Eur. Phys. J. Plus (2024) 139:783
https://doi.org/10.1140/epjp/s13360-024-05583-4

THE EUROPEAN
PHYSICAL JOURNAL PLUS

Regular Article

Critical infrastructure monitoring in CBRNe scenarios: a reliable and robust communication network for distributed multimodal sensors

Sebastian Sporrer[a], **Norman Niemann**[b], **Christof Hammer**[c]

Institute for the Protection of Terrestrial Infrastructures, German Aerospace Center (DLR), Rathausallee 12, 53757 Sankt Augustin, North Rhine-Westphalia, Germany

Received: 22 February 2024 / Accepted: 22 August 2024

Abstract The malicious misuse of CBRNe agents can inflict extensive damage to critical infrastructure and terrorize public society. The effects of such attacks can range from financial or structural drawbacks to significant casualties. Recent terror attacks in Cucuta (2021) or Kabul (2021) demonstrate the need to protect infrastructures such as airports, power plants, and transportation infrastructure. In the face of these modern-day threats, the need for a reliable and effective monitoring system for critical infrastructure has become increasingly important. In a first contribution to the long-term development, we present an initial version of a concept that requires the implementation paradigm of diverse redundancy and secure communication using MQTTS relying on the included cipher suite. An initial set of node types is defined and assigned requirements for implementation. The software tools architecture we propose based on these requirements is designed to support network operators and developers by providing standard features for network management and sensor node implementation. This includes the definition of a secure semi-automatic onboarding process for new sensor nodes, which is presented in detail. We strive for independency from specific hardware platforms, software frameworks, and network technologies to enable an open standard for communication within a critical infrastructure sensor network and also between such networks in the future.

1 Introduction

The prosperity of society strongly depends on multiple critical infrastructures (CI). The German Federal Office of Civil Protection and Disaster Assistance defines the term CI as

> "[...] organizations and facilities with important significance for the state community, the failure or impairment of which would result in lasting supply bottlenecks, significant disruptions to public safety or other dramatic consequences [1]."[1]

The German Federal Ministry of the Interior and Community and the German Federal Office for Information Security identified ten sectors of CI [1, 2], as shown in Fig. 1. CIs are exposed to various natural and anthropogenic threats, like earthquakes, floods, or terrorist attacks. Due to inter-dependencies between some sectors of CI (e.g., information technology and telecommunications toward energy or food toward water), there is a high risk of domino or cascade effects, where the breakdown of a single infrastructure in one sector will have a negative impact on one or many infrastructures of other sectors [1]. Particularly in connection with technological developments and ubiquitous digitization, a further increase in these dependency relationships is assumed in the future [3]. Protecting these infrastructures is therefore a core task and a central issue of German security policy [1].

The rapid proliferation of sensor networks due to the Internet of Things (IoT) and Industrie 4.0 technologies has revolutionized the industry and our daily lives by enabling seamless data exchange and real-time monitoring and control. However, this exponential growth has led to numerous challenges regarding the reliability and security of these networked systems. The following briefly summarized works identifies these challenges. Sarwat et al. identify multiple research challenges for a connected smart city, especially related to CI. They not only point out the lack of vendor-independent standardization for communication between multiple CI networks, but also highlight the need to address cyber-physical security due to the recent increase in cyberattacks on various CIs, such as power grids [4]. In concluding their overview of security for IoT-based CI, Liu et al. discuss future research opportunities. On top of an increased need to study vulnerabilities and resulting risks, they note that it is critical to develop defensive schemes to

[1] Translated by the authors of this work.

[a] e-mail: sebastian.sporrer@dlr.de (corresponding author)
[b] e-mail: norman.niemann@dlr.de
[c] e-mail: christof.hammer@dlr.de

Published online: 02 September 2024

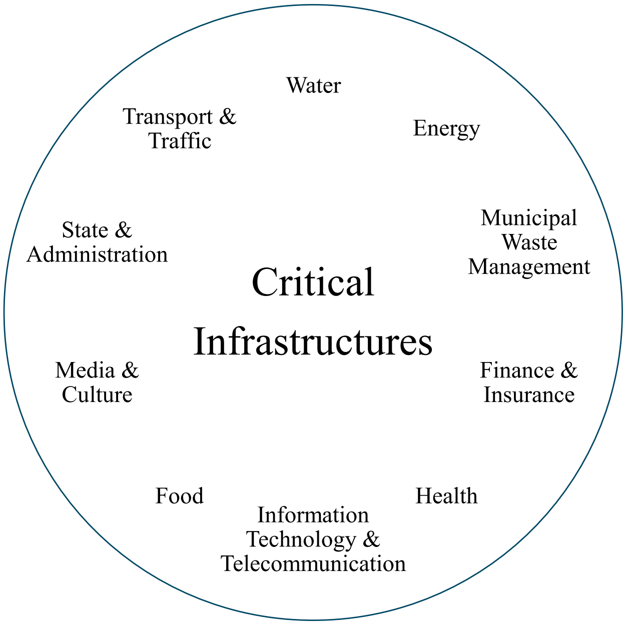


Fig. 1 Sectors of CI in Germany, as identified by the German Federal Ministry of the Interior [1] and Community and Section 2 (10) Sentence 1 BSI Act [2]

protect against cyberattacks. Such a defensive scheme can be characterized by a number of processes, for example the design of resilient IoT-based systems and the study of optimal IoT-based system configurations [5]. With regard to IoT technology reliability, Moore et al. have reviewed a wide range of works and derived various research directions to improve reliability. Among other things, they illustrate that the ideal reliability solution should be independent of specific hardware platforms, software frameworks, and communication protocols which change rapidly as they evolve. Additionally, they conclude that full stack awareness through the development of an end-to-end reliability solution has the potential to significantly improve the overall reliability of an IoT system [6]. Wisniewski et al. identify various future research directions as a result of their systematic literature review on the impact of Industry 4.0 solutions on CI safety and protection. Besides standard application programming interfaces for developers and network technologies, the general topics of security of industrial control systems and reliability and resilience of CIs are mentioned [7]. Villar Miguelez et al. compare different low-power wide area network (LPWAN) solutions in terms of providing a framework to digitize CI based on Industry 4.0 established standards. They conclude that further advances in standardization are needed [8].

The goal of this work is to contribute to the long-term development of a reliable and secure sensor network concept that enables the detection of chemical, biological, radiological, nuclear, and explosive (CBRNe) substances as a potential threat to CI through distributed multimodal detection systems. Within this context, initial considerations on functional safety, especially at the topological level, are formulated as requirements for the network units and introduced into the concept. In addition to the concept itself, the architecture of multiple software tools to support the implementation and management of such a sensor network are described as part of this article. Due to the high importance of data integrity and data protection, the use of current standard ciphers for the encryption of sensor data and the authentication of network entities is an essential part of our work. We strive for a concept that is as independent as possible of specific hardware platforms, software frameworks, and network technologies. However, the results presented in this paper are not to be understood as a fully comprehensive standard, but rather as a basis for further development of such a standard. Our contributions can be summarized as follows:

- An initial set of requirements for a robust sensor network regarding functional safety and information security with a focus on availability, authenticity, and cryptography is proposed.
- The architecture of software tools to validate the proposed requirements and to provide a minimal standard of data transmission in terms of the proposed concept is introduced.
- A secure onboarding process to add new sensor nodes semi-automatically to a network of the proposed kind is defined.

During our literature research, we identified the following closely related works, which are introduced briefly in this paragraph. We especially worked out the respective differences against our own contributions. Gomez and Ulmer propose an architecture for a secure sensor network to protect CI. The application scenario for their work is the surveillance of a stadium utilizing a wireless sensor network (WSN) with sensors to measure the acceleration, noise, and temperature in critical areas of the infrastructure. Their focus lies on the whole data transportation chain from data acquisition on sensor nodes to its delivery to the monitoring application software of the decision makers. Their proposal does not cover considerations regarding availability and reliability of the sensor network. Regarding security, part of their concept is the usage of a symmetric cipher only. The authenticity of the delivered data is evaluated based only on the trustworthiness of noise and acceleration information of the used sensors [9]. The symmetric cipher itself does not cover any measures to authenticate an entity of the network and also bears additional requirements concerning private key distribution and management. Their approach to authenticate valid data using measures based on expected sensor values can be used to identify faulty behavior of sensors but is not an adequate measure for authentication in an information security context. Berizzi et al. propose a heterogeneous sensor network for the protection of CI. Their design consists of four subsystems for sensing physical threats, information security threats, recovery planning, and central data management. The sensor network they describe is more of a security system with defined sensor modalities, surveillance of cyberattacks, and corresponding recovery rules rather than a concept or architecture for a sensor network infrastructure itself [10]. Grilo et al. propose a supervisory control and data acquisition (SCADA) system based on IoT technology. They identify the benefits of integrating wireless sensor and actuator networks (WSAN) with classic SCADA systems and then show an exemplary implementation to monitor a power grid. While redundancy is mentioned as a key feature of WSANs, no specific requirements are derived addressing the availability of the network. In the area of communication security, a symmetrical cipher is used in the network, which is part of the data transmission protocol also co-developed by Grilo. Authenticity of the network members is not part of their concept. They state that a standardized cipher suite should be part of the security concept, but do not go into further detail [11–13]. In their work regarding CI surveillance with secure WSNs, Niedermeier et al. describe their implementation of such a network with a focus on a unique security concept, which includes security on the communication level and mechanisms that ensure functional safety during its operation. The requirements identified with respect to the latter topic focus on the reliability of the sensor nodes. The availability of the nodes is not addressed directly by topological considerations and only relies on the robust design and diagnostic coverage of the sensor node. The authenticity of the communicating entities is not part of their concept [14]. If a sensor node within their proposed network architecture fails, there is no fallback defined. Regarding the communication security aspect of their work, a symmetric cipher is used, which dependent on the hardware co-processors of the selected microcontroller architecture is very fast and does not consume much resources. As mentioned above, a symmetric cipher alone does not cover authentication. Aghenta and Iqbal propose an open-source solution for a SCADA system to monitor the status of a photovoltaic power plant. They utilize commercial-off-the-shelve components like ESP32 development boards and Raspberry Pies as hardware platforms and MQTT as data transmission protocol. Their work can be considered as a feasibility study to avoid proprietary and expensive SCADA solutions for distributed and smaller or cheaper CI entities. No considerations regarding availability and communication security are part of their work [15].

Our paper is organized as follows: Section 2 presents state-of-the-art methods and technologies used in this work. The proposed sensor network concept is described in Sect. 3, followed by an introduction to the proposed software tools architecture in Sect. 4. Before the work is summarized in Sect. 6, an exemplary setup is presented in Sect. 5, in which the described network concept is implemented on a small scale.

2 Methods and technologies

The methods and technologies introduced in this section are used in the sensor network concept or the software tools architecture. They are well-documented and widely used in current standards and resemble a minimal set of core technologies and paradigms for the proposed work.

2.1 Functional safety

To increase the reliability and robustness of a technical system, engineers and designers can use the concept of redundancy. Classical redundancy, as described in the IEC 61508, states to implement vital parts of a system more than once but does not specify any restrictions in terms of part or design variation. Depending on the system's reliability to be achieved, specific restrictions and requirements for redundancy are demanded, the highest being the use of diverse redundancy [16].

The diversity approach enhances the traditional redundancy, which allows the use of identical components, designs, or systems to achieve higher robustness and make a system less prone to latent systematic faults and common mode failure. To establish diverse redundancy, it is crucial to meticulously replicate the functionality of all critical modules or subsystems of a safety-critical technical system. This replication has to be conducted so that no components, methods, designs, or algorithms from the original system are recycled. Frequently, manufacturers assign the design and development of such a redundancy block to entirely new teams of engineers. These teams receive specific guidelines outlining which components and designs to avoid to create something identical in function but with a significantly different design [17–19].

The parity of the number of redundant units is very important concerning the availability of stateful services. In order to ensure data consistency through the election of the main coordinating entity in a group association, even if the communication network is segmented (so-called "split-brain" scenarios), a clear quorum has to be guaranteed. This can be achieved by providing an odd number of redundant units [20, 21]. The same principle is also useful for data-collecting units such as sensor systems. In aviation, for example, "two-out-of-three" voting is predominantly implemented in the design of sensor systems. This method, which is based on the premise that sensor systems only have a low probability of failing at the same time, allows to validate the correctness of the collected data by majority voting, even if the individual sensor units have no or faulty self-diagnosis [16, 22, 23].

With regard to digital communication networks, diverse redundancy enhances reliability by ensuring the network remains operational even in the event of component failures, which is crucial for systems with high uptime requirements. Additionally, it improves

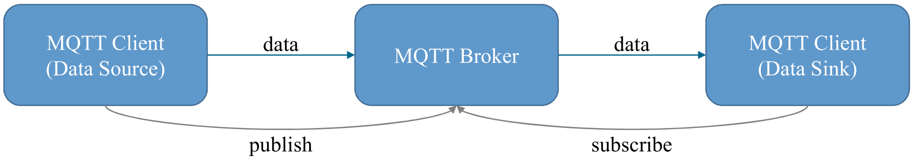


Fig. 2 Fundamental MQTT concept

robustness against attacks, as the diverse nature of redundant components and paths makes it more challenging for attackers to compromise the entire network, thereby enhancing cybersecurity. Moreover, diverse redundancy enables the network to adapt to changing environmental conditions, technological advancements, and emerging threats, ensuring long-term resilience. Within a network infrastructure, diverse redundancy can be achieved, i.e., by using multiple heterogeneous signal paths, different types of technologies, and protocols for data transmission to ensure continuous operation despite individual component failures and provides several advantages. While diverse redundancy offers significant benefits, it also presents challenges, such as increased complexity in network design and management, higher initial costs, and potential issues with interoperability between different technologies. Careful planning, robust testing, and ongoing management are therefore essential to effectively implement diverse redundancy [24, 25].

2.2 Communication security

The transport layer security (TLS) is the standard protocol for encrypted digital communication in computer networks such as the Internet. This cipher suite is an up-to-date method of communicating digitally in a secure way with optimized performance.

It combines asymmetric private/public key ciphers for the authentication of communication partners and the exchange of symmetric keys, which are then used to create a performant and secure symmetric cipher to protect the data transmission itself. All encrypted datagrams are additionally secured with a hash-based message authentication code (HMAC), which can be used to detect not only transmission problems, but also malicious tampering by third parties. TLS aligns with industry standards and best practices, safeguarding the integrity and confidentiality of the transmitted information [26–28].

While the detailed description of the included ciphers and key exchange algorithms itself is beyond scope of this work, a brief introduction to the certificate-based validation used as part of the cipher suite is crucial to understand how the authentication process works in the proposed approach in this work. The authenticity of the participating communication partners is guaranteed by certificates and associated private keys. The certificates can be used to encrypt or digitally verify data, and the private keys can be used to decrypt or digitally sign data. Therefore, the certificates can be shared among all potential participants and the private keys need to remain in a protected area, which can only be accessed by the respective participant itself. During the key exchange process, both communication partners will send their certificates to each other and encrypt or sign parts of the messages with their own private key. To validate, if the certificates are truly belonging to the expected communication partner, all participants have to trust a third party called certificate authority (CA). A CA is also in possession of a private key and public certificate. It digitally signs the certificates of all communication participants. The CA is trusted for checking and guaranteeing that a signed certificate really belongs to the specific entity. By making the CA's certificate available to all communication partners, they can use it to validate the digitally signed certificates of their counterparts [26, 29].

2.3 Data exchange

MQTT is a lightweight communication protocol designed for data exchange between devices. It operates on a publish/subscribe architecture, where clients can publish information on specific topics or subscribe to receive data from those topics with a central broker managing the data flow and distribution. The architecture is schematically depicted in Fig. 2. This data transmission utilizes the transmission control protocol (TCP), which enables reliable and efficient data transmission. MQTT supports retained messages enabling publishers to attach lasting messages to topics. New subscribers instantly receive the latest stored message, ensuring the clients receive essential data. MQTT's efficiency originates from its simplicity and low overhead, making it well-suited for scenarios where bandwidth and resource constraints are crucial. It is mainly used for machine-to-machine communication (M2M) or connection types like IoT [30, 31].

MQTTS (secure MQTT) utilizes the TLS cipher suite and therefore incorporates encryption and authentication for enhanced security as described in Sect. 2.2. This ensures that communication between brokers and clients is handled over a secure, encrypted channel and both are authenticated by a trusted authority.

3 Network concept

This section is structured according to the four distinguished network node types and their corresponding role in the proposed concept. Each node type is briefly described, and a first set of required implementation paradigms regarding functional safety

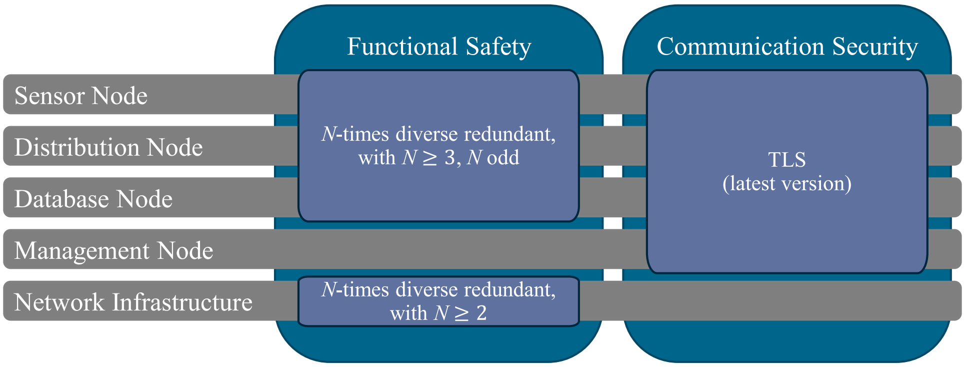


Fig. 3 Identified requirements of node types and network infrastructure regarding functional safety and communication security

and communication security is listed. Figure 3 gives an overview on these requirements. Considerations to follow these required paradigms during the development or implementation of the node types are also included. Additional requirements for the network nodes, especially concerning access control on the physical and cyber-physical level are beyond scope of this work. Yet, these requirements have to be considered, if the proposed concept is to be implemented in the future and have to be introduced accordingly for future design iterations. In addition to the defined network nodes, the underlying network infrastructure that allows communication on the media layers of the open systems interconnection (OSI) model [32] between the network nodes is included in the considerations.

3.1 Sensor node

Sensor nodes collect data from an analog or digital sensor and send it to distribution nodes utilizing the secure MQTTS protocol. These sensor nodes are implemented as a cyber-physical system consisting of the sensor itself, appropriate driver circuits for the sensor and a data processing platform that provides a suitable communication interface for the sensor and an additional communication interface for the network. The core software components are an MQTTS client, a component for certificate generation and the implementation of the configuration protocol as defined in this work. The software architecture of this component is described in more detail in Sect. 4.2. Without knowing the specific hardware platform of a sensor node and the reliability of the components used, an odd number of at least three diverse redundant sensor nodes are required for the same modality. This requirement follows the parity considerations regarding the number of sensor nodes used and diverse redundancy in Sect. 2.1.

3.2 Distribution node

The distribution nodes provide configuration information to the sensor nodes and distribute the acquired sensor data to other nodes that can visualize or otherwise process the sensor data. Distribution nodes will most likely be implemented on a cloud platform or server cluster running MQTTS brokers and software bridges interconnecting them in a high-availability configuration. Because of the beneficial feature to retain messages as described in Sect. 2.3, the distribution nodes have to be considered stateful in this concept. According to the redundancy-related methods proposed in Sect. 2.1, it is required to run at least three, but always an odd amount of diversely designed distribution nodes. The diverse design concerns not only the hardware of the computer systems used in the server cluster, but also their software. In addition to the services that are essential for this concept, such as the MQTTS broker and bridge, this also includes the operating system and all relevant modules. By requiring the usage of the secure version of the MQTT protocol, all relevant communication to other sensor nodes is secured by the TLS cipher suite. It is also required to configure the MQTTS broker to only accept connections, if the corresponding node's client certificate is also validated by the distribution node.

3.3 Database node

The concept presented in this paper includes nodes to persistently record the generated sensor data and make it available to other instances. The database nodes act as a data sink for the sensor data and enable further processing of the gathered data in the form of time series. These can be used for artifact detection and enhanced behavior simulations as carried out in the context to digital twins. Like the distribution nodes, the database nodes can be implemented on a server cluster. They also have to meet the same standards in terms of diverse redundancy and security for the utilized hardware and software. The main services that will run on the database nodes are MQTTS clients subscribing to the sensor data topics, a suitable and scalable database service allowing for data synchronization between the nodes to establish high-availability, and a data management module responsible for storing the received data into the database.

3.4 Management node

Management nodes are network operator personal computers or terminals running application software that implements the necessary core software components to manage and monitor the network with respect to the required implementation paradigms as defined in this section. These components are an MQTTS client, a module to sign certificates, the implementation of the configuration protocol as defined in this work, and a graphical user interface (GUI). The architecture of this application software is described in more detail in Sect. 4.1. As the management nodes are not necessary for the sensor data distribution within the network, they do not have to fulfill any special requirements regarding functional safety. However, they are subject to the requirement for secure communication using TLS, since they play an important role during the secure semi-automatic sensor node onboarding process.

3.5 Network infrastructure

The network infrastructure in the context of this work is defined by all entities that are crucial to establish a digital connection between the nodes on the media layers according to the OSI model. These are for example wiring, switches, and access points. At least a double diverse redundant implementation is required. For example, this can be achieved by connecting nodes via Ethernet [33] and Wi-Fi [34] simultaneously.

4 Software tools for design, implementation and administration of a reliable and robust sensor network

This section introduces the architecture of fundamental software modules to aid in the design, implementation and administration of robust and reliable sensor networks. These and other future software modules will be pooled as a software suite named Robust and Reliable Sensor Network Toolkit (R^2SNT). Initially, we focus on software modules for the sensor and management nodes and a detailed description of the secure semi-automatic onboarding process for new sensor nodes.

4.1 Management node application (appMN)

The application software has been designed to support the network operator in two core tasks: sensor node management and compliance checking assistance against the implementation paradigms defined in Sect. 3. Figure 4 gives an overview of the essential parts of appMN. The sensor node management module is used to accompany the entire life cycle of a sensor node. It starts with the semi-automatic and secure integration of new sensor nodes into the network. This process is described in detail in Sect. 4.3. After integration, this application software can be used to add meta-information such as the location of the sensor, a human-readable name, and a description. If the sensor node also supports additional specific variables, e.g., the publication interval for sensor data, these can also be configured with appMN. At the end of the life cycle, the application offers functions for removing and blocking the node so that no further data from the decommissioned node can be distributed via the network.

To help with the compliance check against the implementation paradigms, the nodes integrated into the network and their connections to each other are visualized in the software. In support of the operator, the software visually warns of non-fulfilled paradigms, such as the lack of redundancy of a sensor node, and provides information on how the conditions can be improved. The entities participating directly in the MQTT-based configuration protocol are introduced to the network model automatically. Network infrastructure-related entities such as switches or access points not participating in stated protocol also have to be added to the model to enable the evaluation of the network infrastructure itself against the required implementation paradigms. Because of the variety of components of which some do not have suitable means for automatic detection, the software additionally helps with the guided manual collection of meta-information for not automatically incorporated entities.

The GUI provides intuitive access to the software functions for the operator. It helps to visualize the model of the network and shows associated problems with regard to the implementation requirements from the concept presented in Sect. 3 and proposes suitable solutions. The functionality of the node management is also reflected in the elements of the GUI, which guides the operator through processes such as sensor integration or configuration.

Another essential part of the software is an MQTTS client implementation that can connect to distribution nodes to retrieve and send data to the network, enabling the automatic creation of a network model and sensor management in general. Regarding the network model generation function, the software also has to implement the configuration protocol for sensor nodes. In addition to the actual configuration of the sensor, this protocol also defines the processes important for sensor management, such as integrating new sensor nodes. Finally, a software module for signing certificates with a CA private key has to be part of the software required during the secure semi-automatic onboarding process.

4.2 Sensor node library (libSN)

The sensor node library was designed to provide sensor node developers with an easy-to-use software module encompassing a minimalistic application programming interface (API) that can be integrated into their own sensor node software. Figure 5 gives

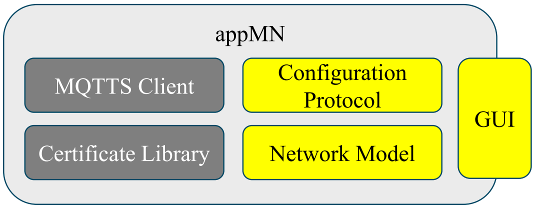


Fig. 4 Essential parts of appMN (Design responsibility: gray—third parties; yellow—this work)

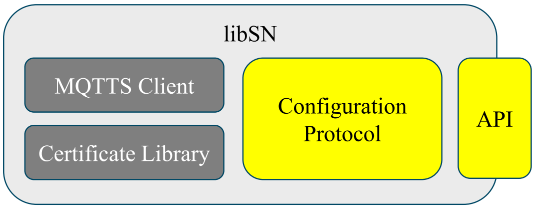


Fig. 5 Essential parts of libSN (Design responsibility: gray—third parties; yellow—this work)

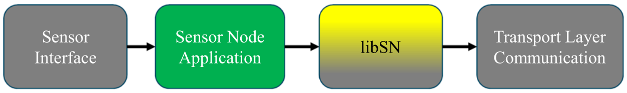


Fig. 6 Flow of sensor data on a sensor node (Design responsibility: green—sensor node developer; gray—third parties; yellow—this work)

an overview of the essential parts of libSN. It includes third-party libraries, such as an MQTTS client, a library for creating digital certificates and corresponding private keys, and an implementation of the proposed configuration protocol for sensor nodes. The configuration protocol itself is fully based on the MQTTS protocol and enables a secure semi-automatic onboarding process for new nodes (see Sect. 4.3). The developer has to provide static configuration artifacts, like the CA certificate to validate the certificates of distribution nodes. In addition, the IP address or domain name of one or multiple distribution nodes or a distribution node cluster and the data signature of the sensor node has to be defined. This data signature determines how the messages published by the sensor node are interpreted. This includes the quantity, type and arrangement of the data. Furthermore, the data signature of a sensor node can describe certain additional variables that are used to configure the behavior of the node. These variables can be changed via appMN (described in Sect. 4.1). After providing these artifacts, developers are able to process the collected sensor data and the configuration changes initiated by appMN with minimal effort.

During runtime, libSN has to be initialized together with other parts of the sensor node application. In this phase, the developer registers specific functions that are called when configuration variables have been changed over the network and when errors have occurred. Depending on the platform of the cyber-physical system selected for the sensor node, additional functions for communication on the transport layer, e.g., via TCP, also have to be registered by the developer. The MQTTS client uses these to communicate with the distribution nodes. The initialization method also checks whether the sensor node has a valid key/certificate pair and triggers the onboarding process if necessary. The final steps of the method ensure that the sensor's data signature is published and the sensor node's configuration is downloaded, if available. After initialization, the acquired sensor data can be transferred to the library with a single function call. The secure and reliable data transfer to the distribution nodes is completely encapsulated by the library. The data flow is shown schematically in Fig. 6.

4.3 Secure semi-automatic sensor node onboarding

The secure semi-automatic onboarding process for new sensor nodes is mainly implemented by two software functions: the initialization function of libSN and the onboarding function of appMN. The flowcharts of the two functions are shown in Figs. 7 and 8. During initialization, the sensor node checks for the presence of its individual signed key/certificate pair needed to authenticate itself to the distribution nodes. If not available, a new pair including a signing request (SR) is generated and the corresponding fingerprint is displayed in a human-readable way on a potential display of the sensor device or via another local interface. The generated private key has to be kept secure on the sensor node and suitable measures have to be taken to prohibit access to it. A connection is then established to a distribution node on which the SR is published to a specific topic. For this initial connection, in which new sensor nodes do not originally possess the necessary individual valid key/certificate pair for authentication, new nodes utilize a common key/certificate pair, which allows very limited access only to the topics relevant for the onboarding process. The management node, running the appMN onboarding function, is subscribed to the topic for newly published SR. After receiving the SR, the operator manually checks whether it belongs to the sensor node to be integrated and validates the displayed fingerprint. After successful validation, the operator signs the new certificate with the private key of the CA using the corresponding password. The signed certificate for the sensor node is then published to a specific topic to which the sensor node has subscribed earlier. Once the sensor node has downloaded its individual signed certificate, it establishes a new connection to the distribution node, publishes its data signature, and downloads its configuration, if available. The node is now able to participate in the secure data exchange of the sensor network as intended.

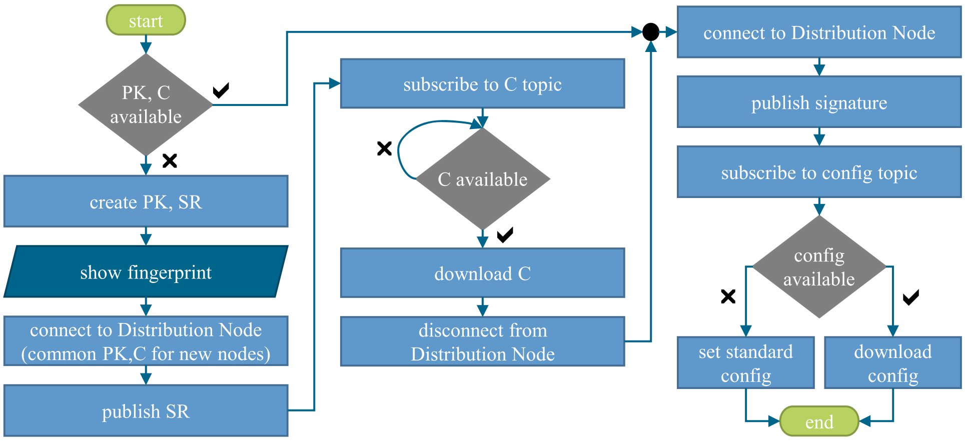


Fig. 7 Flowchart of libSN initialization (PK—private key; C—signed certificate; SR—signing request including unsigned certificate)

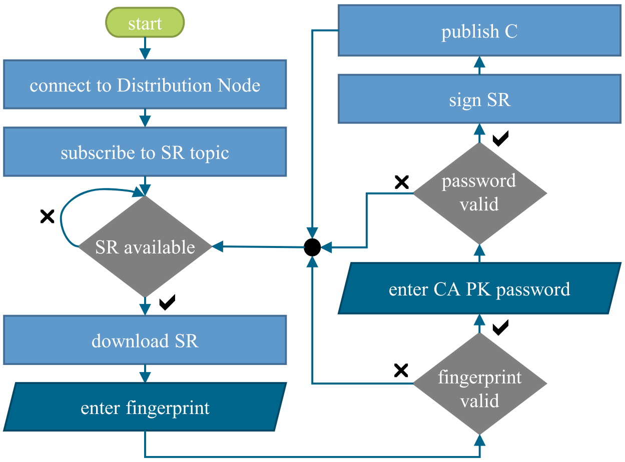


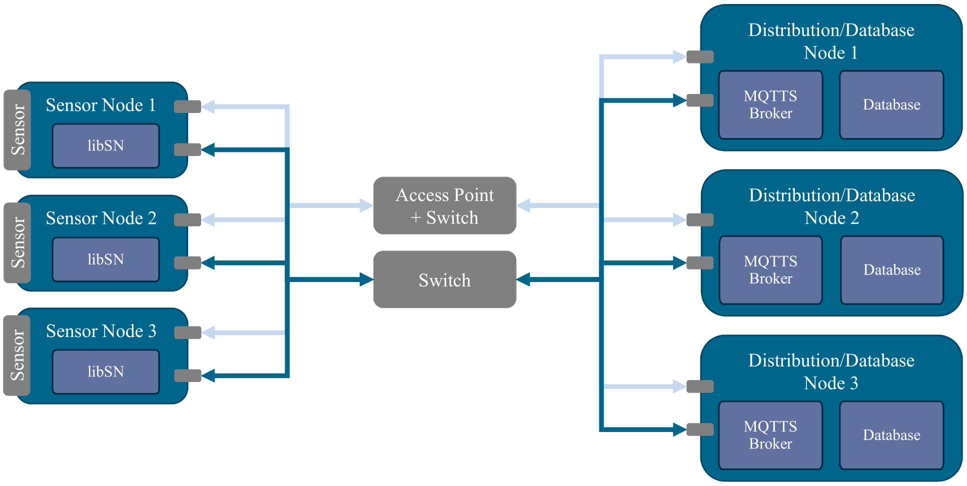
Fig. 8 Flowchart of appMN onboarding function (PK—private key; C—signed certificate; SR—signing request including unsigned certificate)

5 Exemplary setup

The objective of the basic setup presented in this section is to record the temperature of a single room. It is a proof-of-concept setup in which the distribution nodes, database nodes, and network infrastructure were not implemented in a diverse but rather in a classically redundant manner due to resource restrictions of the underlying project. Yet, the sensor node hardware is implemented according to the design paradigm of diverse redundancy, as requested by the concept proposed in Sect. 3.

The setup shown schematically in Fig. 9 consists of three temperature sensor nodes and a triple-redundant, high-availability server cluster used as shared hardware for the distribution and database nodes. They are connected via a double-redundant network infrastructure. The first communication path (light blue in Fig. 9) consists of a Wi-Fi access point/Ethernet switch combination and is used by the wireless network interfaces of the sensor nodes and one of the wired interfaces of each server. The second path (dark blue) is used by another wired interface of each server and the wired interfaces of the sensor nodes and runs via a separate Ethernet switch. In the event of a network failure on one path, the nodes seamlessly start using the remaining path, ensuring continuous connectivity. The sensor nodes are implemented using three different sensors and computing platforms running a preliminary implementation of libSN to send data to the distribution nodes. The distribution nodes are realized with the help of MQTTS brokers that run on each server and are connected via so-called bridges. These bridge services also run on each server and are subscribed to the broker running on the same server to receive all published messages and forward them to the other brokers on the remaining two servers. In addition, the upstream configuration enables the brokers to be reached via a shared IP address and distributes all incoming connection requests evenly among them (load balancing). This and the bridges allow the brokers to act as a single entity and prevent possible disruptions caused by the failure of a single server. The database nodes are implemented on the same server cluster as the MQTTS brokers. They consist of a highly available database service that runs on all three servers to distribute the permanently stored data across them, thus reliably mitigating data loss if a single server fails. Similar to the MQTTS brokers, the database cluster behaves like a single server and can be accessed via a shared IP address. The last service to complete the setup is the so-called database handler. These services, which subscribe to all relevant topics on the MQTTS broker cluster, store all published sensor data in the database.

Fig. 9 Schematic overview of the exemplary implementation



6 Conclusion

In a first contribution to the long-term development of a reliable and secure sensor network for CI monitoring, we have presented an initial version of a concept that requires the topological paradigm of diverse redundancy and secure communication using MQTTS relying on the included TLS cipher suite. After introducing the basics of these standard methods and protocols, we defined an initial set of node types and assigned requirements for their implementation. Based on these requirements, software tools were designed to support sensor node developers by providing standard features for sensor node management and implementation. In particular, the secure semi-automatic onboarding process was presented in detail. The concept and software tools architecture were developed as independently as possible from specific hardware platforms, software frameworks, and network technologies to enable an open standard for communication within a critical infrastructure sensor network and also between such networks in the future. The results presented in this paper are to be understood as a basis for further developments.

The next step in developing the concept should be defining requirements considering access control on the physical and cyber-physical level to protect the private keys of the network nodes. While the cyber-physical level requires, among other things, strictly configured firewalls, anti-tampering measures are needed on the physical level, especially for network nodes in public areas that attackers can easily compromise without those countermeasures. Regarding the implementation of the software tools, the libSN software module needs to be implemented at least three-times to meet the diverse redundancy requirement on software level. For each of these implementations, different third-party libraries have to be used or re-implemented. In addition, three disjoint development teams have to carry out the three implementations. We aim to publish one variant of libSN as part of the R^{2}SNT software toolkit as an open-source and platform independent reference implementation in the future. Based on this implementation and an experimental setup addressing specific test cases, the effectiveness of the proposed concept will be evaluated as part of our subsequent work.

Acknowledgements This work is part of the project *Distributed Sensor Systems for Near-Real-Time, Highly Automated Monitoring of Critical Infrastructures (VeStokrit)* at the German Aerospace Center—Institute for the Protection of Terrestrial Infrastructures. The authors would like to thank Dr. Kostyantin Konstantynovski for his advice regarding CBRNe applications. In addition, the authors would like to thank Prof. Dr.-Ing. Norbert Jung, Prof. Dr. Peter Kaul, and Prof. Dr. Alexander Asteroth of Bonn-Rhine-Sieg University of Applied Sciences for their advice in their respective fields of research and for supervising this work as part of the doctoral proceedings of Sebastian Sporrer.

Author contributions Sebastian Sporrer was responsible for designing the network concept and the software tools architecture. Norman Niemann dealt with the basics of the MQTTS protocol, was responsible for the parts of the exemplary implementation based on this, and contributed corresponding parts to this manuscript. Dr. Christof Hammer supplemented basics of functional safety, wrote about the method of diverse redundancy and its implications for the concept, and supervised this work. The first draft of the manuscript was written by Sebastian Sporrer, and all authors commented on previous versions of the manuscript. All authors read and approved the final manuscript.

Funding Open Access funding enabled and organized by Projekt DEAL.

Data Availability Statement No datasets were generated or analyzed during the current study.

Declarations

Conflict of interest The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

References

1. Nationale Strategie zum Schutz Kritischer Infrastrukturen (KRITIS-Strategie), Berlin (2009). German Federal Ministry of the Interior and Community - Referat KM 4
2. Act on the Federal Office for Information Security: BSI Act (2009 (Federal Law Gazette I p. 2821), last amended by Article 12 of the Act of 23 June 2021 (Federal Law Gazette I p. 1982)). https://www.gesetze-im-internet.de/bsig_2009
3. 10 Jahre "KRITIS-Strategie": Einblicke in die Umsetzung der Nationalen Strategiezum Schutz Kritischer Infrastrukturen. Praxis im Bevölkerungsschutz, vol. 21. Bonn (2020). German Federal Office of Civil Protection and Disaster Assistance - Refereat II.3
4. A.I. Sarwat, A. Sundararajan, I. Parvez, M. Moghaddami, A. Moghadasi, Toward a smart city of interdependent critical infrastructure networks. In: Amini, M.H., Boroojeni, K.G., Iyengar, S.S., Pardalos, P.M., Blaabjerg, F., Madni, A.M. (eds.) Sustainable Interdependent Networks. Studies in Systems, Decision and Control, vol. 145, pp. 21–45. Springer International Publishing, Cham (2018). https://doi.org/10.1007/978-3-319-74412-4_3
5. X. Liu, C. Qian, W.G. Hatcher, H. Xu, W. Liao, W. Yu, Secure internet of things (IoT)-based smart-world critical infrastructures: survey, case study and research opportunities. IEEE Access **7**, 79523–79544 (2019). https://doi.org/10.1109/ACCESS.2019.2920763
6. S.J. Moore, C.D. Nugent, S. Zhang, I. Cleland, Iot reliability: a review leading to 5 key research directions. CCF Trans. Pervas. Comput. Interact. **2**(3), 147–163 (2020). https://doi.org/10.1007/s42486-020-00037-z
7. M. Wisniewski, B. Gladysz, K. Ejsmont, A. Wodecki, T. Van Erp, Industry 4.0 solutions impacts on critical infrastructure safety and protection-a systematic literature review. IEEE Access **10**, 82716–82735 (2022). https://doi.org/10.1109/ACCESS.2022.3195337
8. C. Villar Miguelez, V. Monzon Baeza, R. Parada, C. Monzo, Guidelines for renewal and securitization of a critical infrastructure based on IoT networks. Smart Cities **6**(2), 728–743 (2023). https://doi.org/10.3390/smartcities6020035
9. L. Gomez, C. Ulmer, Secure sensor networks for critical infrastructure protection, in *2010 Fourth International Conference on Sensor Technologies and Applications*, pp. 144–150 (2010). https://doi.org/10.1109/SENSORCOMM.2010.30
10. F. Berizzi, C. Callegari, S. Giordano, A heterogeneous sensor network for the protection of critical infrastructure, in *Proceedings of 2014 Mediterranean Microwave Symposium (MMS2014)*, pp. 1–6 (2014). https://doi.org/10.1109/MMS.2014.7088918
11. A. Grilo, A. Casaca, P. Pereira, L. Buttyan, J. Gonçalves, C. Fortunato, A wireless sensor and actuator network for improving the electrical power grid dependability, in *Proceedings of the 8th Euro-NF Conference on Next Generation Internet NGI 2012*, pp. 71–78 (2012). https://doi.org/10.1109/NGI.2012.6252167
12. A.M. Grilo, J. Chen, M. Diaz, D. Garrido, A. Casaca, An integrated WSAN and SCADA system for monitoring a critical infrastructure. IEEE Trans. Industr. Inf. **10**(3), 1755–1764 (2014). https://doi.org/10.1109/TII.2014.2322818
13. B. Marchi, A. Grilo, M. Nunes, Dtsn: Distributed transport for sensor networks, in *2007 12th IEEE Symposium on Computers and Communications*, pp. 165–172 (2007). https://doi.org/10.1109/ISCC.2007.4381601
14. M. Niedermeier, X. He, H. De Meer, C. Buschmann, K. Hartmann, B. Langmann, M. Koch, S. Fischer, D. Pfisterer, Critical infrastructure surveillance using secure wireless sensor networks. J. Sens. Actuator Netw. **4**(4), 336–370 (2015). https://doi.org/10.3390/jsan4040336
15. L.O. Aghenta, M.T. Iqbal, Design and implementation of a low-cost, open source IoT-based SCADA system using esp32 with Oled, Thingsboard and mqtt protocol. AIMS Electron. Electr. Eng. **4**(1), 57–86 (2020). https://doi.org/10.3934/ElectrEng.2020.1.57
16. IEC 61508:2010 - Functional safety of electrical/electronic/programmable electronic safety-related systems - Parts 1 to 7 (2010-04)
17. National Instruments Corp.: Redundant Systems: Definition & System Redundancy Models (2023-05-17). https://www.ni.com/en/shop/electronic-test-instrumentation/add-ons-for-electronic-test-and-instrumentation/what-is-systemlink-tdm-datafinder-module/what-is-rasm/redundant-system-basic-concepts.html Accessed 2024-07-11
18. I. Malynyak, Functional diversity design of safety-related systems. The Educational Review, USA **2**(1) (2018) https://doi.org/10.26855/er.2018.01.004
19. E.C. Ramirez, Diverse redundancy used in SIS technology to achieve higher safety integrity (2008). https://library.e.abb.com/public/c8ebe6fed0204975852575ac0061e959/1462_DiverseRedundancy_Final.pdf Accessed 11.07.2024
20. T. Critchley, High Availability IT Services, 1st edn. Auerbach Publications, New York (2014). https://doi.org/10.1201/b17958
21. M. Riesewijk, High availability orchestration of Linux containers in mission-critical on-premise systems (2020). http://essay.utwente.nl/80468/
22. T. Grof, P. Bauer, Voting-based fault detection for aircraft position measurements with dissimilar observations. IFAC-PapersOnLine **53**(2), 14724–14729 (2020). 21st IFAC World Congress. https://doi.org/10.1016/j.ifacol.2020.12.1841
23. H. Benítez Pérez, J.L. Ortega Arjona, G.R. Latif Shabgahi, Definition and empirical evaluation of voters for redundant smart sensor systems. Computación y Sistemas **11**(1), 39–60 (2007)
24. M.C. Lucas-Estan, B. Coll-Perales, J. Gozalvez, Redundancy and diversity in wireless networks to support mobile industrial applications in industry 4.0. IEEE Trans. Industr. Inf. **17**(1), 311–320 (2021). https://doi.org/10.1109/TII.2020.2979759
25. J.P.G. Sterbenz, D. Hutchison, E.K. Çetinkaya, A. Jabbar, J.P. Rohrer, M. Schöller, P. Smith, Redundancy, diversity, and connectivity to achieve multilevel network resilience, survivability, and disruption tolerance invited paper. Telecommun. Syst. **56**(1), 17–31 (2014). https://doi.org/10.1007/s11235-013-9816-9
26. Internet Society: TLS Basics. https://www.internetsociety.org/deploy360/tls/basics/ Accessed 2024-07-11
27. E. Rescorla, The Transport Layer Security (TLS) Protocol Version 1.3. RFC Editor (2018). https://doi.org/10.17487/RFC8446 . https://www.rfc-editor.org/info/rfc8446
28. German Federal Office for Information Security: Mindeststandard des BSI zur Verwendung von Transport Layer Security: nach § 8 Absatz 1 Satz 1 BSIG – Version 2.4 vom 25.05.2023, Bonn. https://www.bsi.bund.de/SharedDocs/Downloads/DE/BSI/Mindeststandards/Mindeststandard_BSI_TLS_Version_2_4.pdf Accessed 2024-07-11
29. Telecommunication Standardization Sector of International Telecommunication Union (ITU-T) Study Group 17: Information technology – Open Systems Interconnection – The Directory: Public-key and attribute certificate frameworks: Recommendation ITU-T X.509 | ISO/IEC 9594-8, Ed. 9 (Okt. 10 2019). https://handle.itu.int/11.1002/1000/14033 Accessed 2024-07-11
30. S. Cope, How MQTT Works -Beginners Guide (2021). http://www.steves-internet-guide.com/mqtt-works Accessed 2024-07-11
31. A. Banks, E. Briggs, K. Borgendale, R. Gupta, (eds.) MQTT Version 5.0: OASIS Standard, (2019). https://docs.oasis-open.org/mqtt/mqtt/v5.0/os/mqtt-v5.0-os.html
32. ISO/IEC 7498-1—Information technology—Open Systems Interconnection - Basic Reference Model: The Basic Model (1994)

33. IEEE 802.3 - Standard for Ethernet (2022)
34. IEEE 802.11 - Standard for Information Technology–Telecommunications and Information Exchange between Systems - Local and Metropolitan Area Networks–Specific Requirements - Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications (2021)