Journal of Universal Computer Science, vol. 22, no. 9 (2016), 1225-1248
submitted: 9/7/16, accepted: 30/8/16, appeared: 1/9/16 © J.UCS

A Comprehensive Dependability Approach for Building Automation Networks

Lukas Krammer, Wolfgang Kastner
(Technische Universität Wien, Vienna, Austria
lkrammer, k@auto.tuwien.ac.at)

Thilo Sauter
(Technische Universität Wien, Vienna, Austria
and
Danube University Krems, Austria
sauter@ict.tuwien.ac.at)

Abstract: Building automation is a widespread topic that evolved over the past decades. Initially, building automation systems were used for heating, ventilation and air conditioning applications as well as for lighting and shading. Today, the term building automation covers many more application domains such as alarm systems, access control or life safety systems. In particular in the latter application domains, reliable, safe and secure communication systems are of utmost importance.

This paper introduces a generic concept for bringing dependability – especially reliability, safety and security – into the area of building automation. The proposed approach is able to extend existing building automation systems with dependability features. For this purpose, the communication stack of a particular system is extended by adding an intermediate layer. This so-called Generic Dependability Layer is transparent to provide seamless integration. Thereby, reliability is addressed in terms of fault tolerance by offering redundant network topologies. A heartbeat mechanism and an acknowledgment procedure as well as a specific message format satisfy the safety requirements. Moreover, the dependability layer offers security mechanisms that establish a secured channel among communicating nodes providing integrity, confidentiality, data freshness and availability.

Key Words: Dependability, Reliability, Safety, Security, Availability, Building Automation

Category: C.2.0, D.4.5, K.6.5

1 Introduction

Buildings contain a number of technical services in order to be able to fulfill their task of providing a comfortable, secure and safe environment. Apart from heating, ventilation and air-conditioning as well as lighting and shading, critical services such as fire alarm or access control systems are nowadays added to building automation. However, each application domain has different demands regarding dependability with its main attributes reliability, availability, safety, confidentiality, integrity and maintainability [Avizienis et al., 2004].

While well-established dependable technologies are widely used in industrial automation or in the automotive area, they can hardly be applied to building automation systems (BASs). One reason for this is the long life-cycle of automation systems. This often results in very heterogeneous installations consisting of many different technologies [Kastner et al., 2005]. However, the most important reason, why industrial components are not applied to the BA domain is the cost pressure. In contrast to industrial automation, the willingness to invest in non-functional features guaranteeing reliable and secure communication is very limited. Thus, today's building automation technologies hardly take care of a holistic approach for reaching dependability. In some cases, technologies contain of basic security mechanisms, but they rarely support any mechanisms that increase their reliability. Although desirable, the replacement of existing building automation networks and installation is not imaginable, due to the previously introduced reasons.

This article introduces a generic dependability framework as sketched in [Krammer et al., 2016] which is based on the idea of a generic sublayer extending the original communication stack of an automation system by dependability features. This so called Generic Dependability Layer (GDL) uses services of the underlying stack for communication and provides a native communication interface for the upper layers. Depending on the particular communication system and the envisaged network structure, the GDL can be placed between different layers of almost every communication system, if particular requirements are satisfied.

Before details of the GDL are discussed, relevant state of the art and technologies are presented in Section 2. Subsequently, the system architecture of the GDL is introduced in Section 3 embracing the system model and the device model. Additionally, a fault model specifies all types of faults that can be tolerated. After discussing the prerequisites, the detailed concept is introduced in Section 4. Therein, the mechanisms are distinguished between *reliability*, *safety*, and *security*. In order to evaluate the presented approach, Section 5 sketches the static fault analysis for showing the correctness of safety and reliability mechanisms, and a theoretical analysis discussing the security features. Finally, Section 6 introduces a performance evaluation based on simulation that shows the feasibility and applicability of the new approach.

2 Terminology and related technologies

Dependability plays an important role in system design. There exist standards addressing dependability management, risk management and system design [IEC, 2010a, IEC, 2007]. In the context of computer systems, the term was introduced in [Laprie, 1992]. The term *dependability* covers different aspects and

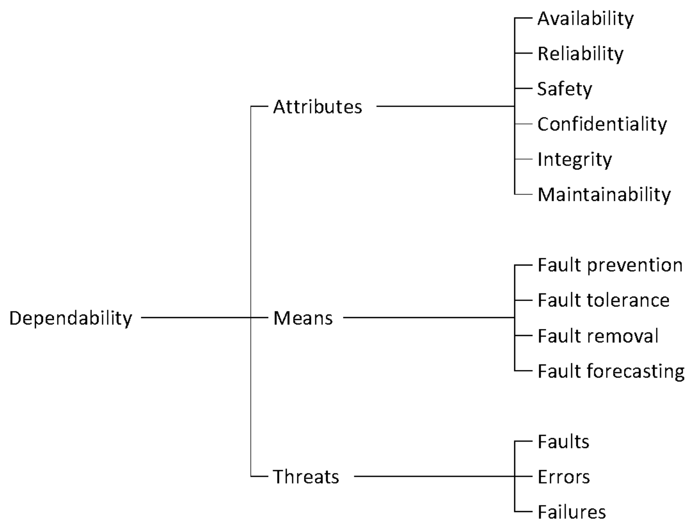

Figure 1: Dependability tree (adapted from[Avizienis et al., 2001])

properties which can be categorized in three parts: *attributes*, *means* and *threats* illustrated in Figure 1. This article is structured according to the dependability attributes.

Safety in automation is a far-ranging topic [Sauter et al., 2011] with many different standards addressing its requirements. Although most of the standards are concerned with application areas in the context of industrial automation or communication, IEC 61508 [IEC, 2011a] is a generic standard that is not dedicated to a specific automation domain and focuses on functional safety. The IEC 61784-3 standard series specifies a number of safety-related communication protocols [IEC, 2010b]. Most of them are intended to be used in industrial automation systems. Nevertheless, one technology in this series, called "openSAFETY", establishes safe communication based on a black channel approach and thus could even be deployed to BASs.

Reliability is addressed by various network redundancy standards in the context of industrial automation. The redundant ring protocol (RRP), as standardized in IEC 62439-7 [IEC, 2011b], is a network recovery protocol based on Ethernet and relying on physical ring topologies. The Ethernet automatic protection switching (EAPS) protocol, as specified in [IETF, 2003b], defines a ring protection protocol based on a set of VLANs. Another ring protection protocol which is standardized in IEC 62439-2 [IEC, 2010c] is named media redundancy protocol (MRP). While all previously mentioned approaches are based on conventional Ethernet medium access, *TTEthernet* [TTTech, 2010, Kopetz et al., 2005] follows a time-triggered approach and is standardized in SAE AS6802 [SAE, 2011].

Most of the **security** applications rely on cryptographic mechanisms. A summary of security methodologies for distributed systems can be found in [Uzunov et al., 2012]. In the context of encryption and decryption, cryptographic ciphers are used. On the other hand, cryptographic hash functions allow signing and authenticating data. Besides dedicated algorithms for computing cryptographic hash functions such as MD5 [IETF, 1992] or SHA [NIST, 202], even ciphers can be used to compute cryptographic hash functions. Cryptographic ciphers are subdivided between stream ciphers or block ciphers. However, stream ciphers such as A5/4 [ETSI, 2008] or RC4 [Rivest, 1992, IETF, 2003c] provide only limited security. Thus, block ciphers are mainly applied in today's applications (e.g., DES [NIST, 1999], TDES [NIST, 2005b], AES [NIST, 2001a]). Since block ciphers require inputs of a predefined length, so-called *modes of operation* come into play as introduced in [NIST, 2001b].

In the present paper, **maintainability** has been excluded from the analysis as it strongly relies on the deployed system.

Unfortunately, support of full dependability is hardly addressed in any building automation technology. While BACnet and ZigBee provide state-of-the-art security services, LON only offers basic authentication mechanisms which are insufficient for today's systems [Treytl et al., 2005]. Recently in KNX, security has been extended by a comprehensive security suite as introduced in [Krammer et al., 2013] and examined in [Judmayer et al., 2014]. Safety for KNX was addressed in [Kastner and Novak, 2009] specifying a safe device model and a frame format for safety messages. Also reliability in terms of network redundancy was investigated in [Krammer et al., 2012]. In LON, safety features were proposed by a project called safetyLON (cf. [Novak and Tamandl, 2007, Mentzel, 2010]) and resulted in comprehensive safety features. Currently, availability and reliability are addressed by hardly any technology. By default, ZigBee allows rerouting in case of a fault which would satisfy basic reliability demands. However, due to the unreliable nature of the wireless medium, reliability is limited in general. BACnet does neither support safety nor reliability claims. However, since BACnet relies on various transmission media, they can be used to establish a reliable channel.

In conclusion, none of the mentioned technologies or standards provides sufficient dependability support. While security features are getting more and more important even in building automation, reliability, availability and safety is hardly addressed. In order to close the gap between current BA technologies and future BASs by addressing also the heterogeneity technologies, this article introduces a concept applicable to the mentioned technologies.

3 System architecture

3.1 Device model

The architecture of the presented approach is based on a communication topology where physical devices are connected by two or more wired communication channels or independent (i.e., no mutual influences possible) wireless channels. The communication system is based on a layered model. In contrast to overlay networks, the GDL resides between two layers of the communication stack as depicted in Figure 2. The GDL relies on communication services of the underlying system which are further denoted as *low-level communication services*. The GDL is transparent for operative communication services (i.e., this does not include management services). Thus, it provides exactly these services (i.e., *high-level communication services*) to the upper layers. In addition to the communication services, which are used to control the GDL, *management services* are offered to the application. Furthermore, these services are used to notify the application about dependability related events. Since most of the communication systems specify additional services (e.g., management services) between two layers of the communication stack, these services have to be handled by a technology-specific module (*optional services*).

In contrast to conventional devices, a *dependability node* consists of multiple communication interfaces which are internally connected to a so called *dependability module*. This module is further connected to the upper layers or even directly to the application. Additional services are handled by a technology-specific module. Two physical interfaces are required in order to provide basic redundancy (denoted *prim* and *secn*). If the device represents a *routing node*, it contains an additional interface (denoted *conn*).

3.2 System model

A system adopting the GDL basically provides reliability, safety and security features. Regarding **reliability**, special network topologies are used. These topologies offer physical redundancy and allow continuous communication even if a fault occurs. The redundancy mechanism supports different topologies depending on the network structure of the underlying protocol. In order to increase the scalability and flexibility, networks can be composed of these network segments forming a so called *internetwork*. **Safety** is primarily achieved by the reliability mechanism. However, the system is also capable of detecting communication failures and allows the application to perform particular fail-safe actions such that catastrophic consequences to users or the environment can be prevented. In order to detect communication failures, a heartbeat mechanism is used between two communication partners which is further referred to as *high-level heartbeating*.

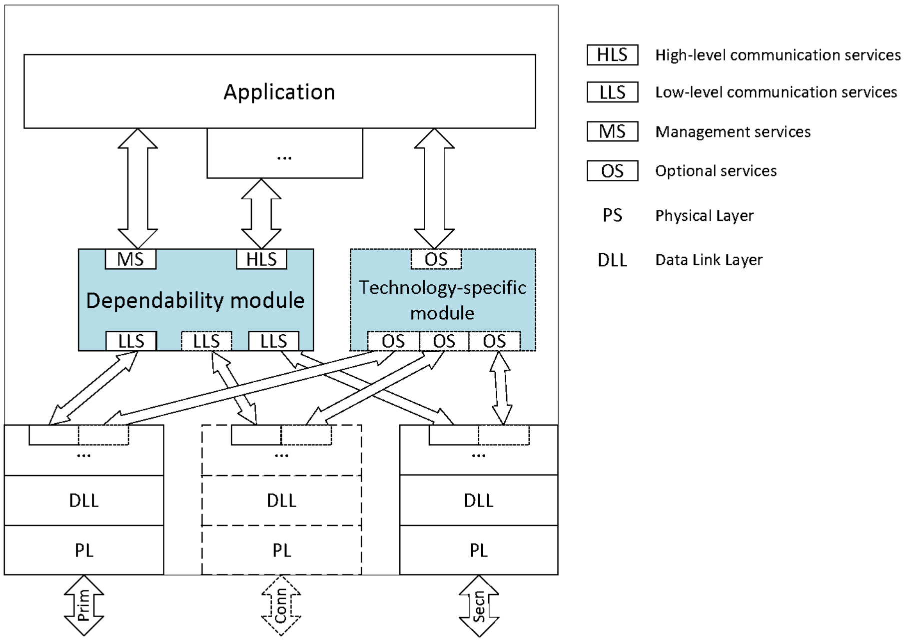

Figure 2: Device model

Besides that, a *high-level positive acknowledgment or retransmit* (PAR) protocol is applied. **Security** is achieved by addressing the following attributes: integrity can be guaranteed by applying a message authentication code to the messages. Additionally, messages are encrypted. Availability is addressed by cryptographic mechanisms in combination with the reliability mechanism. This means that security mechanisms ensure that only authenticated traffic passes a particular communication channel. If messages are modified or the channel is jammed (e.g., by a denial of service attack), the reliability mechanisms block the attacked channel and use the redundant connection.

As shown in Figure 3, the upper layer starts a transaction by generating a high-level data request. Subsequently, the GDL processes this request and executes dependability mechanisms. These mechanisms generate *high-level messages* which are transmitted by initiating *low-level data requests.* Upon reception of a message, the underlying layers generate a *low-level data indication.* Thereby, *high-level messages* are distinguished from *low-level messages.* High-level messages contain the payload of an end-to-end communication. They are initiated by a high-level request at the originator and are possibly forwarded by the GDL of a device on the route to its destination. Finally a high-level message triggers a high-level indication at its destination device. By contrast, low-level messages are used by the GDL to protect the communication channel and are not forwarded. For example, low-level heartbeats check whether the connection between two adjacent nodes is still working. The low-level acknowledgment ensures that selectively dropped high-level messages are detected (a missing low-level acknowledgment causes no retransmit, but causes the GDL to block the channel).

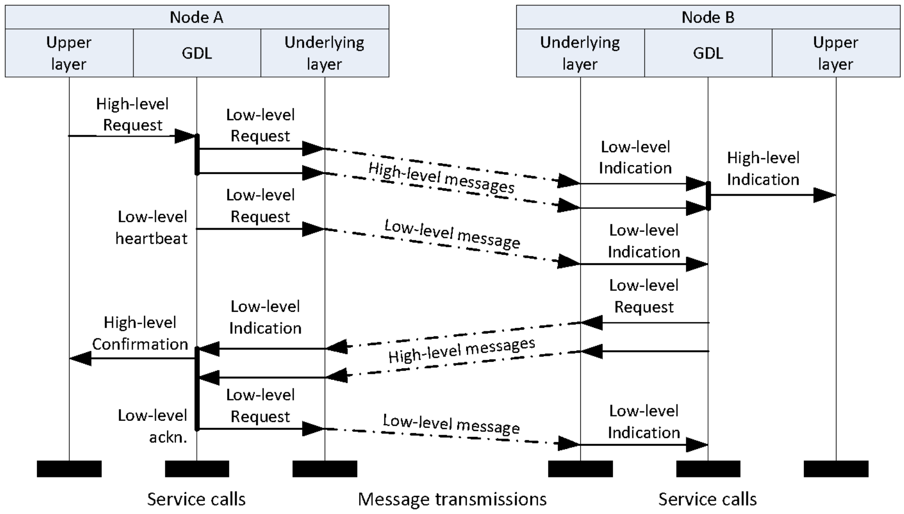

Figure 3: Communication model

3.3 Prerequisites

Although the proposed approach defines a "generic" intermediate layer, there are some requirements on the used communication technology. These requirements have to be satisfied, if the GDL shall be applied to a particular system.

- It is assumed that the underlying communication system separates its particular layers by clearly specifying the interfaces between them (i.e., communication services). The GDL requires basic communication services from the underlying protocol stack. Since the GDL allows transparent communication, it is necessary to provide all services to the upper communication stack. However, it is not required that the GDL uses all low-level services.

- The communication of the deployed system must be based on *stateless messaging*. Messages need to be independent of each other, which implies the use of connectionless communication services.
- Every device needs to be *uniquely identifiable* with any kind of address. The address can have any structure and any representation.
- The underlying communication system must support at least one of the following modes of connection: (1) A so called ***line*** connection links a group of nodes (i.e., at least two) such that pairwise bidirectional communication and *one-to-all* communication is possible. (2) A ***p2p*** connection, establishes a bidirectional link between exactly two nodes.

The overall system consists of arbitrarily many network nodes which are connected such that pairwise communication is possible. An internetwork consists of interconnected network segments, each of them following either a ring or redundant line topology. In such an automation system, the following basic faults can occur: (1) faults of links and (2) node faults.

The fault hypothesis basically covers one link fault per network segment, which means that the correct service of the whole system can be delivered even if one arbitrary link fails. Furthermore, it is assumed that the overall distributed application is capable of tolerating faults of nodes without influencing the overall function. More precisely, it is assumed that one node is allowed to fail in each network segment without influencing the residual network.

Link faults are subdivided into unintended faults and intended faults (i.e., security attacks). Basically, links have to tolerate security attacks, which may be either *preventable security* attacks or *non-preventable attacks*. Preventable attacks are interception or modification as well as fabrication without exhausting the channel or the system's performance. Non-preventable attacks are fabrication by exhausting the channel and interruption of the channel (i.e., denial-of-service attacks). While preventable attacks do not have to be considered further, since they are reliably handled by the security mechanism, non-preventable faults are mapped to fail-stop faults of a channel. This has to be implemented by detecting a particular attack and temporarily disabling the channel. Thereby, it is assumed that security attacks are detected.

3.4 Message types and structure

In order to implement the described mechanisms and services, different types of messages are required. Due to the generic nature of this approach, only the structure of the messages and the specific parts of the intermediate layer can be specified. Further details about the message structure depend on the underlying communication protocol. The basic message format is structured as illustrated in

Figure 4. The message starts with a protocol specific header which typically includes protocol information of the underlying layers such as flags and addresses. The header field is followed by a payload field which is basically used for carrying the encapsulated data of the upper layers (i.e., payload parameters). In some cases, communication protocols have a specific trailer containing a CRC for example. The GDL uses the payload field for carrying encapsulated and protected data of the upper layers and additionally for transmitting internal protocol information. Besides the message type, the header contains a Sequence Identifier (SI) and a Node Identifier (NI) which are both required to protect the message and provide data freshness. Subsequently, the message contains data of arbitrary length that depends on the particular message type. Finally, the field contains a Cryptographic Message Authentication Code (CMAC) to authenticate the message and guaranteeing integrity.

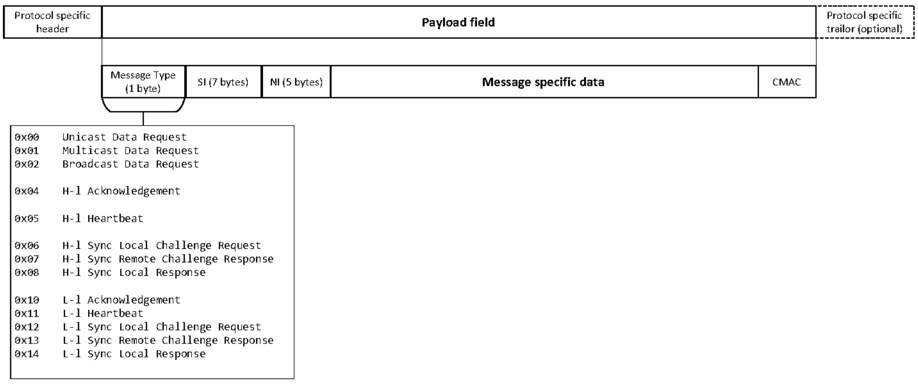

Figure 4: Generic message format

4 Dependability mechanisms

4.1 Safety

Safety strongly relies on reliability as it is generally assumed that ongoing communication is required to guarantee a safe system behavior. Nevertheless, the communication channels need to be supervised such that communication failures can be detected and fail-safe mechanisms can possibly be applied. Basically, these particular mechanisms follow the safety measures specified in openSAFETY with special respect to the overall system model and the integrative approach.

In order to allow safe communication, a special **message format** is specified. As the proposed approach represents an intermediate layer of the communication stack, a message contains the parameters of the calling service of the upper layer and a special header field. The physical message consists of the header of the underlying protocol, the dependability header, and the encapsulated payload of the upper-layer service call. The dependability header supports safety as well as reliability and security mechanisms. Besides the address fields (i.e., source and destination), the header contains a so called *Sequence Identifier (SI)* which uniquely identifies a message. This SI consists of a time stamp as well as two counter values: (1) The retransmit counter is only increased if a message is retransmitted. (2) The sequence counter increases on each message transmission and ensures that messages can be distinguished which are transmitted at the same instant of time. The cryptographic message authentication code (CMAC) represents the last part of the message and contains a message authentication code computed over the specific header and the payload field. Besides its security tasks, it ensures that corrupted messages can be safely detected.

As the GDL has no influence on the underlying communication system, messages may get lost or may be received in wrong order. Thus, the GDL contains a **message reordering** mechanism that is also able to detect message losses. For this purpose, the header of a high-level message contains an additional counter that is specific for each communication link. Additionally, a ring buffer is maintained for each communication link, where each element stores one incoming message. The size of the buffer is limited and corresponds to the length of the counter such that each message has a predetermined location in the buffer. As the buffer size is strongly limited, the reordering is only possible for a limited number of messages. If a message is delayed too much, a message loss is detected and the buffer is cleared.

Duplicate filtering takes place at two different locations of the GDL. On the one hand, it is used to prevent duplicate message deliveries to the application. On the other hand, it is applied at each communication interface to ensure that equal messages are not transmitted multiple times. The filtering mechanism is based on a data structure containing the SIs of previous messages. As the size of this data structure is limited, it cannot be guaranteed that all messages are filtered. Thus, the application shall be aware of multiple message deliveries (e.g., by using only state-based information).

Since messages within the network are forwarded or routed and the receiver may have failed, it has to be ensured that each message in the network is eventually dropped. For this purpose, a **time-to-live** counter is introduced in the dependability header. This counter is initialized a priori to a value reflecting the maximum number of hops which depends on the network size and topology.

A **positive acknowledgment and retransmission** mechanism is similarly adopted in many communication systems and ensures that lost messages are retransmitted after a given timeout. Due to the retransmission counter which is part of each message, even retransmitted messages are different. This counter is further required for passing the duplicate filtering mechanism.

The **heartbeating** mechanism is used to monitor the state of end-to-end communication channels. This mechanism ensures that messages are periodically exchanged over each communication channel. If no application message is sent for a predefined period of time, a heartbeat message is generated. This message has no particular content and is only used for checking the connectivity. If a node does not receive a message for a given period of time, a communication failure is assumed.

4.2 Reliability

In order to achieve reliability w.r.t. the fault hypothesis, a fault-tolerant approach is applied. Thereby, two basic physical topologies are facilitated: a *ring topology* and a *redundant line topology*. In contrast to arbitrary topologies (i.e., general graph), the complexity of these specific topologies is controllable. These basic topologies can be coupled such that large and flexible overall topologies can be set up.

4.2.1 Rings

A ring topology is the most efficient way for providing single-fault resilience. Thereby, the nodes within the ring are connected to their immediate neighbors by *p2p* links (Figure 5, top). For allowing pairwise communication, nodes need to forward messages. For this purpose an efficient forwarding mechanism is proposed in [Krammer, 2015]. This mechanism is based on an accurate determination of the link state which is performed by so called low-level heartbeat messages.

If a node needs to transmit a message, it sends it via one healthy interface. The header of the message contains a flag which indicates whether the direction of the message was previously changed. If the forwarded message is received by another node, it is first checked if the message is destined to itself. If not, the message needs to be forwarded. If the opposite interface is healthy, the message is forwarded independent of the flag. If the other interface is not healthy and the direction of the message was not changed previously, the message is forwarded via the incoming interface by setting the flag. If the opposite interface is not healthy and the flag is set, the message is dropped. This case can only happen if two different faults occur.

Since the interface of the outgoing message can be selected arbitrarily, different methods can be applied for balancing the communication load within the ring. Although a random or alternating selection would achieve this claim, the latency tolerance would increase due to a varying number of hops.

4.2.2 Redundant lines

In this topology, the nodes are connected by two disjoint and independent *line* connections as illustrated in Figure 5 (bottom). If a message needs to be transmitted, it is sent via both interfaces. Consequently, it arrives at its destination node even if one of the line connections partially or completely fails. Since each dependable node – independent of the type – implements a duplicate filtering mechanism, only the first arriving copy of the message is delivered.

4.2.3 Internetworking

In order to increase the flexibility and scalability of a network, the basic topologies (i.e., network segments) can be reliably connected. For providing a fault tolerant connection, two segments are coupled by two disjoint network nodes. In contrast to conventional dependable nodes, each of these nodes has three different network interfaces (Figure 5, middle). While one interface is connected to each network segment, the third interface is used to establish a connection with the second routing node. Although each type of routing nodes implements different forwarding mechanisms, the basic routing algorithm is the same. This algorithm decides whether to forward a message to the other network segment or not. This algorithm is based on the forwarding mechanism of Ethernet Switches [Tanenbaum et al., 2013].

With these routing nodes, almost arbitrary network topologies can be built. For example, one ring can be connected to multiple rings or redundant line segments. However, a redundant line segment can only be connected to two other segments (ring or redundant line). However, it has to be ensured that network segments do not form loops.

In order to provide a fault tolerant connection between two network segments, two routing nodes are required. Depending on the type of connection (i.e., ring $\leftrightarrow$ ring, redundant-line $\leftrightarrow$ redundant-line, ring $\leftrightarrow$ redundant-line), different mechanisms are applied as defined in [Krammer, 2015]. These mechanisms ensure that the fault of one node of a pair of routing nodes or the connection among the routing nodes can fail without interrupting the connection between the connected network segments.

4.3 Security

The security mechanisms of this approach reside on two pillars. At low-level, security is used to protect the point-to-point communication which is necessary

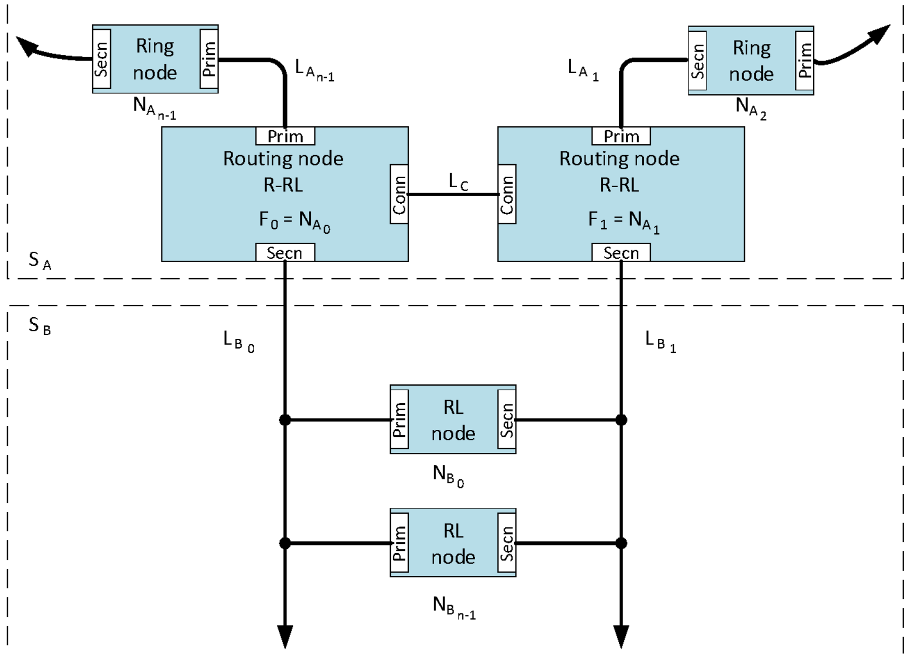

Figure 5: Routing nodes between ring and redundant line

for link-state determination. At high-level, a secure communication channel is established among communicating peers. In both cases, the protection of messages is based on symmetric and well-established security algorithms. The basic cipher that is used by the proposed concept is AES128. This symmetric block cipher was chosen as it is a commonly used state-of-the-art algorithm. There exist very efficient hardware and software implementations. The block length was chosen as trade-off between performance and security level. In order to authenticate and encrypt arbitrary messages, CCM (cf. [NIST, 2005a, IETF, 2003a]) is used as *mode of operation*. An important reason for choosing this mode in combination with AES is that its security was proven (cf. [Jonsson, 2003]).

This security mechanism requires shared symmetric keys. For this purpose, keys can either be assigned statically during engineering or exchanged between peers (high-level security) or adjacent nodes (low-level security). Key management in distributed automation systems is a challenge on its own (cf. [Treytl and Sauter, 2015]). Therefore, selection as well as exchange of keys are out of scope of this article and left for future work.

The cryptographic nonce ensures that messages with the same plain-text and encrypted with the same key result in different cipher-texts. Thus, the nonce has to satisfy the condition that no message is retransmitted with the same key and the same nonce [IETF, 2003a]. The nonce contains two main parts: (1) the sequence identifier (SI) and (2) the node identifier (NI). While the SI guarantees uniqueness of a message within a node, the NI guarantees uniqueness of nodes in the network. The SI of high-level and low-level communication are independent. Also, the NI is specified differently for high-level and low-level messages.

In order to guarantee secure communication, the nonce needs to be synchronized. Since the NI is predetermined for each node and does not change over time, only the SI has to be synchronized. This synchronization has to be performed at low-level between adjacent nodes sharing a *p2p* connection and at high-level among communicating peers. The synchronization is based on a mutual challenge response approach.

4.3.1 Low-level security

Low-level security mechanisms are basically used to protect the reliability mechanism relying on the low-level heartbeat mechanism. In order to protect heartbeat messages from injection and manipulation, they are protected with the aforementioned cryptographic mechanism. However, an attacker is able to partially drop messages such that heartbeat messages can pass and only high-level messages are blocked. In this case, it would not be possible to detect a link fault and consequently the reliability mechanism on the ring would fail. To prevent this, high-level messages are acknowledged. A failed low-level acknowledgment does not trigger a retransmit, but causes the particular node to set the interface to *failed*, even if low-level heartbeat messages are received successfully. This forces the reliability mechanism to perform a fail-over and consequently no high-level message will be sent via this attacked link.

The NI for low-level messages needs to be unique for each *p2p* interface of a node. As this cannot be required from the underlying communication stack, a random number is selected. This number is generated once during the synchronization process.

4.3.2 High-level security

The high-level security mechanisms are used to establish a secured channel among the different nodes. Thereby, the integrity of messages is checked and the payload (i.e., the whole encapsulated frame) is encrypted. Furthermore, data freshness is guaranteed and lost messages can be detected.

The NI is basically adopted from the address of the node. Due to the generic nature of this approach, the length of this address is unlimited. Thus, the definition of the NI is distinguished. If the size of the address is smaller than 40 bits, the address is directly adopted and padded if necessary. Otherwise, a compression function is applied to the address based on an AES chain as introduced in [Krammer, 2015]. In this case, the uniqueness relates to a specific likelihood. Upon reception of a message, the received SI is checked to identify whether the message was replayed or significantly delayed.

5 Theoretical evaluation

In order to analyze the behavior of the presented approach, different methods are applied. In a first step, the dependability properties were theoretically evaluated. In a second step, the dynamic behavior of such a system was evaluated by using simulation.

5.1 Static fault analysis

The theoretical analysis took a closer look on the fault-tolerance properties of the proposed approach. It was proven that messages sent by any correct node within a network eventually arrive at least once at an arbitrary destination node even in case of one fault per network segment (ring or redundant line). In a first step, this was proven for ring and redundant line networks consisting of an arbitrary number of nodes. In a second step it was proven, that the assumption even holds in internetworks. For this purpose, each combination of two basic topologies was investigated. Finally, it can be argued that there is no difference between messages originated at a particular node and messages that are solely forwarded by this particular node. This allows to extend the proof to arbitrary connections of ring and redundant line segments.

In the analysis, only static topologies were considered. The analysis proved the correct transmission of a message between arbitrary nodes (N_s, N_r) of the particular topology which is denoted as *transaction*. A transaction is initiated by a high-level request at N_s and is finished after a high-level indication at N_r. Thereby, all types of faults are considered which are covered by the fault hypothesis. However, faults were assumed to be static which means that they occur before the transaction starts and do not change during the transaction. Furthermore, it was assumed that nodes have determined the link state of p2p connections correctly, before the transaction was initiated. All proofs were constructed to show the eventual arrival of at least one message instance at the destination node. Due to the local filtering mechanism, it is ensured that no duplicate message is delivered, if the SI buffer is sufficiently large.

The theoretical analysis was based on proving theorems addressing the two proposed topologies with arbitrary nodes. Additionally, the interconnection of any two topologies with arbitrary size was shown. The proofs are mainly based on deduction and induction in ring-based topologies [Krammer, 2015].

5.2 Security

In contrast to IT systems, automation systems have to deal with different security threats. [Granzer et al., 2006] states that a BAS is exposed to different types of security threats, whereas only network attacks are relevant for this generic communication scheme.

Interception is prevented by establishing a secured channel among communicating nodes, as previously shown. Thereby, data is transmitted in a confidential and authentic way by using strong and standardized security mechanisms.

Due to the use of a CMAC, only entities that own the shared key are able to generate correct messages and thus **modification** is not possible. Without the knowledge of the key, an entity is not able to compute a valid CMAC based on given input data or to modify a message such that it matches with a valid CMAC.

Due to appending a CMAC, only nodes possessing the secret key are able to generate messages. Thus, **fabrication** is not possible by other entities as they are not able to create valid messages that pass the integrity check. If messages are retransmitted, they pass the integrity check, but they are rejected after the check of the SI and the NI. In case of low-level messages, sending nodes do not authenticate themselves with an individual address, and receiving nodes do not know the expected neighbor. Thus, random numbers are used to mutually authenticate adjacent nodes. These random numbers are exchanged during the synchronization of the SI. However, an attacker can harm the system if messages are redirected through an auxiliary channel such that non-adjacent nodes establish a connection. Then an attacker is able to impersonate a correct node and redirect heartbeat messages. However, due to the low-level acknowledgment mechanism, the impact of this vulnerability does not influence the overall security.

Due to the reliability mechanism, any kind of **interruption** attack on one channel can be covered by initiating a fail-over and deactivating the attacked connection. The detection is based on monitoring the rate[1] of unsecured or replayed messages or messages with different source or destination addresses. Due to the security mechanisms based on individual addresses and SI/NI, it is not possible to generate valid messages without the knowledge of the key (not even by wormhole attacks). Additionally, heartbeat messages and acknowledgment messages are used to detect interruptions on p2p channels. After a fail-over, it is necessary to determine whether the interruption was removed from the blocked channel to continue with normal operation. However, this can only be done by re-enabling the channel after a predefined period of time. If the security attack is still active, a few messages may get lost, before the attack is detected and the

[1] needs to be individually adjusted for a particular protocol depending, for instance, on the bandwidth

channel is blocked. However, the failed transmissions are covered by the PAR mechanism.

A **man-in-the-middle** can basically perform all the previously mentioned attacks such as interception, modification, fabrication or interruption. The GDL is able to resist all these attacks except fabrication. Messages can successfully be fabricated during the low-level synchronization by using valid messages of other links. Thus, high-level and low-level messages can be redirected by a man in the middle.

Since the GDL uses only one shared key, a **compromised key** compromises the whole network. With this key, an attacker can arbitrarily generate and modify messages in the network. If the application applies sophisticated security mechanisms, the impact of a compromised key can be reduced. Thereby, even *perfect forward secrecy* [Menezes et al., 1996] can be achieved.

6 Dynamic analysis

In addition to a theoretical analysis, simulation was used to evaluate the dynamic behavior and the performance of the proposed approach. The proofs introduced in the previous section showed that the mechanism behaves correctly in the presence of one fault per network segment. However, it was assumed that a fault does not appear or disappear during a transaction. The simulation evaluates how the mechanism behaves in presence of frequently changing (appearing/disappearing) faults.

6.1 Simulation model

For this simulation task, OMNeT++ (cf. [Varga et al., 2001]) was chosen. According to the proposal of the concept, two different types of dependable nodes and three different types of routing nodes were modeled. Due to the generic nature of the proposed concept, no details about the deployed communication system and its properties are known. Therefore, only one network specific parameter is considered: the *transmission delay.*

In the simulation model, two types of messages were taken into account: one for low-level communication and one for high-level communication. For both message types, a *message type* field, an *SI* field and a *payload field* were defined. In addition to these common parameters, high-level messages contain fields for *source and destination address.*

6.2 Parameters and metrics

The basic parameter of a communication system is the *transmission delay.* It is the only parameter that reflects the properties of the underlying communication system. This parameter subsumes all properties of the transmission channel (e.g., operating system, communication medium, routing devices). Thus, it is the only time base for the simulated system, and all other temporal parameters rely on this value. Due to the generic nature of this approach, no specific behavior of the communication channel can be assumed and thus no value or distribution can be adopted for modeling the transmission delay. In the simulation model, a truncated normal distribution is selected. Instead of this, any other distribution can be used that is bounded in both directions (e.g., truncated Poisson distribution), as the mechanisms as well as the results of the simulation are not influenced by the selection of the transmission delay. However, the use of a constant value is omitted for almost any simulation parameter in order to prevent *repetitive behavior* of the simulation meaning that if two or more periodic parameters of a system are set to fixed values, the scenario would repeat after a certain interval.

To prove the behavior of the system, high-level channels are established, where messages are exchanged with a constant interval. Depending on the topology and simulation scenario, different communication channels with different intervals are defined. Due to the generic nature of this approach, no information about the message interval can be adopted. In this model, the message interval is based on a truncated normal distribution similar to the transmission delay.

A further parameter set addresses the PAR mechanism. Thereby, two parameters are considered: the *maximum number of retransmissions* and the *retransmission timeout*. While the retransmission timeout can be set depending on the upper bound of the transmission delay and the maximum number of hops between two communicating nodes, the number of retransmits can be set arbitrarily.

As previously introduced, the simulation aims at analyzing the behavior of the presented approach in case of transient faults. The *fault interval* basically reflects the frequency of fault changes, and represents the main simulation parameter. In the simulation studies, this parameter is varied to adopt a threshold between a correctly working system and a faulty one.

In contrast to redundant line topologies, all nodes that are connected to or part of a ring topology need to determine the link state of p2p channels by using are the low-level heartbeat mechanism. The mechanism is based on a parameter set consisting of the *HearbeatInterval* and the *HeartbeatTimeout*. The *HearbeatInterval* specifies the period in which the heartbeats are transmitted and reflects the accuracy of the fault detection. The *HeartbeatTimeout* reflects the maximum time between two received heartbeats. This parameter depends on the *HearbeatInterval* and the *TransmissionDelay*. As this parameter set strongly influences the behavior of the system and thus also the threshold of the fault interval, it is varied in those scenarios, where link state estimation is required. The value of the *HearbeatInterval* is varied and the *HeartbeatTimeout* is set accordingly.

As basic measure for the quality of a transmission, the *retransmission count* is used. If no retransmit is necessary, the quality of the transmission is good. If the last retransmit attempt is successful, the transmission is still successful, but many messages were lost or significantly delayed on the channel. Thus, the number of necessary retransmits until the message is successfully delivered is used as metric for the simulation. This is also a discrete indicator for the response time of a system, as messages are retransmitted after constant time intervals. Basically, the retransmit count can only assume discrete values between 0 and $maxRtr$. However, it is further necessary to indicate whether a message transmission failed. Therefore, a monotonic scale is defined as follows:

$$\begin{aligned} m &= (maxRtr + 1) - RtrCnt \text{ , if } RtrCnt \leq maxRtrCnt \\ &= 0 \qquad\qquad\qquad\qquad\quad \text{ , if transmission failed} \end{aligned}$$

Thereby, $RtrCnt$ represents the number of retransmissions and $maxRtr$ the maximum number of retransmissions. Thus, m is set to $m = maxRtr + 1$, if no retransmit is necessary. If the last possible retransmit attempt is successful, m is set to 1. Consequently, $m = 0$ indicates a failed transmission.

In order to get a single significant parameter that primarily reflects the question to the simulation, the particular results of the transmissions are aggregated by using a minimum function: $M_{min} = \min(m_i)\ \forall m_i$.

6.3 Scenarios

In order to analyze the behavior of the approach in presence of dynamic faults, different simulation studies were performed. Basically, five network structures were taken into account. These topologies result from the two basic topologies and the combination of them. A ring topology as well as a pure redundant line topology were analyzed first. Then, the combination of two rings and two redundant line structures as well as the combination of a ring with a redundant line structure were examined.

In each of the following simulation studies, the nominal value of the transmission delay was set to $1ms$. As this parameter is the only direct relation to the simulation time, it can be set arbitrarily, whereas all other temporal parameters are relying on it. The retransmit timeout was set to $100ms$, reflecting the maximum number of hops. For the maximum number of retransmits, 10 was chosen. The interval of high-level message transmissions follows a truncated Gaussian distribution and was individually set for each connection. The mean values of these time intervals were set such that they are co-prime (w.r.t. a unit of $100\mu s$) to all other set parameters and to each other transmission interval. In particular, they were varied between $70.9ms$ and $140.9ms$. The particular high-level channels were specific for each network topology and simulation scenario.

6.4 Results

The results of the simulation show that the efficiency of the proposed mechanism depends on the heartbeat interval and the fault interval. These parameters are directly depending on the transmission delay, due to the discrete event simulation. Other parameters such as the maximum number of retransmissions and the retransmission delay do not influence the result if they are selected sufficiently high. The particular values of these parameters strongly depend on the network topology (i.e., number of hops). Other parameters strongly depend on the properties of a particular communication technology and cannot be considered in a generic simulation.

Figure 6 shows an example of a simulation result. The diagram depicts a colored map and illustrates the worst case results of each simulation run. There, the metric M_{min} is represented by a color scale in a two dimensional plane, where the X and Y axis represent the fault interval and the heartbeat interval. The color represents the quality of the worst transmission of a particular parameter set. A deep red color denotes a failed simulation run whereas yellow and green shades represent medium and good quality, respectively. Thus, the diagram illustrates a border between failed and non-failed runs w.r.t. a particular retransmission count.

Summing up the results of the evaluation, it can be concluded that a system implementing the proposed concept is capable of tolerating one crash fault in each network segment as claimed by the fault hypothesis. However, reliable communication is not possible in any physical system, if the fault interval is unbounded (i.e., frequently appearing and disappearing faults). If the heartbeat interval is increased above a certain threshold, data requests fail independent of the fault interval. This is because heartbeats determine the network state in ring based segments and the reliability mechanisms depend on this information. If the heartbeat interval is too large, all messages of a transaction (i.e., original message, retransmits and acknowledgment messages) may be forwarded depending on a wrong network state. This is also the case when the fault interval of a particular node is shorter than the heartbeat interval.

7 Conclusion

In this article, a comprehensive dependability approach for building automation networks was presented covering safety and reliability as well as security attributes. Due to its generic nature, the proposed concepts allows an integration into existing installations taking into account low demands regarding computational power of available equipment typically found in this domain. The correctness of the approach was shown in a theoretical evaluation, while the dynamic behavior was analyzed using simulation. In general, it needs to be admitted that

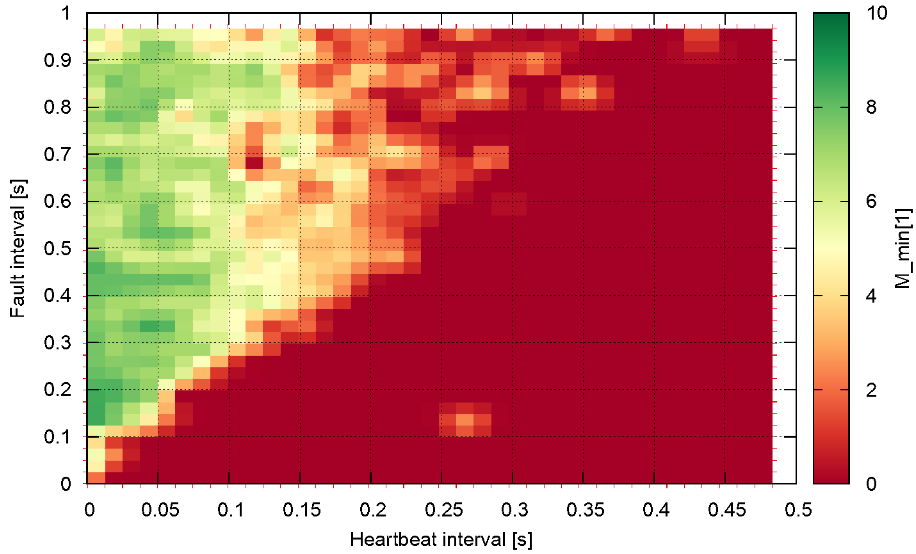

Figure 6: Example of a simulation result

dependability causes significant overhead in a communication system. Besides the need of security algorithms (and thus extra computational power), additional data need to be transmitted. Further header information is just as necessary as auxiliary or redundant messages. When applying this approach to a specific technology, these issues have to be considered. During deployment, communication parameters have to be chosen carefully depending on the selected communication technology. Hereby, the dynamic evaluation serves as a first guideline.

In a further step, a new device type can be defined that even allows interoperability between dependable and conventional devices. Such a device class would need to address safety as well as security issues. For addressing maintainability aspects, the present concept can be extended with a framework that supports device management and replacement. Also tool support for engineering and provisioning may be included for future work. Although key management was not considered in this article, this is an important aspect of an automation system and needs to be addressed. However, it has to be noted that key management depends on the kind of application of the communication system and the related use cases.

Due to the increasing importance of the Internet of Things (IoT) even in the BA domain, also dependability needs to be addressed in this context as introduced in [Frühwirth et al., 2015]. Thereby, our next steps will also be directed how how the presented approach fits in such environments.

References

[Avizienis et al., 2001] Avizienis, A., Laprie, J.-C., Randell, B., et al. (2001). *Fundamental concepts of dependability.* University of Newcastle upon Tyne, Computing Science.

[Avizienis et al., 2004] Avizienis, A., Laprie, J.-C., Randell, B., and Landwehr, C. (2004). Basic concepts and taxonomy of dependable and secure computing. *Dependable and Secure Computing, IEEE Transactions on*, 1(1):11–33.

[ETSI, 2008] European Telecommunications Standards Institute (ETSI) (2008). TS 155 226 – Digital cellular telecommunications system (Phase 2+); 3G Security; Specification of the A5/4 Encryption Algorithms for GSM and ECSD, and the GEA4 Encryption Algorithm for GPRS.

[Frühwirth et al., 2015] Frühwirth, T., Krammer, L., and Kastner, W. (2015). Dependability Demands and State of the Art in the Internet of Things. In *Proceedings of the 20th IEEE Conference on Emerging Technologies and Factory Automation (ETFA)*, pages 1–4, Luxembourg.

[Granzer et al., 2006] Granzer, W., Kastner, W., Neugschwandtner, G., and Praus, F. (2006). Security in networked building automation systems. In *Proceedings of the IEEE International Workshop on Factory Communication Systems (WFCS)*, pages 283–292.

[IEC, 2007] International Electrotechnical Commision (IEC) (2007). 62347 – Guidance on system dependability specifications.

[IEC, 2010a] International Electrotechnical Commision (IEC) (2010a). 60300 – Dependability management.

[IEC, 2010b] International Electrotechnical Commision (IEC) (2010b). 61784-3 Industrial communication networks – Profiles – Part 3: Functional safety fieldbuses.

[IEC, 2010c] International Electrotechnical Commision (IEC) (2010c). 62439-2 High Availability Automation Networks - Media Redundancy Protocol (MRP).

[IEC, 2011a] International Electrotechnical Commision (IEC) (2011a). 61508 – Functional Safety of Electrical/Electronic/Programmable Electronic Safety-related Systems.

[IEC, 2011b] International Electrotechnical Commision (IEC) (2011b). 62439-7 Industrial communication networks - High availability automation networks - Part 7: Ring-based Redundancy Protoco (RRP).

[IETF, 1992] Internet Engineering Task Force (IETF) (1992). RFC1321 – The MD5 Message-Digest Algorithm.

[IETF, 2003a] Internet Engineering Task Force (IETF) (2003a). RFC3610 – Counter with CBC-MAC (CCM).

[IETF, 2003b] Internet Engineering Task Force (IETF) (2003b). RFC3619 – Extreme Networks' Ethernet Automatic Protection Switching (EAPS) Version 1.

[IETF, 2003c] Internet Engineering Task Force (IETF) (2003c). RFC4345 – Improved Arcfour Modes for the Secure Shell (SSH) Transport Layer Protocol.

[Jonsson, 2003] Jonsson, J. (2003). 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. Springer-Verlag.

[Judmayer et al., 2014] Judmayer, A., Krammer, L., and Kastner, W. (2014). On the security of security extensions for IP-based KNX networks. In *Proceedings of the 10th IEEE Workshop on Factory Communication Systems (WFCS)*, pages 1–10.

[Kastner et al., 2005] Kastner, W., Neugschwandtner, G., Soucek, S., and Newmann, H. (2005). Communication systems for building automation and control. *Proceedings of the IEEE*, 93(6):1178–1203.

[Kastner and Novak, 2009] Kastner, W. and Novak, T. (2009). Functional safety in building automation. In *Proceedings of the IEEE Conference on Emerging Technologies Factory Automation (ETFA)*, pages 1–8.

[Kopetz et al., 2005] Kopetz, H., Ademaj, A., Grillinger, P., and Steinhammer, K. (2005). The time-triggered Ethernet (TTE) design. In *Object-Oriented Real-Time Distributed Computing, 2005. ISORC 2005. Eighth IEEE International Symposium on*, pages 22–33.

[Krammer, 2015] Krammer, L. (2015). *Dependability in building automation networks.* PhD thesis, TU Wien.

[Krammer et al., 2013] Krammer, L., Bruyne, S. D., Kastner, W., and Granzer, W. (2013). Security Erweiterung fuer den KNX Standard. In *Tagungsband – innosecure 2013 – Kongress mit Ausstellung fuer Innovationen in den Sicherheitstechnologien Velbert Heiligenhaus*, pages 31–39. german.

[Krammer et al., 2016] Krammer, L., Kastner, W., and Sauter, T. (2016). A generic dependability layer for building automation networks. In *IEEE World Conference on Factory Communication Systems (WFCS)*, pages 1–4.

[Krammer et al., 2012] Krammer, L., Klein, M., Kasberger, J., and Kastner, W. (2012). A fault-tolerant Communication Scheme for Fire Alarm Systems based on KNX. In *KNX Scientific Conference*.

[Laprie, 1992] Laprie, J. (1992). Dependability: Basic Concepts and Terminology. In Laprie, J., editor, *Dependability: Basic Concepts and Terminology*, volume 5 of *Dependable Computing and Fault-Tolerant Systems*, pages 3–245. Springer Vienna.

[Menezes et al., 1996] Menezes, A., van Oorschot, P., and Vanstone, S. (1996). *Handbook of Applied Cryptography.* Discrete Mathematics and Its Applications. Taylor & Francis.

[Mentzel, 2010] Mentzel, M. (2010). Funktionale Sicherheit mit SAFETYLON. Technical report, LonMark Deutschland. german.

[NIST, 1999] National Institute of Standards and Technology (NIST) (1999). FIPS PUB 46-3 – Data Encryption Standard (DES).

[NIST, 2001a] National Institute of Standards and Technology (NIST) (2001a). FIPS PUB 197 – Advanced Encryption Standard (AES).

[NIST, 2001b] National Institute of Standards and Technology (NIST) (2001b). NIST SP 800-38A – Recommendation for Block Cipher Modes of Operation: Methods and Techniques.

[NIST, 2005a] National Institute of Standards and Technology (NIST) (2005a). NIST SP 800-38C – Recommendation for Block Cipher Modes of Operation: The CCM Mode for Authentication and Confidentiality.

[NIST, 2005b] National Institute of Standards and Technology (NIST) (2005b). NIST SP 800-67 – Recommendation for the Triple Data Encryption Algorithm (TDEA) Block Cipher.

[NIST, 202] National Institute of Standards and Technology (NIST) (202). FIPS PUB 180-2 – Secure Hash Standard (SHS). suspended by NIST in 2008.

[Novak and Tamandl, 2007] Novak, T. and Tamandl, T. (2007). Architecture of a Safe Node for a Fieldbus System. In *Proceedings of the 5th IEEE International Conference on Industrial Informatics (INDIN)*, volume 1, pages 101–106.

[Rivest, 1992] Rivest, R. L. (1992). The RC4 encryption algorithm. *RSA Data Security Inc.*

[Sauter et al., 2011] Sauter, T., Soucek, S., Kastner, W., and Dietrich, D. (2011). The Evolution of Factory and Building Automation. *Industrial Electronics Magazine, IEEE*, 5(3):35–48.

[SAE, 2011] Society of Automotive Engineers (SAE) (2011). AS6802 – Time-Triggered Ethernet.

[Tanenbaum et al., 2013] Tanenbaum, A., Wetherall, D., and Pearson (2013). *Computer Networks: Pearson New International Edition.* Pearson custom library. Pearson Education, Limited.

[Treytl and Sauter, 2015] Treytl, A. and Sauter, T. (2015). Hierarchical key management for smart grids. In *2015 IEEE International Symposium on Systems Engineering (ISSE)*, pages 496–500.

[Treytl et al., 2005] Treytl, A., Sauter, T., and Schwaiger, C. (2005). Security measures in automation systems-a practice-oriented approach. In *Proceedings of the 10th IEEE Conference on Emerging Technologies and Factory Automation (ETFA)*, volume 2, pages 9 pp.–855.

[TTTech, 2010] TTTech Computertechnik AG (2010). *TTEthernet – A Powerful Network Solution for All Purposes*. Vienna, Austria.

[Uzunov et al., 2012] Uzunov, A. V., Fernandez, E. B., and Falkner, K. (2012). Engineering security into distributed systems: A survey of methodologies. *Journal of Universal Computer Science*, 18(20):2920–3006.

[Varga et al., 2001] Varga, A. et al. (2001). The OMNeT++ discrete event simulation system. In *Proceedings of the European Simulation Multiconference (ESM)*, volume 9, page 185.