symmetry

MDPI

Article

Real-Time Monitoring and Management of Hardware and Software Resources in Heterogeneous Computer Networks through an Integrated System Architecture

Constantin Lucian Aldea [1,†,‡], Razvan Bocu [1,*,‡] and Robert Nicolae Solca [2]

[1] Department of Mathematics and Computer Science, Transilvania University of Brasov, 500091 Brasov, Romania; costel.aldea@unitbv.ro
[2] Department of Electronics and Computers, Transilvania University of Brasov, 500024 Brasov, Romania; robert.solca@student.unitbv.ro
* Correspondence: razvan.bocu@unitbv.ro
† Current address: Blvd. Iuliu Maniu, Nr. 50, 500091 Brasov, Romania.
‡ These authors contributed equally to this work.

Abstract: The theoretical and practical progress that has occurred in the field of computer networks during the past fifteen years has enhanced the economical efficiency and social relevance of related real-world use cases. Nevertheless, this ubiquitous usage has also introduced numerous security risks. Therefore, monitoring hardware and software resources represents one of the main instruments used in order to prevent potential attacks and to ensure the security and reliability of a network. Various solutions have been reported in the related scientific literature. In essence, most of the existing approaches are not suitable to implement a real-time hardware monitoring and management solution, particularly in heterogeneous networks. Therefore, the main contribution of this paper is represented by an architectural and implementational model, which is effective in order to build an interconnected system that can help system and network administrators to secure a network. This requirement is met by considering symmetrical design and implementation features related to various operating systems. Thus, the existing symmetrical relationships among identified parameters allow for the data to be wrapped into the same custom network packages, which are transported over the communication medium or are stored using the same data structures or tables. The system has been thoroughly assessed considering several real-world use case scenarios, and the results demonstrate that the proposed model can be applied to software-defined networks, which can be protected by relevant intrusion detection systems (IDS).

Keywords: network security; security monitoring; information security; computer security; data security; intrusion detection system

Citation: Aldea, C.L.; Bocu, R.; Solca, R.N. Real-Time Monitoring and Management of Hardware and Software Resources in Heterogeneous Computer Networks through an Integrated System Architecture. *Symmetry* **2023**, *15*, 1134. https://doi.org/10.3390/sym15061134

Academic Editors: José Carlos R. Alcantud, Jian-Qiang Wang and Sergei D. Odintsov

Received: 11 April 2023
Revised: 9 May 2023
Accepted: 18 May 2023
Published: 23 May 2023

1. Introduction

Cybersecurity represents a major concern, since the world is becoming more and more dependent on technology. Vulnerabilities are constantly being discovered, and malware can threaten any organisation. Common Vulnerabilities and Exposures [1] as well as Common Weakness Enumeration [2] are two important catalogues of disclosed cybersecurity vulnerabilities widely used in software development and IT operations. These vulnerabilities are discovered, ranked and published by various organisations around the world. Although large technical communities are working to mitigate the risks associated with identified vulnerabilities, other threats remain undiscovered, and the responsible stakeholders or actors, such as researchers, managers, software developers, network administrators, and security specialists, must remain vigilant and continue to develop tools to monitor resources and to prevent intrusions or cyberattacks. The range of methods, techniques and tools used to protect against cyberattacks is well documented in the relevant scientific

literature. The cyber kill chain, which is described in [3] by Yadav and Rao, is a model for depicting cyberattacks in an attempt to develop incident response and analysis capabilities. From a risk-mitigation perspective, there are studies on how different methodologies, such as NIST standards, can help and support from a management perspective (Fenz et al.) [4]. In addition, other approaches have proposed technical solutions regarding security in Internet of Things (IoT) environments, by describing challenges and designing solutions that are either IoT-specific units (Mehnen et al. [5]), or solutions that are rather data oriented. Thus, in the article by Manogaran et al. [6], the authors address big data issues in the health industry, while reference [7] proposes an adaptive cybersecurity monitoring system based on correlation techniques to detect intrusion patterns, track event evolution, and analyse and identify security events.

Monitoring is one of the first stages, which functionally defines an intrusion detection and prevention system. Therefore, several related topics and scientific contributions have been researched. Proof of concepts concerning network resource monitoring were studied, designed and developed. Thus, the paper by Lee et al. [8] argues that continuous monitoring of network behaviour helps to define and implement more reliable system configurations. Among other aspects, the paper by Montes et al. [9] proposes a unified cloud monitoring taxonomy. Furthermore, relative to complex modern cloud infrastructures, the properties of a monitoring system are discussed in [10]. Moreover, certain solutions based on Kubernetes, such as the one that is presented in article [11], have been designed. Additionally, the relevant literature describes architectures that are specific to certain attack patterns, such as the solution that was described in the paper by Kshirsagar and Patil [12], which provides a real-time monitoring solution for blackhole attacks.

The main objective of this paper is to present the design and implementation of a unified system architecture that aims to protect all the network types already described, by building a highly modular solution that supports regular networks, IoT devices, and cloud infrastructures.

One of the requirements is to design modules that are cross-platform compatible. This is achieved by considering symmetrical design and implementation features for different operating systems. Thus, the resulting architectural models ensure the use of globally consistent data and related reporting capabilities. For example, the existing symmetrical relationships among identified parameters permits wrapping the data into the same custom network packages, which are transported over the communication medium or are stored using the same data structures or tables.

This paper addresses aspects that software that engineers need to consider when they want to conceptualise and build from scratch the proper monitoring tools in the form of one interconnected system, which is integrated into their own infrastructure to monitor the resources of a software-defined network. Thus, this paper describes an integrated system which can be used for the real-time monitoring and management of hardware and software resources in heterogeneous computer networks.

This paper is structured according to the following sections. Section 2, Materials and Methods, introduces the relevant existing research papers and describes the general features of the proposed approach. The main characteristics of the proposed proof of concept are presented in Section 3, System Architecture and Methodology, which discusses the considered methodology and describes the system architecture. Section 4, Experimental Assessment and Discussion, further discusses the relation to the structural features of the analysed platform, and it presents the experimental assessment. Furthermore, Section 5, Conclusions, summarises the contributions of this paper, concluding the paper.

2. Materials and Methods

Modern software systems are defined by complex architectures that process large amounts of input data. Thus, efficient usage of the respective hardware and software components is determined by the implementation of appropriate software and hardware resource-monitoring modules. The following paragraphs present the most relevant existing

scientific references that have influenced the development of the system that is proposed in this paper.

2.1. Logical Management of Mobile Networks

The article of Gentry [13] analyses deployment approaches regarding the deployment of services in 5G networks. Thus, the network functions (NF) are specified and implemented relative to traditional architectures. Relevant functions are designed and deployed using a virtual machine (VM) or using serverless architectures that distribute the functional features to containers. The real-world behaviour of microservices in connection to Kubernetes was evaluated. The assessment concerns the usage of several HTTP protocol variations. These were chosen in order to design and deploy 5G interfaces, which consider services such as functional entities. The experimental results prove that particular performance improvements can be realised using HTTP/3, which transmits data through the consideration of the QUIC transport protocol (Carlucci et al. [14]). This takes into account problems that relate to delay or loss of transmitted data. Thus, deployment strategies should consider 5G infrastructures, which are capable of proactively monitoring the scalable data connections.

The article by Yan et al. [15] concerns 5GsatEC, which represents an edge computing framework based on 5G data links. The goal is to optimise latency and network coverage. This approach considers microservices that are based on edge computing components and embedded hardware platforms, which are useful during the design and deployment of the involved satellite systems. The reported increased flexibility is based on the optimisation of the resource management. This concerns the central processing unit (CPU), the graphics processing unit (GPU), and field programmable gate array (FPGA) devices.

The article by de Jesus Martins et al. [16] presents SWEETEN, a support system for the operation of a 5G network through proper management components. The enhancement in the related functional stack also implies the design and deployment of an improved model, which concerns both network-related operational features and management functions. More precisely, the paper describes a functional prototype, which is assessed through a dynamic Cloud Radio Access Network (C-RAN) system (Hossain et al. [17]).

The flexible and open architecture of heterogeneous networks must be compatible with various services. These networks are required to approach the problem of adaptability relative to the packet-scheduling mechanism of the user plane function (UPF) (Hsieh et al. [18]). This optimises the usage of the overall data infrastructure. Thus, Xinjian et al. [19] presented a 5G microservices open model, which is based on the usage of object-oriented modelling architectures in the field of industrial Internet. Thus, relative to the IEC61850 standard (Aftab et al. [20]), simple services, such as location and time of the 5G data network, are specified through a public data class. These determine the required microservice components, which are compatible with various logical functions inside the 5G network and with the proper virtual network functions (VNF))Akyildiz et al., Xia et al. [21,22]) logical model. Furthermore, significant contributions, which pertain to the optimal implementation and usage of 5G-related microservices in various fields, including industrial Internet Khalfi et al. [23], are reported by Xu et al. and Spyridis et al. [24,25].

The article by Gholami et al. [26] identified and valued the logical link among computing and network fabric concerning various microservices. This article also addresses an optimisation model concerning the identification of the proper deployment methods for each microservice, which are members of a microservices complex. The article reports two real-world applications that are based on heterogeneous networks. They concern video surveillance and an intelligent transportation system (ITS). This solution is suitable only for particular use cases, and it cannot be easily adapted to other real-world situations. Furthermore, relevant related contributions are proposed by Guija and Siddiqui [27], Xu et al. [24] and Soenen et al. [28], who describe 5G-oriented software platforms that include the required authentication and authorisation mechanisms.

Orduz et al. [29] describes an architecture based on microservices, which offers a claimed "finer scalability" and improved efficiency of the used resources relative to standard monolithic microservice architecture designs. This also pertains to the broader scope of artificial intelligence (AI) as a microservice (AIMS) (Lee et al. [30]). Additionally, related blockchain-based models are described by Prabadevi, B. [31] and Gayialis et al. [32].

Salhab et al. [33] presents a heterogeneous network-based microservices deployment approach, which considers both software-defined networks (SDN) [34] and network function virtualisation (NFV) [35]. This has the role of creating efficient microservice orchestration (generation), which is based on the production of logical 5G data infrastructures.

Low-latency heterogeneous networks determine a rich research scope, which is essentially described by Rao et al. [36], Nadaf et al. [37] and Alencar et al. [38]. This category of scientific contributions is especially important considering that the communication between microservices should be sustained by low-latency data links. Furthermore, related articles report on the consideration of AI-based optimisation models, which effectively reduce the obtained latencies and network loads. Thus, relevant papers belong to Luo et al. [39], Kaur et al. [40], Yan et al. [41], and Hannousse and Yahiouche [42].

Network slicing relates to the logical partition of physical infrastructures into logically independent networks, which are based on heterogeneous networks. The related network slices should conform to the specified service-level objectives (SLO), which relate to the deployed commercial services, as is suggested by Huang et al. [43]. Furthermore, Sheoran et al. [44] presents the difficulties of network orchestrators to allocate the necessary resources for heterogeneous network slices in connection to proper artificial intelligence models. The described approach also implies the consideration of artificial intelligence to determine the core placement and scaling decisions relative to shared infrastructures.

2.2. Software Platforms

Containers and microservices determine natural approaches for the deployment of IoT applications on cloud-based infrastructures. This is suitable for various use cases, such as those related to the agricultural sector, as is proposed by Mateo-Fornés et al. [45]. Thus, the incorrectly patched software containers may determine vulnerable microservices. Ying et al. [46] proposed a system to enhance container-side security models, which concern unknown attack patterns related to a "mimic defence network". Thus, a set of resources regarding attack pattern images was created. Moreover, the variable execution outputs are assessed to detect potential vulnerability patterns.

The integrated acquisition of personal health metrics data determines a significant scope of investigation, which is also determined by the deployment of next-generation heterogeneous mobile networks. Thus, Bocu and Costache [47] described an integrated personal health metrics data management system that is based on a virtualised symmetric 5G network. The personal health data were collected using wearable IoT devices, which are specified by Pruna and Vasilescu [48], using a client mobile application. The proposed system was evaluated through a field trial, which was thoroughly presented. The paper has the merit to prove that it is possible to specify and deploy a comprehensive distributed IoT-related model which ensures complete personal data privacy. Interesting aspects that are connected to the considered homomorphic encryption models are presented by Kim et al. [49].

2.3. Network Monitoring

The available literature describes pertinent and interesting contributions related to network monitoring in various fields that have influenced the approach outlined in this paper, either conceptually or practically.

Stanimirović et al. [50] described a monitoring system for the main electricity distribution company in Serbia. The implemented system is deployed in two different regional hubs of the company. It has been used in production since 2016. During this period, real-life data have been collected, which were used to analyse and investigate the performance of

the low-voltage distribution network. The data have also been used in order to implement artificial intelligence (AI) models, which detect the power grid sections that lose electrical energy. The main contribution of this paper is represented by the description of the monitoring and control software modules. Moreover, the system was fully integrated into the Serbian company's heterogeneous network-controlled software infrastructure, which fully justifies the relevance of this article.

Colace et al. [51] proposed a methodology based on the aggregated use of three graphic models. This has the role to thwart attacks on pervasive systems considering three perspectives: probabilistic, contextual and ontological. Thus, the article reports the use of Bayesian networks, which are built through an ontological definition of the problem. This is related to the particular use case scenario, which is represented by a context dimension tree. Furthermore, the described model is assessed considering a real-world use case scenario, and the overall output of the experiments is satisfactory. Furthermore, an interesting related contribution is reported by Hohemberger et al. [52], which defines the concept of coordinated network monitoring as an essential functional approach for the optimisation of network monitoring in virtualised environments.

Shen [53] proposed SDN-Monitor, a software system that selects network switches to monitor in order to reduce resource consumption. Nevertheless, the control plane resources are limited in relation to SDN regarding the processing capacity of the controller, the network bandwidth, and the switch performance. Therefore, an excessive amount of network monitoring essentially affects the data plane traffic performance. This issue should be addressed by the described software system.

The paper by Abbasi et al. [54] includes a survey regarding the application of deep learning to network traffic monitoring and analysis (NTMA) applications. Furthermore, Li et al. [55] discusses the novel approaches concerning Internet of Things (IoT)-based ecosystem monitoring. These encompass new data loggers and long-distance wireless sensor network structures, which support the rapid transmission of data from devices to wireless networks. Moreover, a relevant real-world use case, which values the monitoring of heterogeneous networks, was treated by Clot et al. [56]. Furthermore, Mahajan et al. [57] analysed a real-world use case, where the performance of a high-speed diesel (HSD) pump is predicted by analysing its effective output. The described approach demonstrates the utility of network traffic prediction and monitoring over heterogeneous wireless networks. Nevertheless, the conceptual and practical presentation can be generalised to any type of heterogeneous network.

Another practical contribution, proposed by MahmoudZadeh et al. in [58], describes an efficient data collection model, which considers a group of unmanned aerial vehicles (UAVs) to monitor and collect data, which are generated by a large distributed sensor network. The proposed approach considers the appropriate network-monitoring mechanisms and required collaborative data collection models. The described mission planner model uses the differential evolution optimisation algorithm, which enables the UAVs to maximise the number of sensor nodes visited, taking into account the priority of the sensors and avoiding the redundant collection of data generated by the sensors. Additionally, the relevant scientific literature includes articles that report technical contributions concerning the effective problem of network and infrastructure monitoring. As an example, Ali et al. [59] proposed Redfish-Nagios, a scalable out-of-band monitoring tool for modern high-performance computing (HPC) systems. More precisely, the described approach conforms to the Redfish telemetry model [60], which allows for a more efficient monitoring of next-generation scalable HPC systems.

Another compelling real-world use case, in the scope of engineering, was proposed Sofi et al. [61], who included a comprehensive review of advances regarding data acquisition, processing, diagnosis, and data retrieval stages relative to structural health monitoring. The review particularly focused on recently used wireless data acquisition hardware and software resources, which include relevant artificial intelligence (AI) algorithms. These were particularly analysed relative to reinforced concrete (RCC) buildings and bridges.

The survey also suggests that the relevant structural health monitoring technologies are still insufficiently developed, and the usage of heterogeneous data networks is regarded as an appropriate design choice for the enhancement of the overall structural health-monitoring process.

Network data traffic analytics and monitoring represent a relevant research subject in the scope of machine learning techniques. Considering the past ten years, researchers in academia and the industry have described several solutions. However, a majority of the proposed approaches consider unreliable measurement tools and methodologies. In this context, two relevant contributions should be mentioned. The first belongs to Aouini and Pekar [62], which describes the design and implementation of NFStream, a network data analysis application. Nevertheless, the approach seems to be relevant more as a common research platform and less as a real-world monitoring and data analysis solution. In this context, Yin and Zhang [63] proposed a network security assessment model which combines elements of fuzzy reasoning and probability density feature detection methods. Furthermore, Tran et al. [64] proposed a related development in the scope of industrial informatics, which describes an integrative model between an IoT platform based on heterogeneous networked structures and a deep learning neural network (DNN) algorithm relative to the online monitoring of computer numerical control (CNC) machines.

Some limitations that should be considered in related software frameworks are represented by the implementation of improper data-routing algorithms. Thus, the packets may be sent from the sender to the recipient node. Thus, Sengan et al. [65] analyses the combined effect of encountered data transfer issues along the heterogeneous network. Moreover, a similar contribution, which pertains to the scope of healthcare, is described by Qi and Su in article [66].

Novel approaches are often intended to be used in particular real-world use case scenarios, and they are generally not comparatively analysed against similar methods. Thus, paper of Yu et al. [67] describes a simulation that is intended to support the comparison of various network-monitoring methods relative to various dynamic network changes. Considering the set of simulated dynamic networks, the performances of several network-monitoring methods presented in the surveyed literature are compared.

The chapter that is authored by Ageyev et al. and is published in [68] proposes a taxonomy of attacks and intrusions, as well as an analysis of anomalies and intrusions relative to decentralised distributed networks. The main focus pertains to intrusion detection systems, and to the monitoring of network traffic anomalies. The algorithmic and data-processing models consider proper statistical methods.

Lu et al. [69] proposed topology optimisation and an iterative parameter identification model, which are useful in order to estimate the common model factors in WSN networks. This topology reduces the overall power consumption and the data channel's distance. Two simulations are described which assess the described approach. The results demonstrate that an improved network-monitoring system positively influences both the computational and energy efficiency of the monitored interconnected structure. Additionally, when managed and accessed over data networks, energy-efficient IoT technologies need to be carefully considered. Additionally, Alsharif et al. [70] highlights the importance of machine-to-machine communication as one of the four factors in building sustainable and energy-efficient IoT technologies.

2.4. Importance of the Proposed Approach

The deployment of an existing network-monitoring solution may take weeks, or even months, presuming that adequate technical expertise exists. Despite the fact that many relevant tools are provided as services, rather than on-premise deployments, this issue still exists. The necessity to customise existing tools generates long deployment times. Furthermore, it is often necessary to integrate existing monitoring tools with certain software systems of target organisations. Many of the existing network-monitoring approaches often lack required structural and functional flexibility, which would allow for their easy

integration into third-party software systems. This paper describes an integrated system architecture, which proposes a decoupled and flexible architecture that mediates efficient integration into most of the existing software infrastructures.

Software system customisation represents another common task which many interested organisations require. Many of the existing network-monitoring tools provide limited functional customisation capabilities. The integrated system architecture, which this paper describes, can be customised or extended efficiently, so that necessary functional requirements are implemented in accordance with particular real-world use case particularities.

Most IT organisations that aim to securely manage, personalise and use their network-monitoring software tools generally experience a negative impact on their business processes. This is caused by the significant resources which are involved in the respective software customisation processes. Additionally, the skilled software engineers that are allocated to their respective tasks delay their involvement in their main software-development processes. The existing experience proves that costs are especially problematic in the case of organisations that implement their own network-monitoring tools, rather than a software solution that can be easily customised. The integrated system that is proposed provides a necessary set of real-time network and resource-monitoring functions, while the eventual functional changes can be implemented efficiently using the decoupled modular architecture of the integrated system. This is one of the very few similar systems which implements a comprehensive set of network-monitoring and management functions, along with a decoupled modular architecture. Therefore, it can be stated that the proposed system architecture effectively fills a conceptual and practical gap.

3. System Architecture and Methodology

3.1. Platform Architecture

Over the years, multiple platform independent architectures have been designed and developed. The OPC Unified Architecture is proposing a model that defines, as its main goals, platform independence, security, and extensibility (Mahnke et al. [71]). Other approaches define different architectures based on data distribution service standards to define and implement platform-independent models (Pardo-Castellote [72]), or they have implemented middleware components to facilitate node-to-node computer communication [73]. The integrated sample tools presented in this paper consider multiple utilisation scenarios regarding the secure management and monitoring of network resources. Considering the various user scenarios, it is worth mentioning the possibility to view the status of the resources and to compare them with the respective previous states. Considering security aspects, the analysis of time series for the functional parameters of a system could detect and prevent bottlenecks and attacks. The end users may perform a manual snapshot or alternatively plan automatic snapshots to watch the CPU, memory or network status. Moreover, an interesting contribution, which relates to the fundamental idea of symmetry, is reported in Wu et al. [74].

The problem of platform-independent architectures in the industry was also studied by Marcos-Pablos and García-Peñalvo [75], which suggests that devices are diverse, considering the range of provided services.

The system that is presented in this paper can be implemented in infrastructures that use this kind of platform-independent architecture, due to its high adaptability. Thus, Figure 1 depicts the main workflows of the proposed system architecture. The hardware installed with operating systems, or the software nodes, which are able to run client applications represent the raw data providers. Using proper system calls, the status information concerning the software-defined network is read and embedded into packages and consequently sent to the server components to be processed. One can store snapshots of the network nodes to continuously watch that the nodes are up and ready. The availability of the nodes or of their internal resources represents a cybersecurity objective (Aldea [76]). Furthermore, by comparing two snapshots, one can detect intrusions or malfunctions of the nodes. Denial of service attacks can also be detected and stopped, if the actual parameters

feature suboptimal values when comparing the actual status of the system with the state of a previous snapshot. Additionally, the lazy (inefficient) use of resources can be detected, and further configuration changes or data-processing jobs can be generated for an improved use of the network node resources.

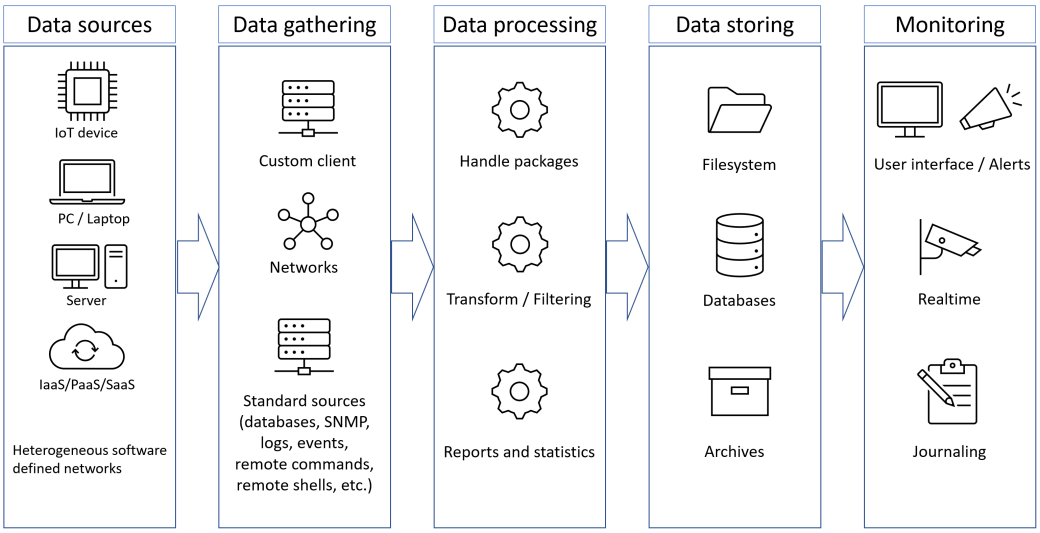

Figure 1. Overview of the proposed model.

Considering that the distributed network of nodes performs distributed and parallel computations, the platform can be customised to watch the health parameters related to the computational power of the network, such as the processing time of installed CPU units and the respective time intervals.

The platform can retain status information in the database, and further data processing can be conducted. The activity of a program or of a user can be analysed by running simple queries against the database, or business intelligence tools can be used to further analyse the status parameters related to compute nodes, services, programs, users, and physical or logical resource usages in different contexts. Even if a relational database is used, the data are properly organised, and the snapshots can be stored into files and properly archived.

The platform architecture can be extended and implemented relative to IoT systems by using secure web services and asynchronous communication to interrogate them and to enhance their usability (Aldea [77]).

The journaling feature of the proposed platform mediates the functional analysis of a distributed computation proper operation, even if the tools involved in that computation are independent and do not provide status information. Considering the operating system and platform-related parameters, all nodes of a distributed system or software-defined network can be monitored to ensure that they are fully operational.

The platform tested and described in the paper is composed of independent components that are able to communicate with each other using a custom defined protocol over TCP/IP (Figure 2). Since the networks and software-defined networks are heterogeneous, the platform-specific components were designed and developed while considering the concept of interoperability. Thus, the client components were developed using platform-independent tools and libraries. The same client subsystem is run on Linux, Windows or Raspberry Pi Linux operating systems. The client application reads status data about the machine and reports the data to the server application that were also compiled using cross-platform capabilities. Relative to the laboratory assessment, the server subsystem is also run under Fedora Linux. The communication between client subsystems and server subsystems is performed using the self-defined protocol.

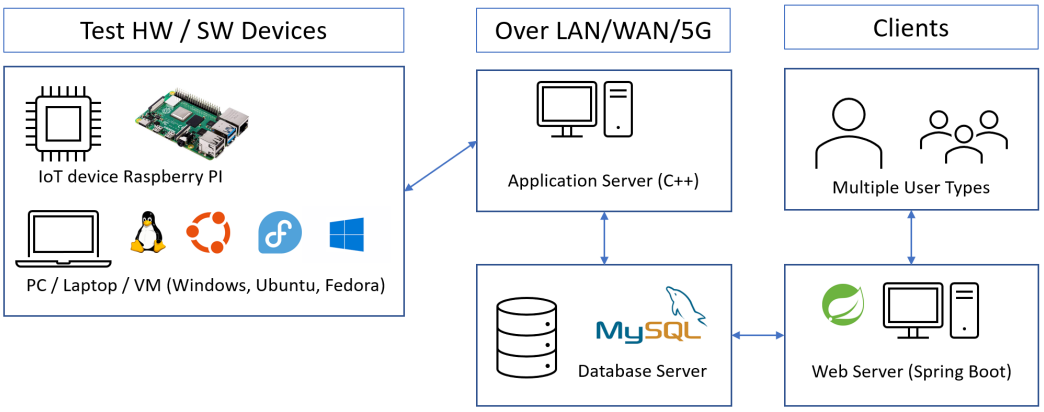

Figure 2. Test environment.

The designed platform includes the following components:

- Cross-platform desktop client subsystem. This is used to gather data and to define, monitor and manage the network. It has its own internal capabilities to retrieve CPU, memory or network status parameters. It is designed to also allow for the call to external commands and tools, and afterwards, the collected data are sent to the server application to be processed and saved.
- Cross-platform server application. This is used to collect the data. The data can be processed, and a comparison between snapshots can be made and related alerts generated. Furthermore, the data are stored in the database for further data processing, or in order to create an overview over the whole monitored network.
- The custom communication protocol is used in order to implement the client–server link. Thus, a structure was designed for the data packets, so that further functionalities or additional data gathering and reporting components can be added to the platform. The exchanged data are packed at the source and unpacked at the destination. Various types of user-defined packets are exchanged over the networks using the TCP headers, allowing the user-defined packets to be encapsulated in transport-layer security packets. These include authentication and authorisation packages, request submission packages, snapshot submission packages, etc.
- The database is used to store data and to prevent big data scalability problems using the entity relationship model.
- The web application subsystem is independent, and it is used to monitor and manage the network nodes visually. Moreover, a Spring Boot web interface was also implemented.

3.2. Client Subsystem

Each node of the network that needs to be monitored is featured with the client subsystem, which is installed (Figure 3). The client subsystem assures node integrity and security using intrusion detection and prevention mechanisms. The node sends snapshot reports to the application server, and it receives back requests regarding the monitored data. The application server modules can decide whether the client has functional problems or not. As an example, it can detect client overload states or the possible malfunction states of certain applications or jobs. Relative to handling CPU, memory, network usage or local system logs, the default client application modules are used.

Further functionalities were designed based on separate operating system tools, such as nmon, which needs to be available to be used in order to generate the required parameters and to send them to the server components for further processing.

The client application is featured with a property that allows it to be run as a daemon or service, but it also has its own graphical user interface. The interface permits local control over the current network node settings and to inspect reports and data.

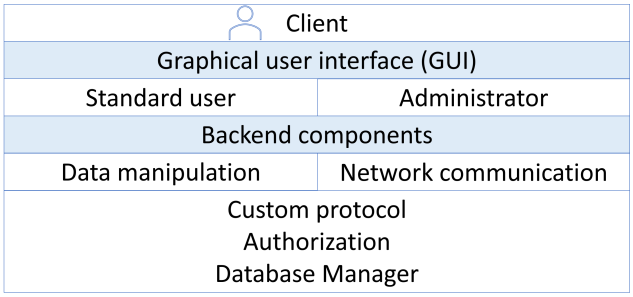

Figure 3. Client architecture.

Among other modules of the client subsystem, there are the data manipulation, network communication, custom protocol encoder and decoder, and authorisation mechanisms.

The next subsections present further details concerning the relevant software modules' functional features.

3.2.1. Data Gathering

Data can be gathered using the APIs that are installed together with the target operating system or separately. The APIs vary, and they can differ among distributions. Nonetheless, the function calls and their underlying software libraries could be installed or not. In some cases, a special middleware component is created to provide a more consistent level of compatibility, as in Tsai et al. [78].

Standard command shells or more specialised command shells allow for the creation and use of scripts to extract relevant system parameters. Thus, log files, where the operating system stores data, can also be a good source for analysing system stability relative to the overall system security. Errors and unwanted access attempts are also logged but are not always reviewed. Schauland and Jacobs [79] demonstrate how Windows applications that come preinstalled with the operating system can be used to collect data.

Furthermore, other preexisting tools, such as ps, mem, lsof, top, htop or nmon, could provide valuable information concerning the overall system sanity.

Considering other essential system resources, such as random access memory (RAM), user list or installed software, usage data can be collected directly. Nevertheless, the resources that are functionally defined over a certain period, such as central processing unit (CPU), network usage or process information, require an indirect method to be defined.

The proposed algorithm, which is considered to solve the problem relative to resources that are defined by usage over a certain period, is:

1. Gather data set 1, using one or more of the above-described methods;
2. Wait for an amount of time t;
3. Gather data set 2, using one or more of the above-described methods;
4. Compute usage.

Considering the proposed system's implementation, values between 100 and 1000 ms were used. The choice for a value that is too small or too large will generate incorrect data. As an example, regarding CPU usage, if the wait time is too low, it will report the respective data as 100% usage.

The CPU usage may be computed through the formula

$$CPU_{usage} = \Delta work\_jiffies / total\_jiffies.$$

Concerning the usage of functions that measure the CPU time in Linux, a software clock is maintained by the kernel. The software clock measures the time in jiffies. The kernel unit of the time jiffy varies across kernel versions and hardware platforms. It is declared, for example, in the header file *linux/jiffies.h*.

Equation $RX_{usage} = RX^{new}_{usage} - RX^{old}_{usage}$ represents a simplified formula for computing the number of received data packets, and RX represents the receive channel, while TX

represents the transmit channel. Thus, the difference between the current and the previous value is essential.

The parallel processing capabilities can improve the performance of a system. For this reason, it is advisable to collect data in parallel, to gain speed and to prevent potential bottlenecks.

3.2.2. Sending Data

The transmission of data to the server requires a centralised custom data structure, called a snapshot, which stores information about a set of system parameters at a certain moment in time. Following, the object is serialised and sent over the network using the custom protocol that is proposed in this paper. If a high-availability system, such as the ones that are presented by Trivedi et al. [80] and Mesbahi et al. [81], cannot be provided, it is recommended to save the serialised data locally and to send it asynchronously as soon as the data connection availability is restored.

The algorithm that is considered for the data transmission procedure consists of the following logical steps:

1. Create a snapshot of the system;
2. The snapshot is serialised;
3. Serialisation is saved locally in a file;
4. Serialisation is sent to the server using the custom protocol.

3.2.3. Authentication and Authorisation

Authorisation mechanisms can be bypassed using modern technologies (You et al. [82]). Therefore, higher levels of cybersecurity are required. The proposed solution to address this problem is to automatically verify if a machine is authorised when the client starts running, by using the current user and a combination of the ID of the operating system and the ID of a hardware component. The authentication and authorisation data are packaged into customised data packets that are sent to the server for processing (Figure 4). Note that other existing transport layer security protocols can be used to transfer custom data packets over networks for an additional layer of security.

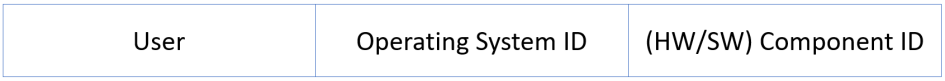

Figure 4. Authorization package.

Database management can be conducted using commands that are sent to the server using the mentioned custom network protocol. Thus, the relevant database management packages are used (Figure 5).

Figure 5. Database management package.

Network monitoring can be conducted by receiving a list of snapshots in a serialised form from the server. The data serialisations are transformed into objects, and each object can be stored into a hash table, which uses the ID of a machine as a unique key and the object as the related value. If a system is present in the hash table but it is not in the list of snapshots, the client application has stopped, which suggests that either the application has stopped working (e.g., the application closed or the proper connection could not be established), or the machine has turned off.

3.3. Server Application

The server application implements software modules for handling communication with the clients, for manipulating multiple client requests over the networks using the

designed protocol, for encoding and decoding the data packages, and for handling the necessary authorisation mechanism. Object-oriented design patterns were used in order to provide a flexible, reliable and extensible software system architecture.

The proper handling of distributed network resources poses performance problems in different types of networks. This remark is particularly valid relative to cloud native networks. An analysis related to network slicing and containerisation problems is presented by Huang et al. [83]. In terms of performance, server-to-client communication attempts to optimise the use of resources through the consideration of small data packets and through the avoidance of unnecessary polling.

3.3.1. Communication with the Client Applications

The proprietary communication protocols, which relate to the proposed integrated management and monitoring platform, are built on top of the TCP/IP protocol stack and are used as a transfer mechanism, but the actual format and structure of the data to be carried are specific to each proprietary protocol. Some examples of proprietary protocols include the Remote Desktop Protocol (RDP), used by Microsoft, and the Simple Network Management Protocol (SNMP), used for the management of network devices. These protocols may not be compatible with other systems or networks and are typically used to support specific applications or services. These protocols are not always easy to understand and can be difficult to implement. Thus, this paper describes a custom protocol which was introduced to simplify the complexity of the existing ones and to allow the users to easily adapt and extend the proposed software platform, as required, in their own environments.

Client–server communication is based on the exchange of custom messages (Figure 6). Thus, the server receives a custom package from the client, and after decoding and processing it, if necessary, the server sends back the appropriate response.

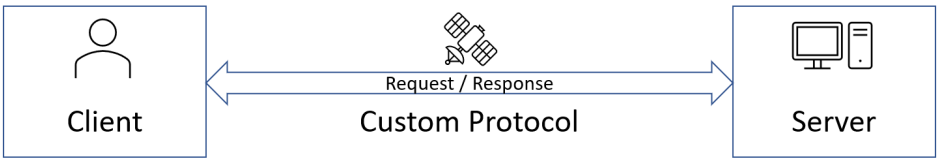

Figure 6. Client–server communication.

As soon as the client first connects to the server application, a new thread can be created for this purpose. This way, bottlenecks that can be caused by slow clients are prevented.

The proposed custom protocol is described in Section 3.4, "Communication Protocol".

3.3.2. Database

Logged data storage and the general concept of storage in the IT industry can cause significant challenges, which are determined by storage space costs and data-processing issues (Katal et al. [84]). The project-related data vary from raw collected data regarding processors, processor memory, users and other logical or physical resources, to semi-structured documents that contain snapshots with values of the collected parameters. Depending on the number of managed devices and nodes of the network, the amount of data can rapidly increase. The stored snapshot can be automatically archived or deleted, but the platform administrator must properly calibrate the snapshot's validity to avoid rapid exhaustion of the storage resources.

Concerning the presented platform functionalities, the database is required to store three categories of entity types (Figure 7):

1. System information—to store general information regarding a system;
2. Authorisation—to store information regarding how authorisation is handled;
3. Snapshots—to store information regarding the status of a system at a particular time.

Figure 7. Abstraction of database structure.

The proposed solution for the implemented platform uses a relational database (Figure 8). In addition, the data- manipulation modules handle parallel connections to store data that are received from the clients. The relational schema contain information related to the usage of the network resources. Among these, there are:

- Users of a machine or service and their logged in times;
- CPU usage of the respective compute node;
- Environment variables;
- Availability of a node or service;
- Users' roles, etc.

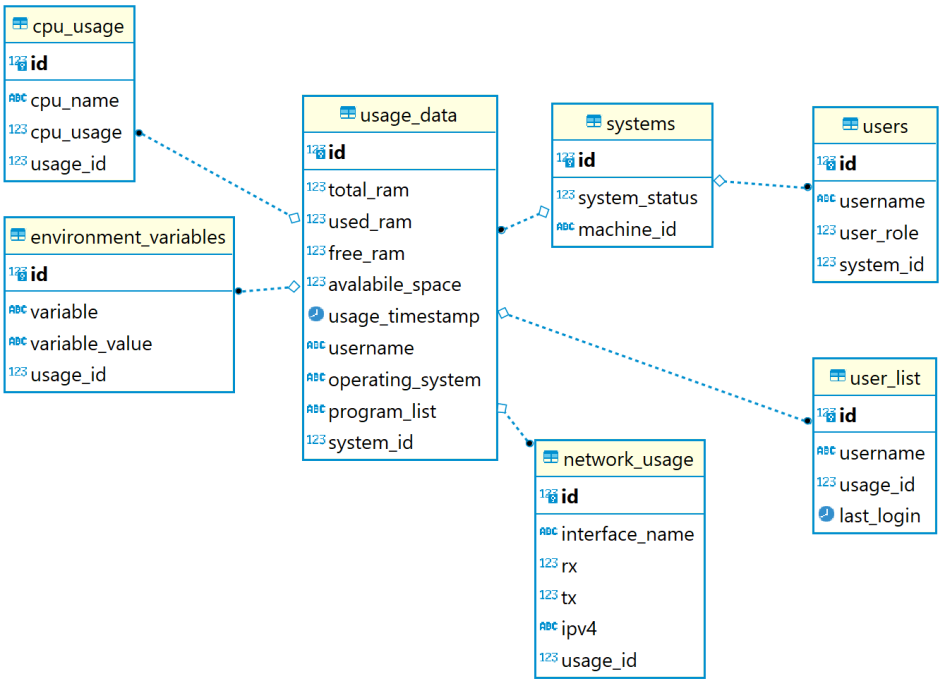

Figure 8. Database diagram.

As soon as the application is started for the first time, several initialisation tasks are conducted, such as initial database setup and creation of an administrator system and account. The server application uses a configuration file. This way, it allows the administrators to first enrol their systems into the network and to introduce other systems and users through the graphical user interface of the client application at a later time.

3.4. Communication Protocol

The enhancement of the communication between the client and the server components, relative to the proposed proof of concept, is realised using a custom network protocol at the level of the application layer (Figure 9). Considering that reliability and in-order data packet delivery are mandatory, the proposed custom protocol is based on TCP and not on UDP.

Thus, TCP works as a stream of data, so that communication based on messages can become problematic, since messages do not have a fixed size. The resolution of this problem is accomplished by a header which contains the message size.

Consequently, the following logical procedure can be considered:

1. Read the n characters from the TCP stream to obtain the message size;

2. Try to read from the TCP stream until the received data are of size n.

This assures that every message entity is fully received, and it contains only its own required data.

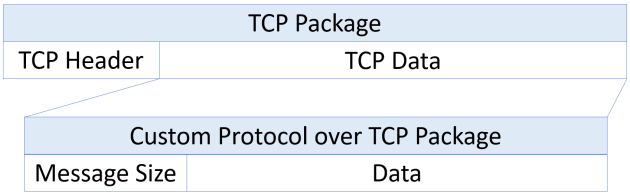

Figure 9. Package sent over the network.

In addition to being designed with security in mind, the custom messages of the communication protocol can be encapsulated within existing secure protocol packages, such as TLS. This additional security is provided by the widespread protocols, and it comes with the overhead of managing (creating, replacing, updating) the encryption keys and certificates, but at the same time, it does not affect the solutions under consideration, as they are applied at different network levels.

3.5. Web Application

The designed architecture also includes a web component which uses the same database as the one considered for the server application. The web component is an independent entity which mainly helps in using the stored data to obtain useful reports. The web application is based on the Spring Boot framework. The spring nature of the web application allows it to be run inside a web server or independently. It is also a cross-platform, and it can be deployed on different types of operating systems, or can even be properly hosted.

Besides the standard functionality as web application, which can be accessed through a web browser, the web application is also capable of exposing web services. These mediate integration with other external or internal applications.

3.5.1. RESTful Services

REST web services bring a number of advantages, such as performance, security and simplicity. The web component of the proposed interconnected system publishes REST services to ensure interoperability with other applications. Thus, it is capable of receiving and processing queries, and it exposes relevant data for reporting purposes (Figure 10).

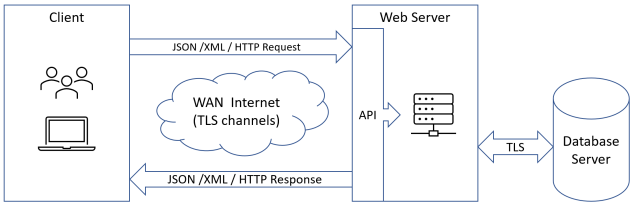

Figure 10. Web Services—Web APIs.

The security of the web services determines a consistent research subject. The reported platform uses an authorisation mechanism (Section 3.2.3), and the data privacy for the web request/response traffic is firstly assured by using transport layer security (TLS) connections between the web server and the client applications.

3.5.2. Real-Time Monitoring of a Network Node

The monitoring of a machine from a web interface can be accomplished using the method that is presented in the section "Client Application". Thus, it is feasible to collect data from the related database and embed it into a snapshot. The presentation of the relevant information to the user is performed using the web interface or the web services.

Thus, the client application helps the local monitoring of the assigned workstation to be conducted (Figure 11). Furthermore, the monitored data can be accessed using the presented web application.

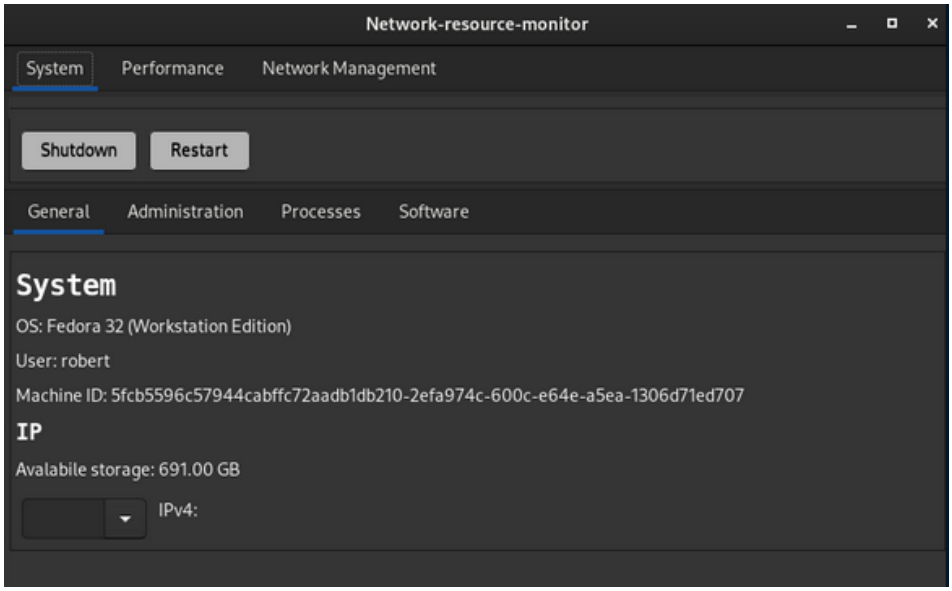

Figure 11. Network resources monitor interface.

3.5.3. Data Access

Communication between the node clients and the server component uses a custom protocol over TCP/IP. Data elements can be added, modified, removed or sent. The database layer and the software module that relies on it then use the native transport layer security encryption provided by the PostgreSQL database server to counter sniffing attacks. On the other side, the data are consumed by the graphical user interface module using RESTful requests over the secure Hyper Text Transfer Protocol (HTTPS). Mutual authentication, confidentiality and integrity are ensured through the use of custom protocol message communications over a secure socket layer.

4. Experimental Assessment and Discussion

The assessment of the designed architecture was conducted using an experimental environment, which includes one machine to support the administrative tasks, multiple virtual machines (VMs) running different distributions of GNU/Linux and Microsoft Windows that act as data providers, a Fedora Linux machine that represents an application server, and a MySQL database VM that stores related data. Relative to the web application, Java Spring Boot was used. The screen capture from Figure 12 presents the test environment with three clients, Fedora, Windows and Raspberry PI running Linux Ubuntu, which supports the assessment of the demo application that concerns the analysed concept.

The implementation of the client and server components was conducted in C++, due to its speed and ability to run cross-platform. Data gathering was performed using Linux Kernel API, WinAPI, and the tools described in Section 3.2.1. The web application relies on the Java Spring Boot framework, and the database component is based on the MySQL server.

Since the proposed concept is implemented using platform-independent technologies, the system can be deployed on different infrastructures without having to completely change the considered technologies.

The specification and implementation of proper policies and tools concerning network monitoring represents a critical aspect of network security.

The monitoring process, which is conducted using the client application, has been tested on various platforms. The application helps to report the status information, which

includes data items that range from network connectivity to processor usage. The consideration of network connectivity data enables better awareness regarding the status of nodes or services. Administrators are alerted when the status of nodes or services indicates lack of connectivity or bottlenecks, which may be indicative of denial of service attacks. The simple presence of a network connection can be used, or further machine learning techniques may be applied to the packets that are sniffed or to the parameters that are reported.

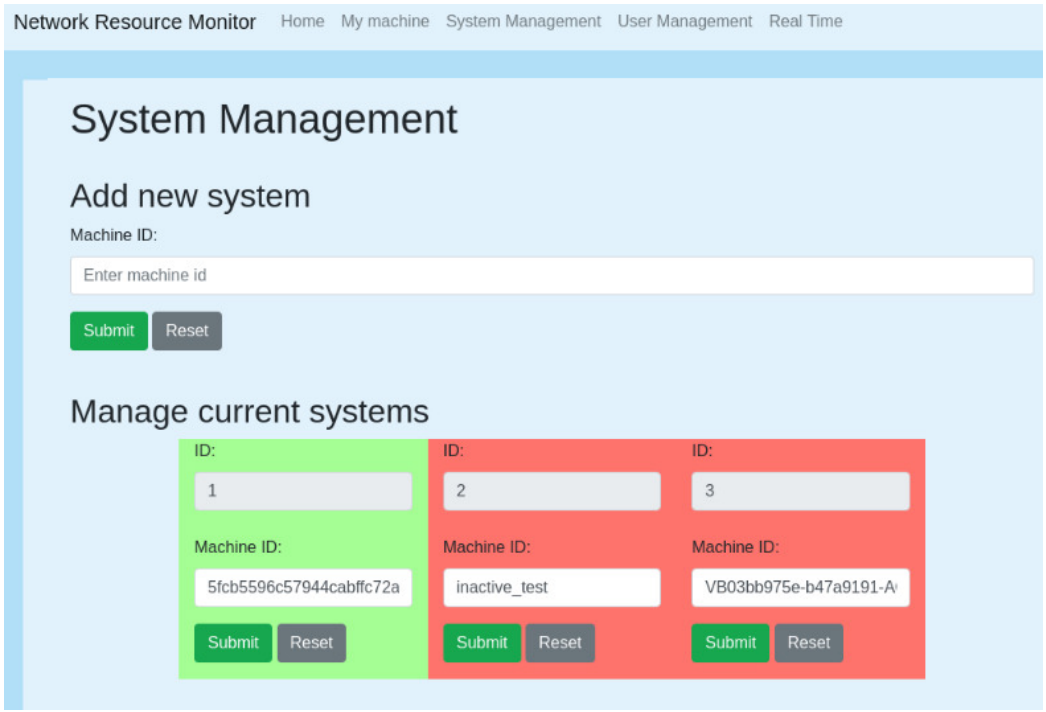

Figure 12. Test environment with three clients.

Improved availability is not the only aspect that is considered relative to network connectivity and its performance. Some missing connections or sensible resource usage may result in additional costs. The lack of redundant connections between components also affects their workload and, more or less directly, their maintenance costs. Although cost is not always a technical requirement, in most cases, it remains a non-functional requirement for all distributed systems. Each system comes with its own configuration and maintenance costs. Considering distributed systems, it is important to place load on resources correctly and to consequently minimise wasted processor time.

Network performance is a matter of debate, and the proposed platform addresses the technical aspects by using appropriate system calls and by integrating the ability to use and extract relevant information and report it to a centralised application. Consequently, the acquired data are consulted through a web interface. The proper implementation of baseline monitoring components implies that the performance degradation of the network can be accurately detected. The application creates snapshots, which can be compared or graphically displayed to observe the deviations from a stable baseline configuration of the entire logical network, on which the monitored distributed system is deployed.

Apart from the legacy security requirements, there are other regulatory compliance requirements that apply to all networks. Some regulations require proper access logs to be preserved for six months, and others require personal data to be deleted from all stored data items. These personal data privacy requirements can be implemented using the proposed integrated management and monitoring tool. The data can be inspected, and further configuration actions can be applied directly or automatically triggered as required.

The administrator that uses the proposed software system observes congestion of network links, or congestion of network nodes or services, and consequently, proper

countermeasures can be quickly applied to ensure improved performance of the overall system. The performance level of a system is determined by the performance levels of each service that it provides. Thus, a suboptimal data access service can trigger other availability problems and consequently slows down the entire system. The number of threads or processes that run on a CPU can affect the perceived compute performance. Thus, even if available network resources are at the highest possible level, the workload on the nodes should be configured and improved. There are tools and software modules which may be used in order to define and impose resource usage policies at the level of the operating system's kernel, such as control groups (cgroups, [85]). Additionally, plain commands may be issued after proper node-monitoring tasks are performed using the proposed concepts. The code of the demo implementation for the proposed interconnected system concept can be accessed in the repository at [86], while the relevant code of the web system's server and client components may be accessed in the repository at [87].

Large and scalable software solutions exist which monitor the network nodes, servers, services and other types of equipment, such as Nagios, SolarWinds, ntopng, etc. Nonetheless, particular large-scale attacks are discovered from time to time. The SolarWinds supply chain attack has affected the compilation of the platform components, and a backdoor was activated inside the digitally signed code. The research that is reported in this paper pleads for continuous improvements in the monitoring and intrusion detection software modules using separately compiled tools for watching and detecting possible intrusion patterns. Consideration of the custom-built platform, apart from the well-known monitoring tools, implies that further security checks can be independently conducted and that proper countermeasures are specified and implemented.

Currently, it is common to build large networks of IoT devices, such as smart-meter gateways fleets, computer tomography fleets, drone fleets, or even self-driving car fleets, which from a certain perspective are monitored by using the closed network monitoring tools to assure their connectivity and healthy functioning, and considering the other perspective, they are connected to proxy microservices for reporting their readings to the facility providers. The proposed platform tools for monitoring timing, quality, and quantity of exchange data can be considered to avoid bogus reporting or missing data. Thus, it is possible to detect if unnecessary reporting appears into the system. Since the data exchanges between the smart devices are usually encrypted, the proper statistics related to the security tools are more relevant if the reporting code is executed at the beginning and at the end of the communication tunnels. Consequently, the proper modification and usage of the proposed proof of concept imply that the qualitative analysis of the exchanged data can be assured and the related attacks prevented.

Thus, Table 1 presents the processor and memory deviation, which is related to the processor and memory usage of a monitored node. The deviation is not acceptable since the node only hosts and controls a device through a microservice, which conducts no planned activity, such as queries and updates, at the moment when the snapshot is requested. Consequently, a security alert must be raised. Moreover, consideration of the mentioned custom monitoring routines helps to distinguish between pop and push device data.

Table 1. Snapshot parameters comparison between threshold and actual snapshot for a host node.

Data Image	CPU Usage	Memory Usage
Threshold	5%	20%
Snapshot	8%	30%

Table 2 illustrates the received data packets and transmitted data packets for a gateway node that manages 100 IoT devices, which report the measurements every 15 minutes. Since the threshold and current snapshot are generated outside the node or software update, no extra network traffic should be present on the lines, except for the measurement reporting packages themselves. Considering the low-energy consumption requirements, no requests

are sent from the gateway node to the connected devices, as they are expected to push the values four times per hour. Usage of the proposed custom monitoring model implies that it is possible to detect malfunctioning devices and to consequently trigger online packages to wake, which can be counted and analysed separately or can use different data snapshots. In addition, depending on the analysed device, the raw data can be transported into a single package, which determines each individual measurement. The received (RX) package number can be lower and not greater than $4 \times 24 \times 100$, due to the implied network connections. Thus, a larger value could suggest replay attacks, which attempt to reproduce older values, or it may suggest initiation of a distributed denial of service attack.

Table 2. Number of packages on receive and transmit channels for a gateway node.

Data Image	RX Packages	TX Packages
Snapshot	9600	0

Consistent consideration of customisable network-monitoring approaches may also provide a precise overview concerning unauthorised access attempts. As an example, there are attacks that send many requests per unit of time, but there are also more sophisticated attacks that send few malware requests, and they are not easily detected. Network traffic monitoring and the requests classified at the level of the compute nodes allow for complex event-processing techniques to be designed and implemented. Consequently, the overall cybersecurity of the networked structure is intelligently enhanced.

Monolithic or large closed solutions may prove difficult to securely operate in certain environments. As an example, the SolarWinds Hack [88] proved that a comprehensive monitoring solution, which does not offer the possibility to proactively customise critical functional code sections, may be prone to catastrophic security breaches. Consequently, the solution is represented by a decoupled and modular architecture, such as the one that is presented in this paper, which may be efficiently tweaked and enhanced considering both security and performance aspects.

5. Conclusions

This paper describes an integrated real-time network-monitoring and management approach which relates to hardware and software resources in heterogeneous computer networks. This paper presents the related existing contributions, and it consequently describes the proposed monitoring and management solutions. Moreover, it was implemented and deployed considering a real-world use case, which was also described. The experimental assessment demonstrates that the proposed solution is capable of efficiently monitoring and managing the intended networked structures.

This article also analyses the concept of network security, with a significant focus on distributed system nodes, IoT devices, and cloud-based deployments. The architectural model described in this paper fully addresses the functional and logical requirements of the three use cases, and the validity of the proposed model is justified by the described experimental assessment. Furthermore, this model is highly adaptable, considering that it is platform independent. Moreover, the proposed system architecture is suitable for the detection of a wide range of security patterns. This allows for an easy and efficient integration of the described solution into the target infrastructures. This relieves the resource monitoring and management of software-defined networked infrastructures. The monitoring agents report the working parameters as snapshots. The comparative analysis of the snapshots is proper for the implementation of efficient cyberattack detection and mitigation mechanisms.

Several benefits related to the general problem of network monitoring have been covered in this paper. Thus, operational cost savings, increased availability, improved performance and security, and baseline monitoring represent some of the cybersecurity policy components, or countermeasures, which are built into the proposed integrated system.

The described architecture may be further customised and extended considering the current state-of-the-art monitoring, network-management approaches and defence strategies, which relate to future threat-hunting methodologies. The proposed modular architectural structure allows for efficient subsequent maintenance and development of the implemented solution, which will be continuously improved upon in the future.

Author Contributions: Conceptualisation, C.L.A. and R.B.; methodology, C.L.A.; software, C.L.A. and R.N.S.; validation, C.L.A., R.B. and R.N.S.; formal analysis, R.B.; investigation, R.B. and R.N.S.; resources, R.B.; data curation, C.L.A.; writing—original draft preparation, C.L.A. and R.N.S.; writing—review and editing, C.L.A., R.B. and R.N.S.; visualisation, R.B.; supervision, C.L.A.; project administration, C.L.A.; funding acquisition, R.B. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Not applicable.

Conflicts of Interest: The authors declare no conflict of interest.

Abbreviations

The following abbreviations are used in this manuscript:

API	Application Programming Interface
CNC	Computer Numerical Control
CPU	Central Processing Unit
C-RAN	Cloud Radio Access Network
DNM	Distributed Network Monitoring
eCAL	enhanced Communication Abstraction Layer
FPGA	Field-Programmable Gate Array
GPU	Graphics Processing Unit
HPC	High-performance computing
HTTPS	Hypertext Transfer Protocol Secure
IDS	Intrusion Detection System
IoT	Internet of Things
IPS	Intrusion Prevention System
ITS	Intelligent Transportation System
MQTT	Message Queue Telemetry Transport
NFV	Network Functions Virtualisation
NIST	National Institute of Standards and Technology
NTMA	Network Traffic Monitoring and Analysis
OPC	Open Platform Communications
OPC UA	OPC Unified Architecture
SLO	Service Level Objectives
SNMP	Simple Network Management Protocol
SVM	Support Vector Machine
QUIC	Quick UDP Internet Connections
RAM	Random Access Memory
RDP	Remote Desktop Protocol
TCP	Transmission Control Protocol
TLS	Transport Layer Security
UAV	Unmanned Aerial Vehicles
UDP	User Datagram Protocol
UPF	User Plane Function
VM	Virtual Machine
WSN	Wireless Sensor Networks

References

1. Common Vulnerabilities and Exposures (CVE). Available online: https://cve.mitre.org (accessed on 18 January 2023).
2. Common Weakness Enumeration (CWE). Available online: https://cwe.mitre.org (accessed on 18 January 2023).
3. Yadav, T.; Rao, A.M. Technical aspects of cyber kill chain. In Proceedings of the International Symposium on Security in Computing and Communication, Kochi, India, 10–13 August 2015; pp. 438–452. [CrossRef]
4. Fenz, S.; Heurix, J.; Neubauer, T.; Pechstein, F. Current challenges in information security risk management. *Inf. Manag. Comput. Secur.* **2014**, *22*, 410–430. [CrossRef]
5. Mehnen, J.; He, H.; Tedeschi, S.; Tapoglou, N. Practical Security Aspects of the Internet of Things. In *Cybersecurity for Industry 4.0*; Springer International Publishing: Cham, Switzerland, 2017; pp. 225–242. [CrossRef]
6. Manogaran, G.; Thota, C.; Lopez, D.; Sundarasekar, R. Big Data Security Intelligence for Healthcare Industry 4.0. In *Cybersecurity for Industry 4.0*; Springer International Publishing: Cham, Switzerland, 2017; pp. 103–126. [CrossRef]
7. Wu, Q.; Ferebee, D.; Lin, Y.; Dasgupta, D. An integrated cyber security monitoring system using correlation-based techniques. In Proceedings of the 2009 IEEE International Conference on System of Systems Engineering (SoSE), Albuquerque, NM, USA, 30 May–3 June 2009; pp. 1–6.
8. Lee, S.; Levanti, K.; Kim, H.S. Network monitoring: Present and future. *Comput. Netw.* **2014**, *65*, 84–98. [CrossRef]
9. Montes, J.; Sánchez, A.; Memishi, B.; Pérez, M.S.; Antoniu, G. GMonE: A complete approach to cloud monitoring. *Future Gener. Comput. Syst.* **2013**, *29*, 2026–2040. [CrossRef]
10. Aceto, G.; Botta, A.; de Donato, W.; Pescapè, A. Cloud monitoring: A survey. *Comput. Netw.* **2013**, *57*, 2093–2115. [CrossRef]
11. Chang, C.C.; Yang, S.R.; Yeh, E.H.; Lin, P.; Jeng, J.Y. A Kubernetes-Based Monitoring Platform for Dynamic Cloud Resource Provisioning. In Proceedings of the GLOBECOM 2017—2017 IEEE Global Communications Conference, Singapore, 4–8 December 2017; pp. 1–6. [CrossRef]
12. Kshirsagar, D.; Patil, A. Blackhole attack detection and prevention by real time monitoring. In Proceedings of the 2013 Fourth International Conference on Computing, Communications and Networking Technologies (ICCCNT), Tiruchengode, India, 4–6 July 2013; pp. 1–5. [CrossRef]
13. Gentry, C. *A Fully Homomorphic Encryption Scheme*; Stanford University: Stanford, CA, USA, 2009.
14. Carlucci, G.; Cicco, L.D.; Mascolo, S. HTTP over UDP: An Experimental Investigation of QUIC. In Proceedings of the 30th Annual ACM Symposium on Applied Computing, Salamanca, Spain, 13–17 April 2015; pp. 609–614. [CrossRef]
15. Yan, L.; Cao, S.; Gong, Y.; Han, H.; Wei, J.; Zhao, Y.; Yang, S. SatEC: A 5G Satellite Edge Computing Framework Based on Microservice Architecture. *Sensors* **2019**, *19*, 831. [CrossRef]
16. de Jesus Martins, R.; Dalla-Costa, A.G.; Wickboldt, J.A.; Granville, L.Z. SWEETEN: Automated Network Management Provisioning for 5G Microservices-Based Virtual Network Functions. In Proceedings of the 2020 16th International Conference on Network and Service Management (CNSM), Izmir, Turkey, 2–6 November 2020; pp. 1–9. [CrossRef]
17. Hossain, M.F.; Mahin, A.U.; Debnath, T.; Mosharrof, F.B.; Islam, K.Z. Recent research in cloud radio access network (C-RAN) for 5G cellular systems—A survey. *J. Netw. Comput. Appl.* **2019**, *139*, 31–48. [CrossRef]
18. Hsieh, C.Y.; Chang, Y.W.; Chen, C.; Chen, J.C. Design and implementation of a generic 5G user plane function development framework. In Proceedings of the 27th Annual International Conference on Mobile Computing and Networking, New Orleans, LA, USA, 25–29 October 2021; pp. 846–848. [CrossRef]
19. Xinjian, O.; Jingjing, L.; Chaofeng, C.; Zilin, Y.; Xiang, L.; Shukai, H. Research on 5G Microservices Capability Open Architecture and Deterministic Bearing Technology. In Proceedings of the 2021 IEEE 21st International Conference on Communication Technology (ICCT), Tianjin, China, 13–16 October 2021; pp. 492–496. [CrossRef]
20. Aftab, M.A.; Hussain, S.S.; Ali, I.; Ustun, T.S. IEC 61850 based substation automation system: A survey. *Int. J. Electr. Power Energy Syst.* **2020**, *120*, 106008. [CrossRef]
21. Akyildiz, I.F.; Wang, P.; Lin, S.C. SoftAir: A software defined networking architecture for 5G wireless systems. *Comput. Netw.* **2015**, *85*, 1–18. [CrossRef]
22. Xia, X.; Xu, K.; Wang, Y.; Xu, Y. A 5G-Enabling Technology: Benefits, Feasibility, and Limitations of In-Band Full-Duplex mMIMO. *IEEE Veh. Technol. Mag.* **2018**, *13*, 81–90. [CrossRef]
23. Khalfi, B.; Hamdaoui, B.; Guizani, M. Extracting and Exploiting Inherent Sparsity for Efficient IoT Support in 5G: Challenges and Potential Solutions. *IEEE Wirel. Commun.* **2017**, *24*, 68–73. [CrossRef]
24. Xu, L.; Collier, R.; O'Hare, G.M.P. A Survey of Clustering Techniques in WSNs and Consideration of the Challenges of Applying Such to 5G IoT Scenarios. *IEEE Internet Things J.* **2017**, *4*, 1229–1249. [CrossRef]
25. Spyridis, Y.; Lagkas, T.; Sarigiannidis, P.; Argyriou, V.; Sarigiannidis, A.; Eleftherakis, G.; Zhang, J. Towards 6G IoT: Tracing Mobile Sensor Nodes with Deep Learning Clustering in UAV Networks. *Sensors* **2021**, *21*, 3936. [CrossRef] [PubMed]
26. Gholami, A.; Rao, K.; Hsiung, W.P.; Po, O.; Sankaradas, M.; Chakradhar, S. ROMA: Resource Orchestration for Microservices-based 5G Applications. In Proceedings of the NOMS 2022-2022 IEEE/IFIP Network Operations and Management Symposium, Budapest, Hungary, 25–29 April 2022; pp. 1–9. [CrossRef]
27. Guija, D.; Siddiqui, M.S. Identity and Access Control for micro-services based 5G NFV platforms. In Proceedings of the 13th International Conference on Availability, Reliability and Security, Hamburg, Germany, 27–30 August 2018; pp. 1–10. [CrossRef]

28. Soenen, T.; Rossem, S.V.; Tavernier, W.; Vicens, F.; Valocchi, D.; Trakadas, P.; Karkazis, P.; Xilouris, G.; Eardley, P.; Kolometsos, S.; et al. Insights from SONATA: Implementing and integrating a microservice-based NFV service platform with a DevOps methodology. In Proceedings of the NOMS 2018—2018 IEEE/IFIP Network Operations and Management Symposium, Taipei, Taiwan, 23–27 April 2018; pp. 1–6. [CrossRef]
29. Orduz, J.S.; Orozco, G.D.; Tobar-Arteaga, C.H.; Rendon, O.M.C. µvIMS: A Finer-Scalable Architecture Based on Microservices. In Proceedings of the 2019 IEEE 44th LCN Symposium on Emerging Topics in Networking (LCN Symposium), Osnabrueck, Germany, 14–17 October 2019; pp. 141–148. [CrossRef]
30. Lee, G.M.; Um, T.W.; Choi, J.K. AI AS A MICROSERVICE (AIMS) OVER 5G NETWORKS. In Proceedings of the 2018 ITU Kaleidoscope: Machine Learning for a 5G Future (ITU K), Santa Fe, Argentina, 26–28 November 2018; pp. 1–7. [CrossRef]
31. Prabadevi, B.; Deepa, N.; Pham, Q.V.; Nguyen, D.C.; Reddy, T.; Pathirana, P.N.; Dobre, O. Toward Blockchain for Edge-of-Things: A New Paradigm, Opportunities, and Future Directions. *IEEE Internet Things Mag.* **2021**, *4*, 102–108. [CrossRef]
32. Gayialis, S.P.; Kechagias, E.; Papadopoulos, G.A.; Konstantakopoulos, G.D. Design of a Blockchain-Driven System for Product Counterfeiting Restraint in the Supply Chain. In *IFIP Advances in Information and Communication Technology*; Springer International Publishing: Cham, Switzerland, 2019; pp. 474–481. [CrossRef]
33. Salhab, N.; Rahim, R.; Langar, R. NFV Orchestration Platform for 5G over On-the-fly Provisioned Infrastructure. In Proceedings of the IEEE INFOCOM 2019—IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS), Paris, France, 29 April–2 May 2019; pp. 971–972. [CrossRef]
34. Costache, C.; Machidon, O.; Mladin, A.; Sandu, F.; Bocu, R. Software-defined networking of Linux containers. In Proceedings of the 2014 RoEduNet Conference 13th Edition: Networking in Education and Research Joint Event RENAM 8th Conference, Chisinau, Moldova, 11–13 September 2014; pp. 1–4. [CrossRef]
35. Mijumbi, R.; Serrat, J.; Luis Gorricho, J.; Latre, S.; Charalambides, M.; Lopez, D. Management and orchestration challenges in network functions virtualization. *IEEE Commun. Mag.* **2016**, *54*, 98–105. [CrossRef]
36. Rao, K.; Coviello, G.; Hsiung, W.P.; Chakradhar, S. ECO: Edge-Cloud Optimization of 5G applications. In Proceedings of the 2021 IEEE/ACM 21st International Symposium on Cluster, Cloud and Internet Computing (CCGrid), Melbourne, Australia, 10–13 May 2021; pp. 649–659. [CrossRef]
37. Nadaf, S.M.; Behera, S.; Rath, H.K.; Mishra, G.; Mukhopadhyay, R.; Patro, S. Microservices-Based Provisioning and Control of Network Services for Heterogeneous Networks. *Int. J. Comput. Syst. Eng.* **2022**, *16*, 89–97.
38. Alencar, D.; Both, C.; Antunes, R.; Oliveira, H.; Cerqueira, E.; Rosario, D. Dynamic Microservice Allocation for Virtual Reality Distribution With QoE Support. *IEEE Trans. Netw. Serv. Manag.* **2022**, *19*, 729–740. [CrossRef]
39. Luo, G.; Yuan, Q.; Li, J.; Wang, S.; Yang, F. Artificial Intelligence Powered Mobile Networks: From Cognition to Decision. *IEEE Netw.* **2022**, *36*, 136–144. [CrossRef]
40. Kaur, K.; Guillemin, F.; Rodriguez, V.Q.; Sailhan, F. Latency and network aware placement for cloud-native 5G/6G services. In Proceedings of the 2022 IEEE 19th Annual Consumer Communications & Networking Conference (CCNC), Las Vegas, NV, USA, 8–11 January 2022; pp. 114–119. [CrossRef]
41. Yan, M.; Liang, X.; Lu, Z.; Wu, J.; Zhang, W. HANSEL: Adaptive horizontal scaling of microservices using Bi-LSTM. *Appl. Soft Comput.* **2021**, *105*, 107216. [CrossRef]
42. Hannousse, A.; Yahiouche, S. Securing microservices and microservice architectures: A systematic mapping study. *Comput. Sci. Rev.* **2021**, *41*, 100415. [CrossRef]
43. Huang, Z.; Friderikos, V.; Dohler, M.; Aghvami, H. Granular VNF-Based Microservices. In *Design Innovation and Network Architecture for the Future Internet*; IGI Global: Hershey, PA, USA, 2021; pp. 250–271. [CrossRef]
44. Sheoran, A.; Fahmy, S.; Cao, L.; Sharma, P. AI-Driven Provisioning in the 5G Core. *IEEE Internet Comput.* **2021**, *25*, 18–25. [CrossRef]
45. Mateo-Fornés, J.; Pagès-Bernaus, A.; Plà-Aragonés, L.M.; Castells-Gasia, J.P.; Babot-Gaspa, D. An Internet of Things Platform Based on Microservices and Cloud Paradigms for Livestock. *Sensors* **2021**, *21*, 5949. [CrossRef] [PubMed]
46. Ying, F.; Zhao, S.; Deng, H. Microservice Security Framework for IoT by Mimic Defense Mechanism. *Sensors* **2022**, *22*, 2418. [CrossRef] [PubMed]
47. Bocu, R.; Costache, C. A homomorphic encryption-based system for securely managing personal health metrics data. *IBM J. Res. Dev.* **2018**, *62*, 1:1–1:10. [CrossRef]
48. Pruna, S.; Vasilescu, A. FitPi: Wearable IoT solution for a daily smart life. *Int. J. Adv. Stat. IT&C Econ. Life Sci.* **2020**, *10*, 67–79. [CrossRef]
49. Kim, S.; Kim, J.; Kim, M.J.; Jung, W.; Kim, J.; Rhu, M.; Ahn, J.H. Bts: An accelerator for bootstrappable fully homomorphic encryption. In Proceedings of the 49th Annual International Symposium on Computer Architecture, New York, NY, USA, 18–22 June 2022; pp. 711–725. [CrossRef]
50. Stanimirović, A.; Bogdanović, M.; Frtunić, M.; Stoimenov, L. Low-voltage electricity network monitoring system: Design and production experience. *Int. J. Distrib. Sens. Netw.* **2020**, *16*, 1550147720903629. [CrossRef]
51. Colace, F.; Khan, M.; Lombardi, M.; Santaniello, D. A multigraph approach for supporting computer network monitoring systems. In Proceedings of the Fifth International Congress on Information and Communication Technology, London, UK, 20–21 February 2020; pp. 470–477.

52. Hohemberger, R.; Lorenzon, A.F.; Rossi, F.; Luizelli, M.C. Optimizing Distributed Network Monitoring for NFV Service Chains. *IEEE Commun. Lett.* **2019**, *23*, 1332–1336. [CrossRef]
53. Shen, S.H. An Efficient Network Monitor for SDN Networks. *ACM SIGMETRICS Perform. Eval. Rev.* **2019**, *46*, 95–96. [CrossRef]
54. Abbasi, M.; Shahraki, A.; Taherkordi, A. Deep Learning for Network Traffic Monitoring and Analysis (NTMA): A Survey. *Comput. Commun.* **2021**, *170*, 19–41. [CrossRef]
55. Li, X.; Zhao, N.; Jin, R.; Liu, S.; Sun, X.; Wen, X.; Wu, D.; Zhou, Y.; Guo, J.; Chen, S.; et al. Internet of Things to network smart devices for ecosystem monitoring. *Sci. Bull.* **2019**, *64*, 1234–1245. [CrossRef]
56. Clot, B.; Gilge, S.; Hajkova, L.; Magyar, D.; Scheifinger, H.; Sofiev, M.; Bütler, F.; Tummon, F. The EUMETNET AutoPollen programme: Establishing a prototype automatic pollen monitoring network in Europe. *Aerobiologia* **2020**, 1–9. [CrossRef]
57. Mahajan, S.R.H.; Kotecha, K. Prediction of Network Traffic in Wireless Mesh Networks Using Hybrid Deep Learning Model. *IEEE Access* **2022**, *10*, 7003–7015. [CrossRef]
58. MahmoudZadeh, S.; Yazdani, A.; Elmi, A.; Abbasi, A.; Ghanooni, P. Exploiting a fleet of UAVs for monitoring and data acquisition of a distributed sensor network. *Neural Comput. Appl.* **2022**, *34*, 5041–5054. [CrossRef]
59. Ali, G.; Hass, J.; Sill, A.; Hojati, E.; Dang, T.; Chen, Y. Redfish-Nagios: A Scalable Out-of-Band Data Center Monitoring Framework Based on Redfish Telemetry Model. In Proceedings of the Fifth International Workshop on Systems and Network Telemetry and Analytics, Minneapolis, MN, USA, 30 June 2022; pp. 3–11. [CrossRef]
60. Redfish Telemetry White Paper. 2020. Available online: https://www.dmtf.org/sites/default/files/standards/documents/DSP2051_1.0.0.pdf (accessed on 23 January 2023).
61. Sofi, A.; Regita, J.J.; Rane, B.; Lau, H.H. Structural health monitoring using wireless smart sensor network—An overview. *Mech. Syst. Signal Process.* **2022**, *163*, 108113. [CrossRef]
62. Aouini, Z.; Pekar, A. NFStream: A flexible network data analysis framework. *Comput. Netw.* **2022**, *204*, 108719. [CrossRef]
63. Yin, L.; Zhang, D. The Calculation Method of the Network Security Probability of the Multi-rail Division Based on Fuzzy Inference. *Mob. Networks Appl.* **2022**, *27*, 1368–1377. [CrossRef]
64. Tran, M.Q.; Elsisi, M.; Liu, M.K.; Vu, V.Q.; Mahmoud, K.; Darwish, M.M.F.; Abdelaziz, A.Y.; Lehtonen, M. Reliable Deep Learning and IoT-Based Monitoring System for Secure Computer Numerical Control Machines Against Cyber-Attacks With Experimental Verification. *IEEE Access* **2022**, *10*, 23186–23197. [CrossRef]
65. Sengan, S.; Khalaf, O.I.; Rao, G.R.K.; Sharma, D.K.; Amarendra, K.; Hamad, A.A. Security-aware routing on wireless communication for E-health records monitoring using machine learning. *Int. J. Reliab. Qual. E-Healthc. (IJRQEH)* **2022**, *11*, 1–10. [CrossRef]
66. Qi, W.; Su, H. A Cybertwin Based Multimodal Network for ECG Patterns Monitoring Using Deep Learning. *IEEE Trans. Ind. Informatics* **2022**, *18*, 6663–6670. [CrossRef]
67. Yu, L.; Zwetsloot, I.M.; Stevens, N.T.; Wilson, J.D.; Tsui, K.L. Monitoring dynamic networks: A simulation-based strategy for comparing monitoring methods and a comparative study. *Qual. Reliab. Eng. Int.* **2022**, *38*, 1226–1250. [CrossRef]
68. Ageyev, D.; Radivilova, T.; Mulesa, O.; Bondarenko, O.; Mohammed, O. Traffic Monitoring and Abnormality Detection Methods for Decentralized Distributed Networks. In *Information Security Technologies in the Decentralized Distributed Networks*; Springer: Cham, Switzerland, 2022; pp. 287–305. [CrossRef]
69. Lu, Z.; Wang, N.; Yang, C. A novel iterative identification based on the optimised topology for common state monitoring in wireless sensor networks. *Int. J. Syst. Sci.* **2022**, *53*, 25–39. [CrossRef]
70. Alsharif, M.H.; Jahid, A.; Kelechi, A.H.; Kannadasan, R. Green IoT: A Review and Future Research Directions. *Symmetry* **2023**, *15*, 757. [CrossRef]
71. Mahnke, W.; Leitner, S.H.; Damm, M. *OPC Unified Architecture*; Springer: Berlin/Heidelberg, Germany, 2009. [CrossRef]
72. Pardo-Castellote, G. OMG data-distribution service: Architectural overview. In Proceedings of the 23rd International Conference on Distributed Computing Systems Workshops, Providence, RI, USA, 19–22 May 2003; pp. 200–206.
73. eCAL—Enhanced Communication Abstraction Layer. A Fast Publish-Subscribe Cross-Platform Middleware Using Shared Memory and UDP. Available online: https://github.com/continental/ecal (accessed on 6 April 2022).
74. Wu, M.Y.; Ke, C.K.; Lai, S.C. Optimizing the Routing of Urban Logistics by Context-Based Social Network and Multi-Criteria Decision Analysis. *Symmetry* **2022**, *14*, 1811. [CrossRef]
75. Marcos-Pablos, S.; García-Peñalvo, F. Technological Ecosystems in Care and Assistance: A Systematic Literature Review. *Sensors* **2019**, *19*, 708. [CrossRef] [PubMed]
76. Aldea, C.L. *Elemente de Securitate a Datelor in Retele de Calculatoare*; Transilvania University of Brasov Publishing House: Brasov, Romania, 2010.
77. Aldea, C.L. RESTEasy JAX-RS Login Web Service and Android Client. *Bull. Transilv. Univ. Brasov Math. Inform. Phys. Ser. III* **2014**, *7*, 81–94.
78. Tsai, C.C.; Jain, B.; Abdul, N.A.; Porter, D.E. A study of modern Linux API usage and compatibility. In Proceedings of the Eleventh European Conference on Computer Systems, London, UK, 18–21 April 2016; pp. 1–16. [CrossRef]
79. Schauland, D.; Jacobs, D. Managing the Windows Event Log. In *Troubleshooting Windows Server with PowerShell*; Apress: New York, NY, USA, 2016; pp. 17–33. [CrossRef]

80. Trivedi, K.S.; Vasireddy, R.; Trindade, D.; Nathan, S.; Castro, R. Modeling High Availability. In Proceedings of the 2006 12th Pacific Rim International Symposium on Dependable Computing (PRDC'06), Riverside, CA, USA, 18–20 December 2006, pp. 154–164. [CrossRef]
81. Mesbahi, M.R.; Rahmani, A.M.; Hosseinzadeh, M. Reliability and high availability in cloud computing environments: A reference roadmap. *Hum. Centric Comput. Inf. Sci.* **2018**, *8*, 20. [CrossRef]
82. You, P.; Peng, Y.; Liu, W.; Xue, S. Security Issues and Solutions in Cloud Computing. In Proceedings of the 2012 32nd International Conference on Distributed Computing Systems Workshops, Macau, China, 18–21 June 2012; pp. 573–577. [CrossRef]
83. Huang, S.Y.; Chen, C.Y.; Chen, J.Y.; Chao, H.C. A Survey on Resource Management for Cloud Native Mobile Computing: Opportunities and Challenges. *Symmetry* **2023**, *15*, 538. [CrossRef]
84. Katal, A.; Wazid, M.; Goudar, R.H. Big data: Issues, challenges, tools and Good practices. In Proceedings of the 2013 Sixth International Conference on Contemporary Computing (IC3), Noida, India, 8–10 August 2013; pp. 404–409. [CrossRef]
85. Introduction to Control Groups (cgroups). Available online: https://access.redhat.com/documentation/en-us/red_hat_enterprise_linux/6/html/resource_management_guide/ch01 (accessed on 23 January 2023).
86. Solca, R.N.; Aldea, C.L. Network Resource Monitor: Cross Platform Network Resource Monitor. Available online: https://github.com/q1e123/Network-resource-monitor (accessed on 24 January 2023).
87. Solca, R.N.; Aldea, C.L. Network Resource Monitor Web Application: Web App For Network Resource Monitor. Available online: https://github.com/q1e123/Network-resource-monitor-web (accessed on 24 January 2023).
88. Reuters. SolarWinds Hack Was 'Largest and Most Sophisticated Attack' Ever: Microsoft President. Available online: https://www.reuters.com/article/us-cyber-solarwinds-microsoft-idUSKBN2AF03R (accessed on 2 May 2023).