fehler bei der Sequenznummer nicht erhöht werden.

Role Table

Laut Definition der Role Table in Punkt 2.3.2.15.3 soll jede Verbindung höchstens einmal in der Role Table vorkommen: *„The Role Table shall be an array Property of which each element shall contain the relation between one Link, through its Link Index, and one Role. Every Link shall be at maximum once in the Role Table."* [31] Passend dazu wird auch bei den Anforderungen der Rollen in Abschnitt 2.2.3.2.3 spezifiziert, dass jeder Verbindung über die Role Table genau eine Rolle zugeordnet werden soll: *„In the receiver, each Link shall be associated with exactly one Role through the Role Table [...]."* [31] Im Widerspruch dazu steht später im gleichen Abschnitt wiederum, dass die Role Table jeder Verbindung der Security Link Table eine oder mehrere Rollen zuordnen soll: *„The Role Table shall associate each link in the Security Link Table to one or more Roles."* [31] Bei der Beschreibung der Punkt-zu-Punkt-Kommunikation wird weiters angeführt, dass der Link Index mehrmals in der Role Table erscheinen kann: *„[...] the Link Index may as well appear multiple times in the Role Table. One sender may thus have multiple Roles in the receiver."* [31] Außerdem wird an dieser Stelle geschrieben, dass die Role Table mehr als eine Rolle für einen Link Index vorhersehen kann: *„In case the Role Table foresees more than one Role for a Link Index [...]"* [31] Beim erklärenden Beispiel einer einfachen Role Table in Abbildung 26 ist weiters ein Link Index zwei Mal vertreten.

Security Mode

Laut Spezifikation des Security Modes des Security Interface Objects in Punkt 2.3.2.5 kann auf diese Eigenschaft nur unter Verwendung von sicherer Kommunikation mit Authentifizierung und Vertraulichkeit mit der Rolle ETS zugegriffen werden. Punkt 2.6.3.4.5, in dem der

S-Mode-Konfigurationsprozess behandelt wird, besagt jedoch, dass der Security Mode mittels ungesicherter Kommunikation aktiviert wird: *„Enable the Security Mode in the MaS: This shall be done using unsecure communication.“* [31]

Dienst `S-A_Sync`

Bei der Spezifikation des Dienstes `S-A_Sync` in Punkt 2.2.1.3.2 kommt es zu den folgenden Widersprüchen.

Schlüssel Bei der Spezifikation des Parameters `key_select`, zu Beginn der Beschreibung des Dienstes `S-A_Sync`, ist definiert, dass der Tool Key oder der Schlüssel, der in der *Security Individual Address Table* für diesen ASAP gespeichert ist, verwendet wird: *„[...] the key that is stored in the Security Individual Address Table for that ASAP = IA_Index.“* [31] In der Security Individual Address Table sind allerdings keine Schlüssel enthalten, diese befinden sich in der Security Key Table und der Verweis zum Schlüssel, der anhand des ASAP ausfindig gemacht werden kann, ist Teil der Security Link Table.

Wert des SCF Abbildung 14 von Punkt 2.2.1.3.2 der Spezifikation zeigt das Format der `S-A_Sync_Req`-PDU. Dabei wird für das SCF der Wert 26h festgelegt, was jedoch nicht mit der Definition des SCF in Abbildung 6 von Punkt 2.2.1.2.1 übereinstimmt. In einer `S-A_Sync_Req`-Nachricht muss demnach der Wert für das SCF 07h, bei Einsatz eines Schlüssel aus der Security Link Table, beziehungsweise 27h, bei Verwendung des Security Tool Keys, lauten.

Ebenso ist in Abbildung 15 beim Format der `S-A_Sync_Res`-PDU der Wert 2Ah für das SCF spezifiziert. Wie zuvor stimmt dieser Wert nicht mit der Definition des SCF in Abbildung 6 der Spezifikation des S-ALs überein. Das SCF einer `S-A_Sync_Res`-PDU müsste demnach den Wert 0Bh, bei Verwendung eines Schlüssels aus der Security Link Table, beziehungsweise 2Bh, beim Einsatz des Security Tool Keys, haben.

Anhang

Im Gegensatz zur Spezifikation selbst hat der Anhang lediglich informativen Charakter. In diesem Unterabschnitt werden dennoch darin auftretende Unklarheiten angeführt.

CFB in Anhang B.1.2 In Abschnitt B.1.2 des Anhangs wird auf den Betriebsmodus CFB für Blockverschlüsselungen eingegangen und erwähnt, dass die sichere KNX-Kommunikation zur Laufzeit diesen Modus verwenden soll. Der S-AL benutzt allerdings CCM als Betriebsmodus für AES und der Abschnitt des Anhangs kann somit aus dem Dokument entfernt werden.

Größe der Sequenznummer in Anhang B.2 Im Abschnitt B.2 des Anhangs wird die Bestimmung der Länge der Sequenznummer beschrieben. Dafür wird die maximale Anzahl an Nachrichten berechnet, die pro Sekunde über KNX RF versendet werden kann, um herauszufinden, welche Bytelänge das Feld, das die Sequenznummer enthält, aufweisen muss, um einen Überlauf zu vermeiden. Als typische Größe eines KNX-Telegramms werden dabei 23 Bytes

angenommen, wodurch sich eine maximale Anzahl an 89 gesendeten Nachrichten pro Sekunde ergeben. Es wird jedoch nicht beschrieben, wie die typische Größe eines KNX-Telegramms ermittelt wurde. Die Länge eines KNX-RF-Telegramms mit einem Byte an Nutzdaten beträgt die erwähnten 23 Bytes. Die Sequenznummer wird allerdings nur für gesendete sichere KNX-Telegramme erhöht, also nur wenn der S-AL eine `S-A_Data-PDU` an die Transportschicht übergibt (siehe 2.3.2.14.3 der Spezifikation). Deshalb müsste für die Berechnung die Länge eines sicheren KNX-RF-Telegramms verwendet werden und nicht die eines normalen ungesicherten Telegramms.

Ein sicheres KNX-Telegramm im Format von KNX RF, das die für den S-AL relevanten Daten SCF, Sequenznummer und MAC sowie ein Byte an Nutzdaten enthält, hat eine Länge von 36 Bytes (die Struktur eines KNX-RF-Telegramms wird in Abbildung 4.9 in Abschnitt 4.4 gezeigt). Ein Byte an Nutzdaten wurde gewählt, weil dies das kürzest mögliche Telegramm und somit den ungünstigsten Fall darstellt. Ausgehend von einer Länge von 36 Bytes, was 288 Bits entspricht, für ein sicheres KNX-Telegramm ergeben sich eine maximale Anzahl von 56,89 gesendeten sicheren Nachrichten in der Sekunde auf dem Medium KNX RF. Die Berechnung in der Spezifikation müsste somit folgendermaßen aussehen:

$$\frac{16\,384\ Bits/Sekunde}{288\ Bits/Nachricht} = 56,89\ Nachrichten/Sekunde$$

Bei einer Sequenznummer mit einer Länge von 48 Bits, wie im S-AL verwendet, läuft die Sequenznummer damit erst nach 156 890 Jahren über und nicht schon nach 100 286 Jahren, wie bei der Länge von 23 Bytes für ein KNX-Telegramm, so wie in der Spezifikation berechnet. Die 23 Bytes, mit denen die Berechnung in der Spezifikation durchgeführt werden, stellen aber jedenfalls einen ungünstigeren Fall vor.

KAPITEL 6

Prototyp

6.1 Einleitung

Um die Realisierbarkeit des KNX S-ALs zu zeigen, wurde ein Prototyp implementiert. Er erlaubt es, KNX-Nachrichten zu senden sowie sichere Nachrichten zu generieren und zu verifizieren. Diese Implementierung dient auch zur Bestimmung der Performance dieser Vorgänge. Der Prototyp wurde auf einem MSP430-Mikrocontroller von Texas Instruments implementiert. Zur Verschlüsselung und Generierung des MACs der Nachrichten wird das RELIC-Toolkit mit einer Erweiterung um AES-128 und CCM verwendet. Die Anbindung des MSP430 an eine KNX-Installation erfolgt mit dem Siemens TP-UART 2 [53], einem Transceiver, um Mikrocontroller mit KNX zu verbinden.

6.1.1 Texas Instruments MSP430

Der MSP430 ist ein Mikrocontroller von Texas Instruments, der für Anwendungen mit niedrigstem Stromverbrauch optimiert ist. Er besitzt eine 16-Bit-RISC-CPU sowie einen Hardwaremultiplizierer. Die MSP430-Familie umfasst verschiedene Serien mit unterschiedlichen Konfigurationen und Ausstattungen, wobei die Implementierung des Prototypen auf einem MSP430F5438A [57, 58] erfolgte. Dieses Modell besitzt 256 Kilobytes Flash und 16 Kilobytes SRAM an Speicher, drei 16-Bit-Timer und bis zu vier Universal Serial Communication Interfaces (USCI), die verschiedene serielle Kommunikationsmodi (UART, SPI und I2C) in einem Hardwaremodul unterstützen.

6.1.2 Siemens TP-UART 2

Der TP-UART 2 [53] von Siemens dient dazu, um eine physische Verbindung mit einem KNX-Bus herzustellen. Es ist ein Transceiver, der über eine UART-Schnittstelle die Kommunikation mit einem Mikrocontroller erlaubt, und so die Anbindung von Mikrocontroller zu einer KNX-TP-Installation ermöglicht. Dabei entlastet der TP-UART durch seine Funktionalität den Mikro-

controller von Kodierungs- und Dekodierungsaufgaben auf Bitebene. Der TP-UART 2 bezieht seine Energie vom KNX-Bus und stellt zudem eine 3,3 Volt oder 5 Volt sowie eine 20 Volt Stromversorgung zur Verfügung. Der TP-UART 2 besteht aus zwei Hauptteilen, einem digitalen und einem analogen Teil, wobei der analoge Teil für das Senden und Empfangen der Nachrichten am KNX-Bus und der digitale Teil für die Verarbeitung der KNX-Nachrichten und die UART-Funktionalität zuständig ist.

Der Mikrocontroller steuert über eine UART-Schnittstelle die Dienste des TP-UART und übermittelt darüber die zu sendenden Daten beziehungsweise empfängt die KNX-Nachrichten. Der TP-UART ist zuständig für die korrekte Kodierung und Übertragung der Daten sowie den Empfang von Daten vom KNX-Bus und deren Weiterleitung an den Mikrocontroller über die UART-Schnittstelle. Zur Anbindung des MSP430-Mikrocontrollers an den KNX-Bus wurde das TP-UART 2 Evaluation Board [52] verwendet. Dieses Board enthält den TP-UART 2-IC sowie eine Schnittstelle zum KNX-Bus und ein Bus Transceiver Interface (BTI), das die UART-Schnittstelle zum Mikrocontroller mit einer Baudrate von 19.200 sowie eine 5 Volt und eine 20 Volt Versorgungsspannung zur Verfügung stellt. Abbildung 6.1 zeigt die Anbindung eines Mikrocontrollers an den KNX-Bus über den TP-UART 2.

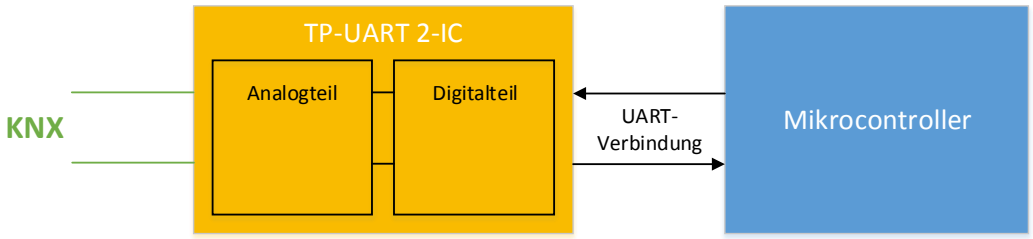

Abbildung 6.1: TP-UART 2 Modul [53]

6.1.3 RELIC-Toolkit und Erweiterung um AES-CCM

Für die Verschlüsselung und Generierung des MACs wird das RELIC-Toolkit mit einer Erweiterung um AES-128 und CCM eingesetzt. RELIC ist eine Bibliothek für Kryptographie, die ihren Schwerpunkt auf Effizienz und Flexibilität legt. Zudem liegt bei RELIC der Fokus auf einfacher Portierung und der Einbindung von Code, der von der Architektur abhängig ist. So ist auch spezifischer Quellcode für den MSP430 enthalten, worunter sich in Assembler programmierte Funktionen befinden. RELIC wird als kryptographische Implementierung von TinyPBC [42] verwendet. Dabei handelt es sich um eine effiziente Implementierung von Pairing-Based Cryptography (PBC) für einen 8-Bit-Prozessor, die kompatibel zu TinyOS ist, einem flexiblen, quelloffenen Betriebssystem für drahtlose Sensornetze. RELIC implementiert Integer-, Primkörper- und Binärfeld-Arithmetik, elliptische Kurven über Primkörpern und Binärfeldern, bilineare Abbildungen sowie kryptographische Protokolle, wie RSA, ECDSA, ECIES, das BLS-Signaturschema oder das homomorphe Benaloh-Kryptosystem.

In [15] wird eine Implementierung von AES-128 und CCM sowie weiteren Betriebsmodi, die Authentifizierung und Vertraulichkeit erlauben, für die MSP430-Mikrocontrollerfamilie präsentiert. Diese Implementierung basiert auf dem RELIC-Toolkit und erweitert dieses. Die Softwareimplementierung von AES wurde modifiziert, um von der 16-Bit-Architektur des MSP430 profitieren zu können. Einige Funktionen wurden auch in Assembler geschrieben. Speziell zugeschnitten wurde die Implementierung auf die MSP430X-CPU. Diese weist im Vergleich zu den herkömmlichen MSP430-CPUs eine erweiterte Speicherarchitektur auf, die bis zu einem Megabyte an Speicher adressieren kann. Sie enthält außerdem einen Adressbus mit 20 Bits Breite sowie 20 Bits breite Register. Die Implementierung unterstützt zudem das AES-Hardwaremodul, das manche Modelle der MSP430-Familie anbieten. Durch diese Hardwarebeschleunigung ist CCM bei der Verschlüsselung zirka zehnmal und bei der Entschlüsselung achtmal schneller als in Software auf der gleichen Plattform [15].

Anpassung von CCM an den KNX S-AL

Wie in Punkt 5.2.2 erläutert, weicht das Format der Blöcke B_0 und Ctr_j des CCM-Algorithmus im S-AL von der ursprünglichen CCM-Spezifikation ab. Für die Änderungen muss jedoch nicht in die Berechnungsoperationen der Authentifizierung beziehungsweise der Verschlüsselung selbst eingegriffen werden, lediglich der Inhalt der einzelnen Bytes der erwähnten Blöcke muss abgeändert werden.

Zu Beginn der Implementierung wurden die Ergebnisse der CCM-Erweiterung von RELIC anhand der CCM-Testvektoren in [11, 61] überprüft. Erst anschließend wurden die Änderungen an den Blöcken B_0 und Ctr_j gemäß der Spezifikation des S-AL [31] vorgenommen, um korrekte Ergebnisse von CCM im KNX S-AL zu erhalten.

6.2 Anmerkungen zur Implementierung

Bei der Umsetzung der Spezifikation des KNX S-ALs [31] kam es zu Problemen und Schwierigkeiten, die in diesem Punkt festgehalten werden.

6.2.1 Inhalt der Nutzdaten

Die Spezifikation des S-ALs legt fest, dass ein sicheres KNX-Telegramm, je nach Security-Modus, die APDU (APCI + Daten) im Klartext beziehungsweise den Geheimtext der APDU enthält (siehe auch Abbildung 5.2). Die APCI hat, je nach Dienst, eine Länge von vier oder zehn Bits, wodurch die APDU eine Bitlänge ungleich eines Vielfachen von acht und somit keine ganzen Bytes ergibt. Weiters sind in der Spezifikation die Eingabewerte für den CCM-Algorithmus festgelegt. Im Falle von nur Authentifizierung setzen sich die zu authentifizierenden Daten A und die Nutzdaten P folgendermaßen zusammen (siehe Punkt 2.2.1.2.2.4.2 der Spezifikation [31]):

A = Secure APCI | SCF | SeqNr | Klartext APDU (= Klartext APCI + Daten)
P = {leer}

Bei zusätzlicher Verschlüsselung bestehen A und P aus:

A = Secure APCI | SCF | SeqNr
P = Klartext APDU (= Klartext APCI + Daten)

Die Secure APCI zu Beginn der Daten A weist eine Länge von zehn Bits auf. Die APCI, als Teil der APDU, hat, je nach Dienst, eine Länge von vier oder zehn Bits. Dadurch würden A und P ohne Auffüllen der restlichen Bits mit Nullen keine ganzen Bytes ergeben, was vom CCM-Algorithmus allerdings gefordert wird. Rücksprache mit der KNX Association ergab, dass zur APCI auch die sechs Bits lange TPCI Teil der Daten von A und P sowie der Nutzdaten im KNX-Telegramm sein soll. Es wird somit nicht die APDU, sondern die TPDU in den Nutzdaten eines sicheren Telegramms übertragen. Dadurch ergibt sich folgende Zusammensetzung von A und P für den Modus nur Authentifizierung:

A = TPCI | Secure APCI | SCF | SeqNr | Klartext TPDU (= Klartext TPCI | APCI + Daten)
P = {leer}

Für Authentifizierung mit Vertraulichkeit sehen die Daten A und P folgendermaßen aus (siehe Punkt 2.2.1.2.2.5.2 der Spezifikation [31]):

A = TPCI | Secure APCI | SCF | SeqNr
P = Klartext TPDU (= Klartext TPCI | APCI + Daten)

Dadurch, dass die TPDU übertragen wird, folgt auch, dass für die eigentlichen Nutzdaten des Endanwenders lediglich ein vollständiges Byte plus gegebenenfalls die sechs zusätzlichen Bits im Falle einer kürzeren APCI übrig bleiben.

6.2.2 Initialisierungsvektor

Wie in den Tabellen 5.2 und 5.4 von Abschnitt 5.2.2 der Arbeit gezeigt, beinhalten die Blöcke B_0 und Ctr_j der CCM-Version in KNX die Quelladresse und die Zieladresse. Weiters enthält B_0 den Adresstyp, den Telegrammtyp und die TPCI. Um den Einsatz von CCM zu ermöglichen, müssen diese Werte bereits im S-AL, der sich innerhalb des ALs befindet, bekannt sein. Da diese Werte in den Schichten unterhalb des ALs beim Senden hinzugefügt beziehungsweise in Empfangsrichtung auf Grund der Abstrahierung weggelassen werden, muss der S-AL Aufgaben dieser Schichten übernehmen, um TPCI, Adresstyp, Telegrammtyp und die Adressen zu ermitteln. Dadurch ist die Abgrenzung der einzelnen Schichten voneinander nicht mehr vollständig gegeben und Operationen werden doppelt ausgeführt, im S-AL und der eigentlichen Schicht.

Die Zieladresse wird anhand des TSAPs ausgelesen, ein Vorgang der eigentlich vom Transport Layer ausgeführt wird. Um die Ver- und Entschlüsselung zu ermöglichen, muss dieser Vorgang auch vom S-AL ausgeführt werden. Der Telegrammtyp kann anhand der Länge der Nutzdaten bestimmt werden, der Adresstyp und die TPCI werden durch den benutzten Dienst spezifiziert. Zum Beispiel ist `A_GroupValue_Write` ein Multicast-Dienst, der auf den Transport-Layer-Dienst `T_Data_Group` abgebildet wird, für den die TPCI 000000b festgelegt ist. In Empfangsrichtung ist die TPCI Teil der Nutzdaten (die, wie weiter oben beschrieben, aus der TPDU bestehen sollen).

6.3 Sicheres KNX-Telegramm

In diesem Punkt werden ein herkömmliches KNX-Telegramm sowie Telegramme unter Verwendung des S-ALs anhand eines konkreten Beispiels gezeigt. Die gezeigten Telegramme beziehen sich auf das Telegrammformat des Übertragungsmediums TP1. Bei dem benutzten Dienst handelt es sich um den Multicast-Dienst `A_GroupValue_Write`, der mit dem Transport-Layer-Dienst `T_Data_Group` übertragen wird. `T_Data_Group` ist durch die TPCI 000000b und `A_GroupValue_Write` durch die APCI 0010b gekennzeichnet.

Bei den Nutzdaten des KNX-Telegramms handelt es sich um den Datenpunkttyp „*Room Heating Controller Status*“, der Statusinformationen der Heizung überträgt [32]. Der übertragene Wert lautet 72h, was bedeutet, dass sich die Heizung im Energiesparmodus befindet, der sogenannte „Morning Boost“ sowie die Start- und Stop-Optimierung aktiviert sind. Da für diesen Datenpunkttyp ein ganzes Byte als Länge spezifiziert ist, werden die sechs freien Datenbits des Feldes APCI nicht benutzt. Die Nutzdaten beginnen im darauf folgenden Byte. Tabelle 6.1 zeigt zum Vergleich mit den folgenden sicheren KNX-Telegrammen ein vollständiges Telegramm auf dem Übertragungsmedium TP1 ohne Einsatz des S-ALs.

Byte 0	Byte 1	Byte 2	Byte 3	Byte 4	Byte 5	Byte 6	Byte 7
Kontrollfeld	Quelladresse		Zieladresse		Kontrollfeld	TPCI | APCI	APCI | Daten
BCh	11h	05h	00h	02h	E2h	00h	80h
10111100b	00010001b	00000101b	00000000b	00000010b	11100010b	00000000b	10000000b

Byte 8	Byte 9
Daten	Prüfsumme
72h	45h
01110010b	01000101b

Tabelle 6.1: KNX-TP1-Standardtelegramm ohne Sicherheitsmechanismen

Die Tabellen 6.2 und 6.3 zeigen jeweils ein vollständiges KNX-Telegramm auf dem Übertragungsmedium TP1 unter Einsatz der Sicherheitsmechanismen des S-ALs. Die Telegramme weisen die maximale Länge eines TP1-Standardtelegramms auf. Die gelb hinterlegten Bytes stellen spezifischen Klartext des Dienstes `S-A_Data` dar. Die blau hinterlegten Bytes enthalten verschlüsselte Daten. Als Schlüssel kam dabei jeweils der Schlüssel der Testvektoren in [61], C0C1C2C3C4C5C6C7C8C9CACBCCCDCECFh, zum Einsatz. Tabelle 6.2 zeigt das KNX-Telegramm, das im Security-Modus mit Authentifizierung erstellt wurde. Die Nutzdaten sind unverschlüsselt, es ist jedoch die Sequenznummer sowie ein MAC enthalten. Der MAC wird über die TPCI, die Secure APCI, das SCF, die Sequenznummer sowie über die darauf folgenden Nutzdaten generiert, wodurch auch diese vor Manipulationen geschützt sind.

Das KNX-Telegramm in Tabelle 6.3 wurde im Modus Authentifizierung mit Vertraulichkeit generiert. Im Gegensatz zum vorhergehenden Telegramm sind die Nutzdaten in den Bytes 15 bis inklusive 17 verschlüsselt.

Byte 0	Byte 1	Byte 2	Byte 3	Byte 4	Byte 5	Byte 6	Byte 7
Kontrollfeld	Quelladresse		Zieladresse		Kontrollfeld	TPCI | APCI	Secure APCI
BCh	11h	05h	00h	02h	EFh	03h	F1h
10111100b	00010001b	00000101b	00000000b	00000010b	11101111b	00000011b	11110001b

Byte 8	Byte 9	Byte 10	Byte 11	Byte 12	Byte 13	Byte 14	Byte 15
SCF	Sequenznummer						TPCI | APCI
02h	00h	00h	00h	00h	D7h	64h	00h
00000010b	00000000b	00000000b	00000000b	00000000b	11010111b	01100100b	00000000b

Byte 16	Byte 17	Byte 18	Byte 19	Byte 20	Byte 21	Byte 22
APCI | Daten	Daten	MAC				Prüfsumme
80h	72h	5Dh	51h	C0h	D6h	11h
10000000b	01110010b	01011101b	01010001b	11000000b	11010110b	00010001b

Tabelle 6.2: KNX-TP1-Standardtelegramm mit Authentifizierung

Byte 0	Byte 1	Byte 2	Byte 3	Byte 4	Byte 5	Byte 6	Byte 7
Kontrollfeld	Quelladresse		Zieladresse		Kontrollfeld	TPCI | APCI	Secure APCI
BCh	11h	05h	00h	02h	EFh	03h	F1h
10111100b	00010001b	00000101b	00000000b	00000010b	11101111b	000000 | 11b	11110001b

Byte 8	Byte 9	Byte 10	Byte 11	Byte 12	Byte 13	Byte 14	Byte 15
SCF	Sequenznummer						TPCI | APCI
03h	00h	00h	00h	00h	D7h	64h	1Bh
00000011b	00000000b	00000000b	00000000b	00000000b	11010111b	01100100b	00011011b

Byte 16	Byte 17	Byte 18	Byte 19	Byte 20	Byte 21	Byte 22
APCI | Daten	Daten	MAC				Prüfsumme
8Dh	C4h	13h	67h	D9h	31h	36h
10001101b	11000100b	00010011b	01100111b	11011001b	00110001b	00110110b

Tabelle 6.3: KNX-TP1-Standardtelegramm mit Authentifizierung und Vertraulichkeit

6.4 Performance

Anhand der Implementierung des Prototyps werden die Performance der Funktionen untersucht. Zur Messung der Anzahl der benötigten Taktzyklen wurde der Cycle Counter des MSP430F5438A benutzt, der mit dem Debugger ausgelesen werden kann.

Abbildung 6.2 zeigt die Anzahl der Taktzyklen, die von der CCM-Implementierung für die Sicherheitsmodi nur Authentifizierung sowie Authentifizierung mit Vertraulichkeit benötigt werden. Es wurden Nutzdaten mit einer Länge von einem bis 242 Bytes authentifiziert beziehungsweise authentifiziert und verschlüsselt. Zusätzlich zu den Nutzdaten werden immer jeweils neun Bytes authentifiziert (TPCI | Secure APCI | SCF | SeqNr). Eine Nutzdatenlänge von 242 Bytes

entspricht der maximalen Länge der Nutzdaten in einem sicheren KNX-TP1-Telegramm, wobei die Nutzdaten die TPDU darstellen (siehe Punkt 6.2).

Die in dem Diagramm zu sehenden Sprünge der benötigten Taktzyklen entstehen jeweils bei Erreichen der Blocklänge der AES-Verschlüsselung. Hier wird die Blockverschlüsselung aufgerufen, um den Schlüsselstrom für den Block zu erzeugen. Dazwischen werden für ein zusätzliches Byte an zu authentifizierenden Daten ein Taktzyklus und für ein zusätzliches Byte an zu authentifizierenden und zu verschlüsselnden Daten zehn Taktzyklen mehr benötigt.

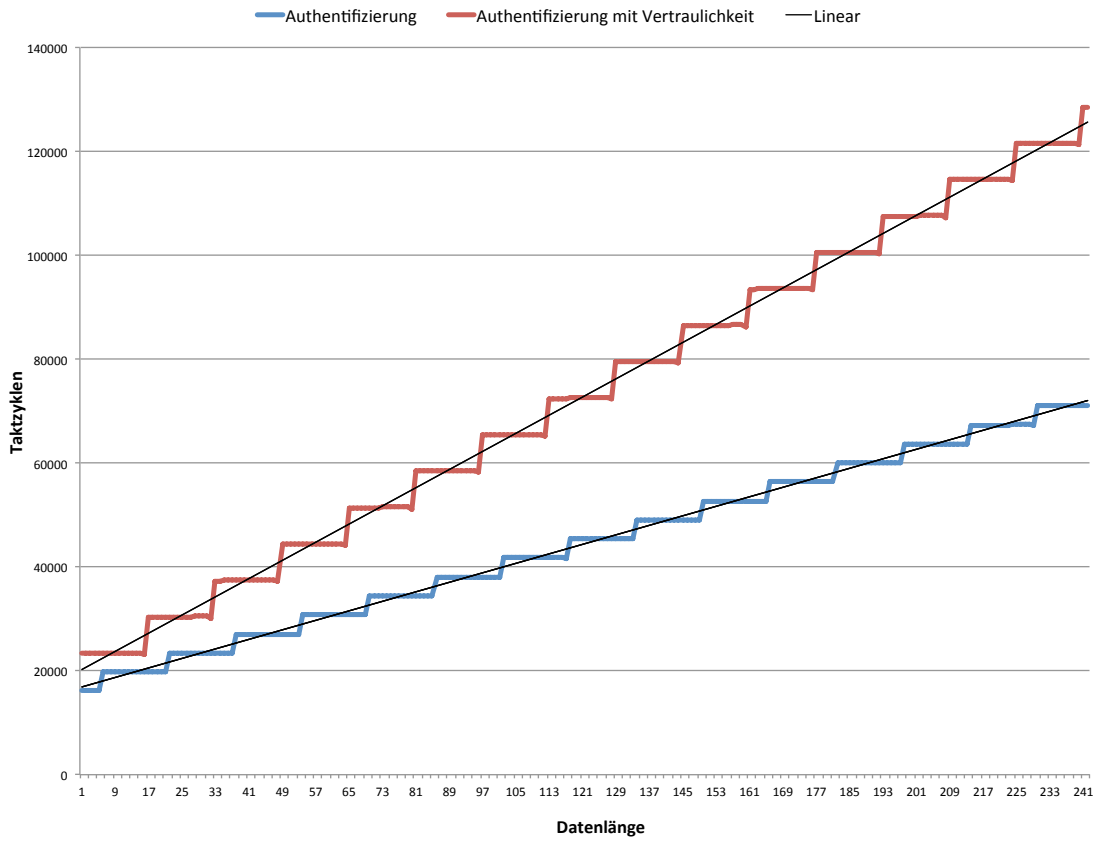

Abbildung 6.2: Performance von CCM

Ein Byte Nutzdaten (zehn Bytes insgesamt zu authentifizierende Daten) benötigen zur Authentifizierung 16.133 Taktzyklen und bei zusätzlicher Verschlüsselung der Nutzdaten 23.328 Taktzyklen. Bei einer Datenlänge von 242 Bytes Nutzdaten werden 71.049 Taktzyklen zur Authentifizierung beziehungsweise 128.638 Taktzyklen zur Authentifizierung und Verschlüsselung benötigt. Tabelle 6.4 zeigt die Anzahl der benötigten Taktzyklen für ausgewählte Längen der Nutzdaten im Detail. Im Vergleich dazu benötigt das Berechnen der Prüfsumme eines TP1-Extended-Telegramms mit maximaler Länge, aber ohne Einsatz des S-ALs, 4480 Taktzyklen.

Bei der Verifizierung von empfangenen Daten werden im Fall des Security-Modus nur Authentifizierung, abgesehen vom Vergleich der beiden MACs, die gleichen Vorgänge durchge-

Bytelänge Nutzdaten	Taktzyklen Authentifizierung	Taktzyklen Authentifizierung mit Vertraulichkeit
2	16 134	23 338
4	16 136	23 358
10	19 787	23 418
50	27 117	44 398
200	63 717	107 638
242	71 049	128 638

Tabelle 6.4: Performance von CCM

führt wie zur Generierung der Nachricht. Somit wird auch die gleiche Anzahl an Taktzyklen benötigt. Bei zusätzlich verschlüsselten Nachrichten läuft die Entschlüsselung und Generierung des MACs geringfügig schneller ab als die Verschlüsselung. Bei einer Nutzdatenlänge von einem Byte ist die Entschlüsselung um vier Taktzyklen schneller als die Verschlüsselung. Umso länger die Daten sind, umso größer wird der Unterschied der benötigten Taktzyklen. Bei einer Nutzdatenlänge von den maximal möglichen 242 Bytes bei einem TP1-Telegramm werden für die Entschlüsselung 19 Taktzyklen weniger benötigt als für die Verschlüsselung. Der Vergleich von zwei MACs mit einer Länge von 32 Bits benötigt 84 Taktzyklen. Tabelle 6.5 zeigt die benötigten Taktzyklen für die Verifizierung einer Nachricht (inklusive dem Vergleich der beiden MACs) in den Sicherheitsmodi nur Authentifizierung sowie bei zusätzlicher Entschlüsselung.

Bytelänge Nutzdaten	Taktzyklen Verifizierung bei Authentifizierung	Taktzyklen Verifizierung bei Authentifizierung mit Vertraulichkeit
2	16 218	23 418
4	16 220	23 438
10	19 871	23 498
50	27 201	44 475
200	63 801	107 706
242	71 133	128 703

Tabelle 6.5: Performance der Verifizierung

Die Tabellen 6.6 und 6.7 zeigen die Anzahl der von der Implementierung benötigten Operationen für den Dienst `S-A_Data` für eine gesendete beziehungsweise empfangene Gruppennachricht mit verschiedenen Längen. Die S-AL-Implementierung ist aufgrund der fehlenden Berechtigungen und Rollen zwar nicht vollständig, doch im Vergleich zur Anzahl der benötigten

Taktzyklen der Performance von CCM in den oben gezeigten Tabellen 6.4 und 6.5 ist dennoch leicht zu erkennen, dass die CCM-Operationen den größten Teil des Aufwandes des Dienstes `S-A_Data` ausmachen. Die zu CCM zusätzlichen Vorgänge, wie das Durchsuchen der Security Link Table, benötigen nur wenige Taktzyklen. Der Mehraufwand dieser Operationen beträgt in der Implementierung lediglich zwischen rund einem und sechs Prozent. Diese Werte wurden aus der Differenz der benötigten Taktzyklen der CCM-Operationen und aller benötigten Operationen des Dienstes `S-A_Data` der Implementierung berechnet.

Bytelänge Nutzdaten	**Taktzyklen Authentifizierung**	**Taktzyklen Authentifizierung mit Vertraulichkeit**
2	16 673	23 951
4	16 683	24 015
10	20 387	24 258
50	28 077	46 315
200	66 027	113 596
242	73 737	135 727

Tabelle 6.6: Performance der S-AL-Implementierung in Senderichtung

Bytelänge Nutzdaten	**Taktzyklen Authentifizierung**	**Taktzyklen Authentifizierung mit Vertraulichkeit**
2	16 937	24 232
4	16 939	24 288
10	20 588	24 454
50	27 918	46 151
200	64 518	112 082
242	71 850	133 835

Tabelle 6.7: Performance der S-AL-Implementierung in Empfangsrichtung

Ein Sonderfall tritt bei der auf dem RELIC-Toolkit basierenden CCM-Implementierung auf, wenn mehrmals hintereinander der gleiche Schlüssel benutzt wird, ohne dass dazwischen der Schlüssel gewechselt wird. Wie oft das bei der Verwendung des S-ALs auftritt, ist einerseits von der Anzahl der benutzten sicheren Kommunikationsbeziehungen des Knotens und andererseits von der Anzahl der definierten Schlüssel für die unterschiedlichen Kommunikationsbeziehungen abhängig. Wird ein Schlüssel mehrmals hintereinander eingesetzt, dann kann nach dem ersten Benutzen des Schlüssels ein Initialisierungsschritt ausgelassen werden, wodurch 4914 Taktzy-

klen entfallen. Damit dieser Sonderfall ausgenutzt werden kann, muss in der Implementierung der aktuelle Schlüssel mit dem zuletzt verwendeten Schlüssel verglichen werden.

6.5 Speicherbedarf

Der Speicherbedarf der Implementierung des KNX S-ALs wurde ebenfalls bestimmt. Die Implementierung des S-ALs umfasst dabei sechs Einträge in der Security Link Table, vier Schlüssel (inklusive Security Tool Key) und drei Einträge in der Security Individual Address Table. In Tabelle 6.8 sind der Speicherbedarf der implementierten Funktionen des S-ALs und der Speicherbedarf der AES-CCM-Implementierung zu sehen. Es muss dazu angemerkt werden, dass die prototypische Implementierung nicht alle Funktionen unterstützt, die in der Spezifikation des S-ALs festgelegt sind.

Der Programmcode und die Konstanten werden im Flashspeicher des Mikrocontrollers abgelegt, die Daten werden im SRAM gespeichert. Die Größe des Stacks, der Teil des SRAMs ist, musste von der Standardgröße auf eine Größe von 1000 Bytes gesetzt werden, damit es bei der Ausführung der CCM-Operationen zu keinem Überlauf des Stacks kommt.

Modul	Speicher Code	Speicher Konstanten	Speicher Daten	Gesamt
S-AL	3334 Bytes	407 Bytes	704 Bytes	4445 Bytes
AES-CCM	2116 Bytes	2257 Bytes	2 Bytes	4375 Bytes

Tabelle 6.8: Speicherbedarf des S-ALs der Implementierung

6.6 Fazit

Der für die Implementierung verwendete MSP430F5438A bietet mit seinen 256 Kilobytes Flashspeicher und 16 Kilobytes SRAM genügend Ressourcen, um den Programmcode des KNX S-ALs sowie eine Vielzahl an Verbindungsinformationen und Schlüssel aufzunehmen (siehe auch Tabelle 5.6). Es werden jedoch viel weniger leistungsfähigere Mikrocontroller mit nur wenig Speicher eingesetzt, wie beispielsweise der MSP430F2121 mit lediglich 4 Kilobytes Flash und 256 Bytes SRAM oder der MSP430F2330 mit 8 Kilobytes Flash und 1 Kilobyte SRAM. Wie in Tabelle 6.8 zu entnehmen ist, benötigt alleine die S-AL-Implementierung des Prototyps über 8 Kilobytes an Flashspeicher. Der S-AL kann in solchen Mikrocontrollern somit nicht eingesetzt werden, da mehr Speicherplatz benötigt wird.

Der MSP430F5438A unterstützt Taktfrequenzen bis zu 25 MHz. Weniger leistungsstarke Mikrocontroller weisen Taktfrequenzen von maximal 8 MHz (beispielsweise die MSP430F1x-Serie) oder 16 MHz (beispielsweise die MSP430F2x-Serie oder ATmega16) auf. Bei einer Taktfrequenz von 4 MHz benötigt die Authentifizierung und Verschlüsselung von zwei Bytes Nutzdaten (23 338 Taktzyklen, siehe Tabelle 6.4) 5,8 Millisekunden. Die zusätzliche Verschlüsselung einer Nachricht benötigt somit auch bei leistungsschwachen Mikrocontrollern lediglich einige

Millisekunden. Tabelle 6.9 zeigt die Zeitdauer der CCM-Operationen für verschiedene Nutzdatenlängen bei einer Taktfrequenz von 4 MHz (die Anzahl der Taktzyklen siehe Tabelle 6.4).

Bytelänge Nutzdaten	**Dauer Authentifizierung**	**Dauer Authentifizierung mit Vertraulichkeit**
2	4,03 ms	5,83 ms
4	4,03 ms	5,84 ms
10	4,95 ms	5,85 ms
50	6,78 ms	11,1 ms
200	15,93 ms	26,9 ms
242	17,76 ms	32,16 ms

Tabelle 6.9: Performance von CCM bei 4 MHz Taktfrequenz

Wie in Punkt 5.2.3 behandelt, beträgt die Länge des MACs beim S-AL 32 Bits, was für ein hohes Maß an Security bei Übertragungsmedien mit großer Kapazität unter Umständen nicht ausreichend sein kann. Aus diesem Grund wurde die Berechnungsdauer längerer MACs untersucht, die für alle Datenlängen konstant ist. Die zusätzlich anfallenden Taktzyklen sind sehr gering, sie stellen somit kein Hindernis für die Integration längerer MACs dar. Tabelle 6.10 zeigt abschließend die genauen Werte.

Länge des MACs	**zusätzliche Taktzyklen**
64 Bits	56
128 Bits	172

Tabelle 6.10: Berechnungsdauer längerer MACs

KAPITEL 7

Schlussbemerkung

7.1 Zusammenfassung

Zu Beginn dieser Arbeit wurde ein Überblick über Gebäudeautomation sowie Sicherheit in der Gebäudeautomation gegeben und Anforderungen an die Security vorgestellt, die bei der Analyse des KNX Secure Application Layers (S-ALs) wieder herangezogen wurden. Darauf folgend wurde der Stand der Technik von kryptographischen Verfahren, insbesondere von AES und dessen Betriebsmodus CCM, die im S-AL zum Einsatz kommen, beleuchtet. Bei CCM handelt es sich um eine Kombination der Betriebsmodi CTR und CBC-MAC, wodurch unter Einsatz eines einzelnen Schlüssels sowohl Authentifizierung und Integrität als auch Vertraulichkeit geboten werden. Nicht außer Acht gelassen wurde dabei der Security-Aspekt der Algorithmen. Bei CCM wird für jede sichere Nachricht unter einem Schlüssel ein einmaliger Initialisierungsvektor gefordert, damit Vertraulichkeit garantiert werden kann. Außerdem wurden die Security-Konzepte der Gebäudeautomationssysteme LonWorks, BACnet und ZigBee untersucht und einander gegenübergestellt. Anschließend wurde der Aufbau und die Funktionsweise von KNX ausführlich vorgestellt. KNX bietet ursprünglich bis auf eine sehr rudimentäre Autorisierung für Managementfunktionen keinerlei Sicherheitsmaßnahmen.

Der kürzlich definierte KNX S-AL soll den Einsatz von sicherheitskritischen Diensten und Anwendungen in KNX ermöglichen. In der Arbeit wurde das Sicherheitskonzept des S-ALs analysiert. Die Forderung von AES-CCM nach einem einmaligen Initialisierungsvektor für jede sichere Nachricht unter einem Schlüssel wird durch die Kombination der einzigartigen Sequenznummer mit der physikalischen Quelladresse des KNX-Knotens erfüllt. Da der S-AL eine leicht abgeänderte Version des CCM-Algorithmus einsetzt, wurde gezeigt, dass der Beweis für die Sicherheit von CCM auch für diese spezifische Variante von CCM gültig ist. In der Arbeit wurden weiters Angriffe auf die sichere KNX-Kommunikation untersucht, wobei der S-AL nur wenige Angriffspunkte bietet, die noch dazu von bestimmten, unwahrscheinlichen Bedingungen, wie dem Erraten eines gültigen MACs einer Nachricht oder der Erlangung eines oder mehrerer Schlüssel, abhängig sind. DoS-Angriffe können, wie bei anderen Systemen auch, nur durch zusätzliche physische Maßnahmen abgewendet werden.

Der Vergleich des KNX S-ALs mit den Sicherheitsmechanismen von BACnet und ZigBee zeigt, dass die Autorisierung des S-ALs deutlich feiner vorgegeben werden kann als bei den anderen Standards. Bei der Datenvertraulichkeit befindet sich der S-AL auf einer Ebene mit BACnet und ZigBee. Mit AES-128 wird auch in allen drei Systemen der gleiche Verschlüsselungsalgorithmus eingesetzt. Bei der Datenaktualität hat BACnet gegenüber ZigBee und dem S-AL den Vorteil, dass die Nachrichten in BACnet zusätzlich zu einer Nummer einen Zeitstempel enthalten. Der MAC, der für die Authentifizierung und Datenintegrität zuständig ist, ist im S-AL mit einer Länge von 32 Bits vergleichsweise kurz, jedoch im typischen Bereich des CCM-Standards. In Abschnitt 5.2.3 wurde erläutert, dass auf Grund der beschränkten Übertragungskapazitäten der Medien KNX RF sowie TP1 ein MAC mit einer Länge von 32 Bits ausreichende Sicherheit bietet.

Die Implementierung einer prototypischen Umsetzung des S-ALs wurde am Ende der Arbeit behandelt, um dessen Machbarkeit zu demonstrieren. Es wurden ein sicheres KNX-Telegramm mit Authentifizierung sowie mit zusätzlicher Verschlüsselung gezeigt. Die Performance von CCM und der Implementierung des S-ALs sowie der notwendige Speicherbedarf wurden ebenfalls untersucht. Die bei der Implementierung aufgetretenen Probleme und dadurch erforderlichen Änderungen betreffend der zu authentifizierenden und zu verschlüsselnden Daten wurden ebenso festgehalten.

7.2 Fazit

Der S-AL integriert Sicherheitsmechanismen in den KNX-Standard. Das wiederum erlaubt die Integration von Diensten aus sicherheitsrelevanten Domänen in KNX. Diese Dienste werden bisher aufgrund der fehlenden Sicherheitsmechanismen vieler Gebäudeautomationssysteme zumeist von eigenständigen, anwendungsspezifischen Subsystemen abgedeckt. Die Analyse des S-ALs zeigt, dass die Anforderungen der Security nach Authentifizierung, Datenintegrität, Autorisierung, Datenaktualität und Datenvertraulichkeit erfüllt werden, wodurch ein sicherer Datenaustausch möglich ist. Der Beweis der Sicherheit von CCM ist auch für die im S-AL verwendeten Variante gültig. Ein Vergleich des S-ALs mit den Sicherheitsmechanismen der Systeme BACnet und ZigBee zeigt, dass sich die Sicherheitskonzepte auf dem gleichen Level befinden.

Die Implementierung des Prototyps demonstriert die Machbarkeit des S-ALs. Aufgrund des zusätzlich benötigten Speichers kann der S-AL nicht auf Mikrocontrollern mit lediglich 4 oder 8 Kilobytes Programmspeicher ausgeführt werden. Die Ausführung der auf dem RELIC-Toolkit basierenden CCM-Verschlüsselung nimmt hingegen bei einer Taktfrequenz von 4 MHz lediglich einige Millisekunden in Anspruch.

7.3 Weiterführende Arbeit

Die prototypische Implementierung des KNX S-ALs basiert auf einer AES-CCM-Erweiterung des RELIC-Toolkits. Es wurden keine anderen CCM-Implementierungen hinsichtlich ihrer Unterschiede in der Performance und dem Speicherbedarf untersucht. Die CC430-Serie der MSP430-Mikrocontroller verfügt über einen AES-Beschleuniger, bei dem AES in Hardware ausgeführt wird. Durch diese Hardwarebeschleunigung ist CCM bei der Verschlüsselung zirka zehnmal

und bei der Entschlüsselung achtmal schneller als AES in Software [15]. In dieser Arbeit wurde nur CCM in Software untersucht. Im S-AL wird eine leicht abgeänderte Version von CCM eingesetzt, in der für KNX spezifische Werte verwendet werden. Für diese CCM-Variante müssen noch Testvektoren erstellt werden. Der KNX-Stack der prototypischen Implementierung ist nicht vollständig. Die Implementierung des S-ALs umfasst nicht alle spezifizierten Funktionalitäten. Eine Erweiterung des KNX-Stacks und des S-ALs erlaubt auch eine genauere Bestimmung des Speicherbedarfs und der Performance des S-ALs. Das Sicherheitskonzept von KNX fußt auf den zwei Säulen KNX IP Secure und KNX Data Security, wobei letzteres in dieser Arbeit untersucht wurde. Eine Kombination des S-ALs mit KNX IP Secure wurde nicht analysiert.

Abbildungsverzeichnis

2.1 3-Schichtenmodell der Gebäudeautomation [29, 59] 4
2.2 2-Schichtenarchitektur der Gebäudeautomation [21, 59] 5
2.3 Dreieck der Anforderungen an die Security [54] 7
2.4 Angriffe, Bedrohungen und Vulnerabilitäten [17, 46, 51] 8
2.5 Angriffe auf Gebäudeautomationssysteme [17] 10

3.1 Eingabe und Ausgabe des States [39] 19
3.2 Struktur des AES-Algorithmus [46] 20
3.3 Anwendung der S-Box in `SubBytes()` [39] 20
3.4 Verschlüsselung mit CTR [10] 24
3.5 Entschlüsselung mit CTR [10] 25
3.6 Verschlüsselung mit CBC [10] 26
3.7 Entschlüsselung mit CBC [10] 26
3.8 Struktur von CBC-MAC [38] 27
3.9 CBC-MAC in CCM [31, 38] 29
3.10 Verschlüsselung im CTR-Modus in CCM [31] 30
3.11 Entschlüsselung im CTR-Modus in CCM [31] 31
3.12 Standards der Heim- und Gebäudeautomation im 3-Schichtenmodell [17] 32
3.13 Authentifizierung in LonWorks [18] 33
3.14 Architektur des ZigBee-Stacks [56] 36

4.1 KNX-Topologie [32] 42
4.2 Struktur der physikalischen Adresse [32] 43
4.3 Schichten im OSI-Modell und in KNX 44
4.4 Interaktivität der Schichten in KNX 45
4.5 Interaktivität des Application Layers [32] 48
4.6 Application Interface Layer [32] 49
4.7 Struktur des KNX-Standard-Telegramm [32] 50
4.8 Struktur des KNX-Extended-Telegramm [32] 50
4.9 Struktur des KNX-RF-Frame [32] 51
4.10 Autorisierung in KNX 53

5.1 Datensicherheit im KNX-Stack [31] 56
5.2 Format der sicheren Daten (bei TP1) [31] 57

5.3 Generierung, Übertragung und Verifizierung des MACs 59
5.4 Beispiel eines Replay-Angriffs 69
5.5 Replay-Angriff in dem die Originalnachricht das Ziel nicht erreicht 69
5.6 Nachricht mit hoher Sequenznummer 70
5.7 DoS-Angriff mit verbindungsorientierter Kommunikationsbeziehung in KNX . . . 72
5.8 DoS-Angriff bei sicherer KNX-Kommunikation 72
5.9 DoS-Erkennung und Prävention in der Gebäudeautomation [22] 74
5.10 Isolation eines Gerätes als DoS-Gegenmaßnahme [22] 75
5.11 Isolation eines Gerätes mit virtuellen Brücken als DoS-Gegenmaßnahme [22] . . . 75
5.12 Erniedrigung der Sequenznummer eines Senders 76
5.13 Erhöhung der Sequenznummer eines Senders auf den Maximalwert 76

6.1 TP-UART 2 Modul [53] 86
6.2 Performance von CCM 91

Tabellenverzeichnis

3.1 Formatierung von B_0 in CCM [11] 28
3.2 Formatierung von Ctr_j in CCM [11] 29
3.3 Security-Levels in ZigBee [62] 37
3.4 Übersicht der Sicherheitskonzepte von Gebäudeautomationssystemen [18] 39

4.1 Passwörter und Berechtigungsstufen [32] 52

5.1 Formatierung von B_0 in CCM [11] 61
5.2 Formatierung von B_0 von CCM in KNX [31] 62
5.3 Formatierung von Ctr_j in CCM [11] 62
5.4 Formatierung von Ctr_j von CCM in KNX [31] 62
5.5 Speicherbedarf der Eigenschaften des Security Interface Objects 66
5.6 Unterschiedlicher Speicherbedarf bei unterschiedlichen Konfigurationen 66
5.7 Speicherbedarf simple Berechtigungstabelle 67
5.8 KNX S-AL Sicherheitsmechanismen im Vergleich 79

6.1 KNX-TP1-Standardtelegramm ohne Sicherheitsmechanismen 89
6.2 KNX-TP1-Standardtelegramm mit Authentifizierung 90
6.3 KNX-TP1-Standardtelegramm mit Authentifizierung und Vertraulichkeit 90
6.4 Performance von CCM 92
6.5 Performance der Verifizierung 92
6.6 Performance der S-AL-Implementierung in Senderichtung 93
6.7 Performance der S-AL-Implementierung in Empfangsrichtung 93
6.8 Speicherbedarf des S-ALs der Implementierung 94
6.9 Performance von CCM bei 4 MHz Taktfrequenz 95
6.10 Berechnungsdauer längerer MACs 95

Literaturverzeichnis

[1] P. Szałachowski and Z. Kotulski. Secure Time Information in the Internet Key Exchange Protocol. In *Annales UMCS Informatica*, volume 11, pages 41–56, Warsaw, Poland, January 2011. Versita.

[2] American Society of Heating, Refrigerating and Air-Conditioning Engineers, Inc. *ANSI/ASHRAE Addendum g to ANSI/ASHRAE Standard 135-2008, BACnet - A Data Communication Protocol for Building Automation and Control Networks*. Atlanta, GA, USA, 2010.

[3] M. Bellare and C. Namprempre. Authenticated Encryption: Relations among Notions and Analysis of the Generic Composition Paradigm. In *Advances in Cryptology - ASIACRYPT 2000*, volume 1976 of *Lecture Notes in Computer Science*, pages 531–545. Springer-Verlag, 2000.

[4] A. Biryukov, O. Dunkelman, N. Keller, D. Khovratovich, and A. Shamir. Key Recovery Attacks of Practical Complexity on AES-256 Variants with up to 10 Rounds. In Henri Gilbert, editor, *Advances in Cryptology – EUROCRYPT 2010*, volume 6110 of *Lecture Notes in Computer Science*, pages 299–319. Springer Berlin Heidelberg, 2010.

[5] A. Biryukov and D. Khovratovich. Related-Key Cryptanalysis of the Full AES-192 and AES-256. In Mitsuru Matsui, editor, *Advances in Cryptology – ASIACRYPT 2009*, volume 5912 of *Lecture Notes in Computer Science*, pages 1–18. Springer Berlin Heidelberg, 2009.

[6] A. Bogdanov, D. Khovratovich, and C. Rechberger. Biclique Cryptanalysis of the Full AES. In DongHoon Lee and Xiaoyun Wang, editors, *Advances in Cryptology – ASIACRYPT 2011*, volume 7073 of *Lecture Notes in Computer Science*, pages 344–371. Springer Berlin Heidelberg, 2011.

[7] W.E. Burr. Selecting the Advanced Encryption Standard. In *IEEE Security Privacy*, volume 1, pages 43–52, 2003.

[8] J. Daemen and V. Rijmen. *AES Proposal: Rijndael*, September 1999.

[9] W. Diffie and M. Hellman. New directions in cryptography. In *IEEE Transactions on Information Theory*, volume 22, pages 644–654, November 1976.

[10] M. Dworkin. *NIST Special Publication 800-38A, Recommendation for Block Cipher Modes of Operation*. National Institute of Standards and Technology, Gaithersburg, MD, USA, December 2001.

[11] M. Dworkin. *NIST Special Publication 800-38C, Recommendation for Block Cipher Modes of Operation: The CCM Mode for Authentication and Confidentiality*. National Institute of Standards and Technology, Gaithersburg, MD, USA, May 2004.

[12] EnOceanGmbH, 82041 Oberhaching, Germany. *EnOcean Standard: Security of EnOcean Radio Networks V1.9*, July 2013.

[13] N. Ferguson, B. Schneier, and T. Kohno. *Cryptography Engineering: Design Principles and Practical Applications*. Wiley Publishing, Indianapolis, USA, March 2010.

[14] Electronic Frontier Foundation. *Cracking DES: Secrets of Encryption Research, Wiretap Politics, and Chip Design*. O'Reilly Media, 1st edition, 1998.

[15] C. P. L. Gouvêa and J. López. High Speed Implementation of Authenticated Encryption for the MSP430X Microcontroller. In *Proceedings of the 2nd International Conference on Cryptology and Information Security in Latin America*, LATINCRYPT'12, pages 288–304, Berlin, Heidelberg, 2012. Springer-Verlag.

[16] W. Granzer. Security in Networked Building Automation Systems. Master's thesis, Vienna University of Technology, Institute of Computer Aided Automation, Automation Systems Group, November 2005.

[17] W. Granzer. *Secure Communication in Home and Building Automation Systems*. PhD thesis, Vienna University of Technology, Institute of Computer Aided Automation, Automation Systems Group, 2010.

[18] W. Granzer and W. Kastner. Security Analysis of Open Building Automation Systems. In *Proc. 29th International Conference on Computer Safety, Reliability and Security (SAFECOMP '10)*, pages 303–316, September 2010.

[19] W. Granzer, W. Kastner, G. Neugschwandtner, and F. Praus. A Modular Architecture for Building Automation Systems. In *Factory Communication Systems, 2006 IEEE International Workshop on*, pages 99–102, 2006.

[20] W. Granzer, G. Neugschwandtner, and W. Kastner. EIBsec: A Security Extension to KNX/EIB. In *Konnex Scientific Conference*, November 2006.

[21] W. Granzer, F. Praus, and W. Kastner. Security in Building Automation Systems. In *IEEE Transactions on Industrial Electronics*, volume 57, pages 3622–3630, 2010.

[22] W. Granzer, C. Reinisch, and W. Kastner. Denial-of-Service in Automation Systems. In *Emerging Technologies and Factory Automation, 2008. ETFA 2008. IEEE International Conference on*, pages 468–471, September 2008.

[23] Wolfgang Granzer, Wolfgang Kastner, Georg Neugschwandtner, and Fritz Praus. Security in Networked Building Automation Systems. In *Proc. 6th IEEE International Workshop on Factory Communication Systems (WFCS '06)*, pages 283–292, June 2006. Best paper award of WFCS '06.

[24] D. Gullasch, E. Bangerter, and S. Krenn. Cache Games – Bringing Access-Based Cache Attacks on AES to Practice. In *2011 IEEE Symposium on Security and Privacy*, pages 490–505, 2011.

[25] D. G. Holmberg. BACnet Wide Area Network Security Threat Assessment. NISTIR Technical Report 7009, National Institute of Standards and Technology, Gaithersburg, MD, USA, 2003.

[26] International Organization for Standardization. *ISO/IEC 7498-1:1994, Information Technology - Open Systems Interconnection - Basic Reference Model: The Basic Model*, 2nd edition, November 1994.

[27] J. Jonsson. On the Security of CTR + CBC-MAC. In *Revised Papers from the 9th Annual International Workshop on Selected Areas in Cryptography*, SAC '02, pages 76–93, London, UK, 2003. Springer-Verlag.

[28] C. Karlof, N. Sastry, and D. Wagner. TinySec: A Link Layer Security Architecture for Wireless Sensor Networks. In *Proceedings of the 2nd International Conference on Embedded Networked Sensor Systems*, SenSys '04, pages 162–175, New York, NY, USA, 2004. ACM.

[29] W. Kastner, G. Neugschwandtner, S. Soucek, and H.M. Newmann. Communication Systems for Building Automation and Control. In *Proceedings of the IEEE*, volume 93, pages 1178–1203, 2005.

[30] W. Kastner, F. Praus, G. Neugschwandtner, and W. Granzer. KNX. In B.M. Wilamowski and J.D. Irwin, editors, *Industrial Electronics Handbook*, volume 2: Industrial Communication Systems, chapter 42, pages 1–14. CRC Press, 2nd edition, 2011.

[31] KNX Association. *KNX Data Security, Application Note 158/13 v02*, May 2013.

[32] KNX Association. *KNX Specifications v2.1*, November 2013.

[33] W. Köhler. Simulation of a KNX Network with EIBsec Protocol Extensions. Master's thesis, Vienna University of Technology, Institute of Computer Aided Automation, Automation Systems Group, August 2008.

[34] L. Krammer, S. De Bruyne, W. Kastner, and W. Granzer. Security Erweiterung für den KNX Standard. In *Innosecure Kongress*, pages 31–40, September 2013.

[35] C. Krügel. *Network Alertness - Towards an adaptive, collaborating Intrusion Detection System*. PhD thesis, Vienna University of Technology, May 2002.

[36] H. Li, Z. Jia, and X. Xue. Application and Analysis of ZigBee Security Services Specification. In *Networks Security Wireless Communications and Trusted Computing (NSWCTC), 2010 Second International Conference on*, volume 2, pages 494–497, 2010.

[37] J. López and R. Dahab. An Overview of Elliptic Curve Cryptography. Technical Report IC-00-10, Institute of Computing – UNICAMP, May 2000.

[38] A. J. Menezes, P. C. van Oorschot, and S. A. Vanstone. *Handbook of Applied Cryptography*. CRC Press, 1996.

[39] National Institute of Standards and Technology, Gaithersburg, MD, USA. *FIPS 197, Specification for the Advanced Encryption Standard (AES)*, November 2001.

[40] T. Novak and A. Gerstinger. Safety- and Security-Critical Services in Building Automation and Control Systems. In *IEEE Transactions on Industrial Electronics*, volume 57, pages 3614–3621, November 2010.

[41] T. Novak, A. Treytl, and P. Palensky. Common Approach to Functional Safety and System Security in Building Automation and Control Systems. In *ETFA*, pages 1141–1148. IEEE, 2007.

[42] L. B. Oliveira, D. F. Aranha, C. P. L. Gouvêa, M. Scott, D. F. Címara, J. López, and R. Dahab. TinyPBC: Pairings for Authenticated Identity-based Non-interactive Key Distribution in Sensor Networks. In *Computer Communications*, volume 34, pages 485–493, 2011.

[43] OMS Group. *Appendix to the Open Metering System Specification - Glossary of Terms used in or related to the OMS V.1.0.1*, November 2011.

[44] G. Padmavathi and D. Shanmugapriya. A Survey of Attacks, Security Mechanisms and Challenges in Wireless Sensor Networks. In *International Journal of Computer Science and Information Security*, volume 4, 2009.

[45] A. Perrig, R. Szewczyk, V. Wen, D. Culler, and J. D. Tygar. SPINS: Security Protocols for Sensor Networks. In *Wireless Networks*, volume 8 issue 5, pages 521–534, Hingham, MA, USA, September 2002. Kluwer Academic Publishers.

[46] C. P. Pfleeger and S. L. Pfleeger. *Security in Computing*. Prentice Hall PTR, Upper Saddle River, NJ, USA, 4th edition, 2006.

[47] S. Ravi, A. Raghunathan, P. Kocher, and S. Hattangady. Security in Embedded Systems: Design Challenges. In *ACM Transactions on Embedded Computing*, volume 3, pages 461–491, New York, NY, USA, August 2004. ACM.

[48] C. Reinisch, W. Kastner, G. Neugschwandtner, and W. Granzer. Wireless Technologies in Home and Building Automation. In *Industrial Informatics, 2007 5th IEEE International Conference on*, volume 1, pages 93–98, 2007.

[49] B. Schneier, J. Kelsey, D. Whiting, D. Wagner, C. Hall, N. Ferguson, T. Kohno, and M. Stay. The Twofish Team's Final Comments on AES Selection, May 2000.

[50] C. Schwaiger and A. Treytl. Smart Card Based Security for Fieldbus Systems. In *Emerging Technologies and Factory Automation, 2003. Proceedings. ETFA '03. IEEE Conference*, volume 1, pages 398–406, 2003.

[51] R. Shirey. *RFC 4949: Internet Security Glossary, Version 2*. The Internet Society, August 2007.

[52] Siemens AG, Regensburg, Deutschland. *Bus Transceiver Module 117/12 PCBA (TP-UART 2 Evaluation Board)*, May 2012.

[53] Siemens AG, Regensburg, Deutschland. *KNX EIB TP-UART 2 - IC Technical Manual*, August 2013.

[54] W. Stallings and L. Brown. *Computer Security: Principles and Practice*. Prentice Hall, Upper Saddle River, NJ, USA, 2nd edition, 2011.

[55] G. Stoneburner. *NIST Special Publication 800-33, Underlying Technical Models for Information Technology Security*. National Institute of Standards and Technology, Gaithersburg, MD, USA, December 2001.

[56] M. Sun and Y. Qian. Study and Application of Security Based on ZigBee Standard. In *Multimedia Information Networking and Security (MINES), 2011 Third International Conference on*, pages 508–511, 2011.

[57] Texas Instruments Incorporated, Dallas, TX, USA. *MSP430F543xA, MSP430F541xA Mixed Signal Microcontroller (Revision D, SLAS655D)*, August 2013.

[58] Texas Instruments Incorporated, Dallas, TX, USA. *MSP430x5xx and MSP430x6xx Family User's Guide (Revision M, SLAU208J)*, February 2013.

[59] W. Kastner und G. Neugschwandtner. Datenkommunikation in der verteilten Gebäudeautomation. In *Bulletin SEV/VSE*, 2006.

[60] J. P. Walters, Z. Liang, W. Shi, and V. Chaudhary. Wireless Sensor Network Security: A Survey. In Yang Xiao, editor, *Security in Distributed, Grid, and Pervasive Computing*. Auerbach Publications, CRC Press, April 2007.

[61] D. Whiting, R. Housley, and N. Ferguson. *RFC 3610: Counter with CBC-MAC (CCM)*. The Internet Society, September 2003.

[62] ZigBee Alliance, San Ramon, CA, USA. *ZigBee Specification (ZigBee Document 053474r17)*, January 2008.