Toward Autonomic Internet of Things: Recent Advances, Evaluation Criteria, and Future Research Directions

Qazi Mamoon Ashraf, Mohammad Tahir, *Senior Member, IEEE*, Mohamed Hadi Habaebi, *Senior Member, IEEE*, and Jouni Isoaho

***Abstract*—With the rise of the Internet of Things (IoT), tiny devices capable of computation and data transmission are being deployed across various technological domains. Due to the wide deployment of these devices, manual setup and management are infeasible and inefficient. To address this inefficiency, intelligent procedures must be established to enable autonomy that allows devices and networks to operate efficiently with minimal human intervention. In the traditional client–server paradigm, autonomic computing has been proven effective in minimizing user intervention in computer systems management and will benefit IoT networks. However, IoT networks tend to be heterogeneous, distributed, and resource constrained, mandating the need for new approaches to implement autonomic principles compared to traditional approaches. We begin by introducing the basic principles of autonomic computing and its significance in IoT. We then discuss the self-* paradigm and monitor, analyze, plan, and execute (MAPE) loop from an IoT perspective, followed by recent works in IoT and key enabling technologies for enabling autonomic properties in IoT. Based on the self-* paradigm and MAPE loop analysis from the existing literature, we propose a set of qualitative characteristics for evaluating the autonomy of the IoT network. Finally, we provide a comprehensive list of challenges associated with achieving autonomic IoT and directions for future research.**

***Index Terms*—Artificial intelligence (AI), autonomic computing, blockchain, edge computing, Internet of Things (IoT), machine learning (ML), self-* paradigm.**

I. Introduction

THE Internet of Things (IoT) is a network of objects connected to the Internet using various protocols that are traditionally not connected. The goal is to exchange information and communicate to achieve various objectives, such as monitoring, tracking and management, depending on the use case. Usually, IoT objects (referred to as devices or things) in IoT are low-cost yet robust devices capable of monitoring, communication, data processing, and, in some cases, actuation. These smart devices use diverse communication technologies and protocols to achieve network-specific and common goals. The number of IoT devices is on the rise and will become an integral part of various applications in the future. IoT will affect many domains ranging from large industrial systems to tiny devices for healthcare. As a result, a wide range of applications are being developed and deployed. Since IoT involves the deployment of a large number of devices, it faces challenges in terms of management and system design, communication, resource allocation, system modeling, networking algorithms, business models, and interoperability [1], [2]. Any management system for the IoT should consider several factors that set it apart from the traditional Internet as more smart devices come online [3]. To manage various operations in the IoT, several interesting research works have been proposed describing various frameworks, middleware options, models, and protocols. However, handling IoT networks still presents significant difficulties, including security, device heterogeneity, and dynamic resource optimization.

A. Motivation

Increasing pervasiveness and adoption of connected devices in daily life give rise to novel challenges. It is extremely challenging to manage IoT due to the following reasons.

1) Smart devices are often remotely located and unattended. These smart devices, realizing the vision of ubiquitous computing, are deployed in a distributed manner and influence how we interact with our surroundings. Due to this reason, the troubleshooting and manual maintenance of devices located in the field is often challenging.
2) With new issues of technological interoperability emerging, deployed IoT systems are becoming more complex. The complexity of integrating heterogeneous devices keeps increasing as more vendors enter the IoT landscape. In due course, human management of the deployed smart devices will become too complex, rendering manual setup, management, and maintenance inefficient and expensive.

Manuscript received 13 March 2023; revised 5 May 2023; accepted 29 May 2023. Date of publication 12 June 2023; date of current version 8 August 2023. *(Corresponding author: Mohammad Tahir.)*

Qazi Mamoon Ashraf is with the Department of Research and Innovation, Telekom Malaysia Research and Development, Cyberjaya 63000, Malaysia (e-mail: mamoon@tmrnd.com.my).

Mohammad Tahir and Jouni Isoaho are with the Department of Computing, University of Turku, 20014 Turku, Finland (e-mail: tahir.mohammad@utu.fi; jisoaho@utu.fi).

Mohamed Hadi Habaebi is with the Department of Electrical and Computer Engineering, International Islamic University Malaysia, Kuala Lumpur 53100, Malaysia (e-mail: habaebi@iium.edu.my).

Digital Object Identifier 10.1109/JIOT.2023.3285359

3) Issues related to privacy and security have become a matter of concern due to increased attacks targeted at resource-constrained devices.
4) Difficulty in using existing standards and technologies for IoT. For example, established algorithms to secure network communication in IoT do not work well due to several limitations.

Therefore, it is necessary to develop automated and intelligent connected device management systems to address the aforementioned issues. One such approach that can reduce user involvement in the management of the IoT ecosystem is autonomic computing. IBM introduced the idea of autonomic computing to define a structural framework that would facilitate the management of computer systems [4]. The term "autonomic system" or "autonomic computing" refers to the idea that computer and software systems could operate similarly to the autonomic nervous system. This biological notion refers to a system that provides automatic response and activity in biological processes. An autonomic computing system can manage itself by executing the required actions in a fully autonomous manner. The idea of autonomic computing effectively reduces user intervention in computer system management. Autonomic computing has evolved beyond the traditional client–server and mobile computing paradigms, and the focus has shifted toward ubiquitous computing. Most IoT implementations currently use centralized client–server architecture to connect to the cloud over the Internet, where the data is processed to provide insights. Centralized implementation is sufficient for current IoT ecosystems where hundreds or even thousands of devices are involved. However, centralized architecture is not a suitable design choice for billions of devices as it will create a bottleneck. Such implementation will require costly upgrades and maintenance of the cloud servers to handle a vast volume of data and manage the devices.

The future of IoT will be distributed and decentralized, and the management of billions of heterogeneous devices needs to be automated. Therefore, autonomic computing concepts are highly relevant in the IoT setting and must be adapted to realize autonomic management in IoT. To manage the entire operation of the network, the IoT ecosystem as a whole (from the device layer to the application layer and all components in between) must be sufficiently intelligent to handle, analyze, and adapt according to the dynamic environment and address various challenges in an IoT ecosystem. This will be enabled by autonomic IoT, allowing distributed decision making to maximize operational efficiency by being aware of the operating environment. Intelligence will come from autonomic implementations at each layer of the IoT ecosystem that will enable properties like self-security, self-organization, self-adaptation, self-optimization, and self-configuration. Autonomic computing in IoT will be aided by advances in edge computing, fog computing, Blockchain, and machine learning (ML) to enable real-time decentralized decision making to optimize various network operations. There has been increased interest by the research community in investigating autonomic computing applications in IoT. However, the progress is relatively slow due to fragmented research, development time, and the inherent complexity of the vast scale of heterogeneous IoT networks to implement self-managing applications.

B. Novelty and Contribution

This research is unique since it captures the subset of IoT relating to autonomic IoT. The work presented is not a survey article on the technologies, architectures, applications, and prototypes covering the complete scope of the IoT paradigm. Our work focuses on the unique challenges involved in minimizing human intervention on a massive scale of IoT deployment using autonomic computing principles. The current work extends the conference version of our prior work [5] and contributes to the current state of literature in the following manner.

1) Discussion on the benefits of applying autonomic principles in the IoT from component level to system level.
2) Analysis on self-* paradigm and monitor, analyze, plan, and execute (MAPE) loop for enabling autonomic behavior in IoT.
3) Role of key enabling technologies toward enabling autonomic computing.
4) Evaluation criteria for determining the level of autonomy in an IoT system.
5) Future research challenges, open issues, and research directions for applying autonomic concepts in IoT.

C. Comparison With Existing Literature

Few surveys have been conducted concerning traditional autonomic computing [6], [7]. The current study, in contrast, highlights the ideas for enabling autonomic behavior in the IoT by taking into account the size, scalability, and limited features of IoT nodes and comparing them with current autonomic techniques. Data mining techniques for IoT-related applications have been presented in [8]. A broad range of industrial applications using autonomic principles have also been proposed, which have been surveyed in [9]. IoT design aspects ranging from low-requirement software solutions to efficient hardware systems have been discussed in [2] and [10]. Regarding the autonomic protection of systems, autonomy[1] is also essential for security. There is a lack of thorough analysis of autonomic security, which is a vast area of study on its own. Ashraf and Habaebi of this article investigated autonomic security protocols suitable for the IoT in [11]. Consequently, the previous work on autonomic security for IoT has motivated us to describe the fundamental components of autonomy in IoT and the methodology for assessing autonomic schemes for IoT. Several surveys exist in the literature covering IoT technology, IoT security, protocols, prototypes, and applications. A broad survey on IoT architecture has been performed by Ray [12]

[1]The goals of both "autonomous systems" and "autonomic systems" are the same, i.e., minimize human intervention. However, the significant difference is that "autonomic systems" work according to the autonomic control loop, allowing an agent/system to respond and recover during unforeseen events. Autonomous systems can operate independently but do not necessarily have self-* properties of autonomic systems.

summarizing the current state-of-the-art IoT architectures in various domains. A survey on autonomic architectures has been conducted by Savaglio and Fortino [13]. The authors also addressed the cognitive computing paradigm in IoT. Another survey paper [14] highlighted the security of the various IoT frameworks. They also explored design flaws and provided in-depth security analysis against potential threats. A survey [15] on security challenges focused on standardization activities and discussed privacy, access control, trust, and identification. Several other surveys have focused on software-defined networking (SDN) [16], Context-aware computing, learning, and big data [17]. More recently, Fizza et al. [18] surveyed various metrics for measuring the quality of IoT applications to enable resilient quality-aware autonomic IoT applications. Based on the evaluation of existing studies, the work presented in this article on autonomic IoT is significantly different. We provide holistic coverage of the application of autonomic computing in IoT, its significance and various enabling technologies, which have not been covered comprehensively. We also provide an evaluation framework to identify various properties of autonomic computing ranging from the individual components to complete systems. Table I provides a summary of the existing literature in a comparative manner, which supports our claim of lack of survey as presented in this article.

D. Research Methodology

This article aims to survey the existing literature on autonomic computing in IoT to identify key characteristics to measure the autonomy of an IoT. The survey covers a range of topics that may enable autonomic computing in IoT and identify research gaps and future research direction, focusing on the literature published predominantly in the last decade. For this purpose, scientific papers were sourced from notable venues like IEEE Xplore, Scopus, WoS, Springer, Science Direct, MDPI, Wiley and the ACM Digital Library. Besides the journals, books, conferences and proceedings, symposiums, edited volumes, magazines, and preprints were also explored. The existing literature was identified by searching terms like IoT, autonomic computing, IoT Models, cloud computing, edge computing, fog computing, artificial intelligence (AI), ML and various combination of such terms, with IoT being a common term in all searches. Based on the analysis and survey of existing literature, this article aims to answer the following questions.

1) What are the key properties required to exhibit autonomic behaviors in each layer of the IoT ecosystem and their significance to IoT?
2) What are the key enabling technologies in an IoT ecosystem and their influence on enabling autonomic computing?
3) How can we evaluate the autonomic behavior of IoT systems to identify whether they exhibit autonomy?

E. Article Organization

Section II presents autonomic computing elements and their functional components and discusses self-* properties. Section III discusses relevant technologies that are being used to enable autonomic concepts in IoT. Section IV discusses the layerwise requirements for autonomy in IoT. Section V presents the evaluation criteria and discusses examples of evaluation usage. Section VI lists key challenges and issues that need to be addressed to enable IoT autonomy. In Section VII, overall observation regarding the autonomic behavior in IoT is summarized in light of existing literature. Finally, the conclusion is drawn in Section VIII.

II. Fundamentals of Autonomic Computing

This section presents characteristics, requirements, and essential features present in an autonomic system and the control loop operation for an autonomic framework. We present sufficient theoretical background and explanations to properly aid the discussion in subsequent sections. Discussion of each topic in Section II is followed by relevant literature for interested readers to explore further.

A. Definition of Autonomic Computing

IBM defined autonomic computing as an architectural framework for simplifying system management in 2001. A later definition of autonomic computing stated that it is "a vision that enables any computing system to deliver much more automation than the sum of its individually self-managed parts" [26]. Autonomic concepts are helpful and are applied in various technological areas. For example, NASA uses the autonomic computing principle to enable several operations in remote missions, where human intervention is impossible, to increase the chances of survival [27]. To establish autonomy, the system is divided into smaller components to establish modularity, and the central authority assigns the roles. As a result, the existence of a central authority is a necessary requirement. An autonomic system can also be described as an intelligent system, or system of systems, where information obtained through sensing or monitoring capability is used in a general autonomic decision-making process. An autonomic system should be able to configure itself in response to varying and unpredictable conditions. The autonomic system manages the computing resources to minimize user (or manual) intervention. The idea of autonomy is created to enable technology to be deployed and manage other technology while maximizing its functionality.

B. Autonomic Control Loop

The MAPE control loop (monitor, analyze, plan, and execute) is the motivation for the concept of autonomic computing. Two elements: 1) an autonomic manager and 2) an autonomic managed resource enable autonomy through the control loop [4]. The MAPE control loop is not a sequential process but a structural arrangement of its subprocesses. The architectural components suggested in the autonomic control loop must be adhered to implement autonomic computing concepts in IoT systems. The reference model for MAPE is presented in Fig. 1. Each element is discussed below based on the functionality it offers.

TABLE I
EXISTING SURVEY ON AUTONOMIC COMPUTING IN IOT

Survey Scope	Focus	Limitations
Autonomic threat management IoT [11]	Deals with various autonomic schemes to mitigate the threat in IoT in an autonomic manner	Focuses specifically on security aspects using autonomic concepts
IoT architecture [12]	Generic architectures for IoT, challenges and enabling technologies	No discussion on enabling autonomy and cognition IoT
Cognitive and Autonomic architecture [13]	Compare several architectures for enabling several characteristics of autonomy and cognition.	Very limited to architecture. No discussion on challenges and research directions. It does not cover enabling autonomy across all layers of IoT. E.g. device layer and the application layer.
Quality of Autonomic Internet of Things Applications [18]	Assessing the quality of Autonomic applications in the IoT ecosystem by defining various quality metrics	Focus on quality metrics with the assumption that the system is autonomous. Lacks discussion on autonomic features and characteristics itself, e.g. Self-* properties.
IoT enabling technologies, Protocols, challenges, Standards [19]	Listing various technologies (e.g. communication technology), protocols at various layers and standardisation work in IoT.	A generic review of ongoing efforts in IoT and enabling technologies.
Autonomic software system [20]	Overview on the application of Autonomic computing to intrusion detection, cloud-based data security, IoT	Lacks in-depth coverage focusing on IoT requirements and evaluation technique.
IoT middleware [21]	Various middleware in IoT and functionality required.	Focus only on one layer of IoT ecosystem
Deep reinforcement learning [22]	Various models for autonomic IoT using reinforcement learning.	Focus only on reinforcement learning
Autonomic cloud computing [23]	Management and security solution in the cloud	Focus on enabling autonomic computing in cloud environment
AI for next-generation computing [24]	Emerging trends and future directions for future computing	Focus on enabling AI in SDN, NFV etc.
Network Protocol Design for Autonomic Internet of Things [25]	Design of network communication protocol supporting autonomic Internet of Things	Focus on MAC and network layer protocols with discussion on evaluation or enabling technologies.

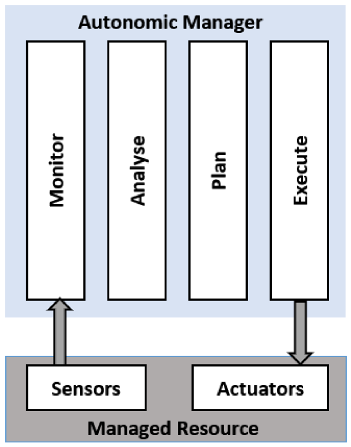

Fig. 1. MAPE control loop.

1) *Monitor Module:* The data collection is done using this module in the MAPE loop. The autonomic system can monitor changes in the environment and be aware of the state of its surroundings with the aid of this module. IoT has a wide range of parameters that can be tracked, including sensor and control data as well as the environment and device states. The monitor module is the primary interface between the system and the environment. Therefore, it generates the bulk of the data in the system.
2) *Analyze Module:* After the system has gathered sensor and control data, it is analyzed to generate useful information. Based on the data gathered, the analyze module enables the modeling of complex scenarios. This model enables the system's central authority to discover, recognize, and forecast environmental patterns.
3) *Plan Module:* The plan module manages the system with the aid of higher level rules and user-defined policies. The higher level policies impose restrictions placed on functionality and services. Planning is a crucial component of any system actuation that is required as it justifies the operation of remote devices and the required action.
4) *Execute Module:* This module manages the workflow execution in an autonomic system. It manages how the policies and rules are implemented and gives feedback to the monitor module. The best combination of services that can be used within the restrictions set by the plan module determines the workflow. The execution module handles the actuation process and related procedures.

C. Functional Components

An autonomic system needs two sets of actors or agents to play the parts of the managed resources component and the autonomic manager component. In the setup that Kephart and Chess [4] suggested, these two agents vary significantly in their capabilities and functionality. The managed resource components are more basic and consist of connected actuators and sensors. The managed resource provides an interface that the autonomic manager can use. In the original design, any piece of hardware or software with autonomic capabilities serves as the managed resource component. Some examples of managed resources are Web servers, printers, and clusters of machines in a grid. In the IoT context, the managed resources include mobile nodes, end nodes, smart objects, sensor nodes, devices, gateways, middleware, motes, smartphones, and electronic media devices. The resources in IoT are assigned requirements so that they portray the features of a managed resource. The devices classified as managed resources in IoT vary in several aspects, such as computational power, battery life, memory, mobility, and quantity.

The autonomic manager is considered to be the central authority. There may be other components that have higher privileges than the autonomic manager. The autonomic system can have a hierarchical arrangement of autonomic managers. For instance, in cluster-based networks, a cluster head or a reasonably powerful device, like a gateway or end router, can be chosen as the autonomic manager for a collection of similar nodes. In addition to running on hardware, autonomic managers can also be represented by cloud-based services or software modules [such as visualization using SDN and network function visualization (NFV)] that are eventually connected. Differentiating between the nature of these two components is one of the difficulties in implementing autonomic computing in IoT. In contrast to the original definition of autonomic computing, which assumed highly rigid and static systems, the IoT is very dynamic and open-ended, with diverse resources and workload capabilities [28]. Which components in IoT are chosen for the role of the managed resource is an important consideration.

In IoT, the autonomic concepts can be implemented in a layered manner at different layers independent of each other. Therefore, based on the system design, autonomy can be implemented in middleware, higher level applications, or at the device layer. A managed resource typically consists of a collection of sensors and actuators. The sensors are only constrained by memory, processor usage, connectivity, and response times as they generate data based on observations of their environment. The IoT ecosystem significantly relies on sensors, which offer feedback based on sensing the environment. A managed resource now refers to environmental and surrounding data rather than server or machine data. Any IoT device generating data is a candidate for the managed resource role. Table II describes the functional component comparing the role of various entities in the IoT context and the original scope of autonomic computing.

The scope of decision making in autonomic IoT is illustrated in Fig. 2. It highlights the interaction between autonomic managers and managed resources at the component and system levels. Fig. 2 also highlights the components and the position of autonomic agents (managed resource and manager). In Fig. 2, the left side shows the standard layers and components of an Internet-based system. The components are organized as fog and things. In fog computing, the computational resources that are usually located in the cloud are brought closer to the end devices [29]. The fog in Fig. 2 comprises the network, visualization capabilities, middleware, and Web applications. In contrast, things are made up of sensors and gateways. In an IoT system, like the one shown in Fig. 2, realizing the control loop of autonomy necessitates determining which parts will be handled as managed resources and managers. Given the flexibility of IoT, any IoT component can be given a manager or managed resource role as long as the hierarchy is maintained. As a result, any component at a higher level can take over as the autonomic manager of the lower component. IoT autonomy will aid in decision making for several tasks, including identity management, device management, and access management. Because it is necessary to manage gateways, end devices, numerous middleware, and remote servers, management issues arise at every IoT layer. The autonomic concepts can be used to manage and increase the effectiveness of each component, whether they are working individually or in a group.

For example, a discussion on the application of autonomic computing technology in the cyber–physical system (CPS), the autonomic computing cycle (ACC) (implementing MAPE control loop) and autonomic computing supervisor (ACS) serving as autonomic manager was proposed in [30]. In the proposed autonomic system for CPS, self-protection is handled by distributed CPS middleware, while ACC implements self-configuration, self-healing, and self-optimization. The complete autonomic cycle is implemented in every object of the CPS. Each CPS object individually, and the collection of CPS objects collectively can deliver seamless service without user involvement. ACS collects a large amount of data from numerous sensors and monitoring devices. Using the information gathered and data stored in the knowledge base, it determines whether the system is experiencing abnormal situations and creates and implements strategies to accomplish management objectives in accordance with the operation policy and service level required.

D. *Goals of Autonomic Computing in IoT*

In this section, various objectives of autonomic computing are outlined. These objectives focus on the common traits discussed in the autonomic computing literature.

1) *Goal 1:* Integration of state-of-the-art technologies, in particular, to manage and improve the operation of the overall system.
2) *Goal 2:* Intelligent decision making based on the contextual data obtained through sensing or monitoring at each layer for optimization and performance improvement.
3) *Goal 3:* Minimize the need for human intervention by automating the healing, optimization, configuration, and security processes.

Minimizing human intervention by using the data collected to make decisions about identity, access, and device management is the common objective and essential for autonomic IoT. Earlier, in Fig. 2, the scope of decision making in autonomic IoT was presented along with the discussion on managed resources and autonomic managers. In the following section,

TABLE II
AUTONOMIC COMPUTING FUNCTIONAL COMPONENT COMPARISON

Component	**Original Context**	**IoT Context**
Managed resources	a grid of computers, web servers, servers, storage units, databases, application servers, and services	Electronic media devices, smart objects, gateways, end nodes, sensor nodes, mobile nodes, middleware entities, smartphones, and motes
Autonomic manager	typically a part of the operating system	Edge device, Fog device, and end router or gateway

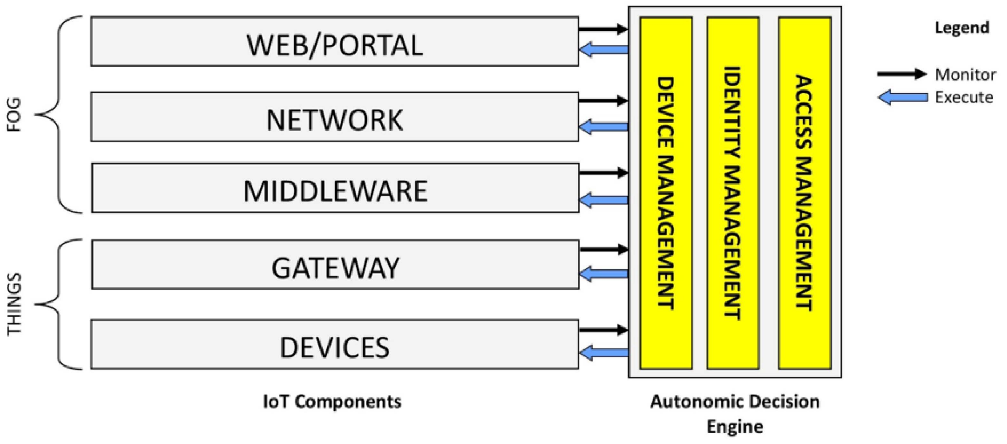

Fig. 2. Model for management of autonomic component using an autonomic scheme.

we discuss the self-* paradigm that has been extensively used in the literature to discuss autonomic computing principles and discusses the key properties of all self-* paradigms relevant to IoT.

*E. Self-*Paradigm*

Self-* (self-star) paradigm comprises concepts, such as self-configuring, self-adapting, self-organizing, self-security, and other similar processes. Depending on the characteristics and goals of the processes, the operation of any autonomic system can be described by a combination of the self-* paradigm. The self-* paradigm is critically analyzed in [31] with several examples for self-managing the elements based on different needs and scenarios. Modularity between various constituent parts of a larger system can enable self-* to be realized. The Internet of Underwater Things (IoUT) is one example [32] that requires high autonomy because maintenance is expensive in the underwater ecosystem, and it is challenging to provide continuous human administration. Such systems must implement self-* capabilities to cater to the deployment in such extreme conditions. Furthermore, requirements, such as mobility, make the system design more challenging, making management and maintenance even harder. The various components included in the self-* paradigm and related literature[2] are briefly described as follows.

1) *Self-Security:* The ability of devices to identify and react to security risks without human intervention is referred to as self-security. This calls for a variety of strategies, such as adaptive security policies, threat analysis and response, and intrusion detection and prevention. Self-security comprises self-protection and self-healing [11]. Self-protect is the ability to prevent, detect, and respond to any action that is deemed adversarial to the functioning of the system, which includes communication, hardware and software components, data, and intellectual property rights. In contrast, self-healing allows the system to detect failures and initiate policy-based action to recover without disrupting the operation of the system. For example, a device may alter its state or make changes in other components of the IoT system. The system must identify the cause of malfunction and take necessary corrective action without human intervention. In self-healing, fault detection is straightforward compared to implementing the mitigation approach required to fix the fault. The self-protecting part under self-security should enable the system to identify and protect itself from random attacks by identifying any hostile activity. Hostile activity may include, for example, Denial-of-Service (DoS) attacks, unauthorized access, replay attack, and the man-in-the-middle attack (MiTM). Healing a system after an attack or failure results in unavailability, which is directly related to the capability goal of the confidentiality, integrity, and availability (CIA) security triad. Therefore, self-healing can be classified under the security paradigm in an autonomic system. A wide variety of IoT devices are available in the market that varies in cost, performance, memory, and processing power. This variation in the capabilities of IoT devices presents a significant security challenge. Furthermore, IoT devices can become an attack surface

[2]Relevant recent works on self-* are presented in Table IV. Only selected literature is presented to give the reader an understanding of the concepts and their application in autonomic IoT.

to further infiltrate the system or launch attacks on other networks after they have been compromised, such as the Mirai Botnet attack. Therefore, self-protection and self-healing are very critical parts of autonomic IoT systems. To implement the self-security feature, Marchal et al. [33] proposed an efficient system for identifying the type of an IoT device by analyzing network communication. The model referred to as "AuDI" uses unsupervised learning to model the periodic communication traffic of IoT devices to perform identification. The model autonomously learns without human intervention or the need for labeled data to identify previously unseen device types after initial setup. Similarly, an autonomic system for detecting compromised IoT devices using federated learning for a distributed system is presented in [34]. The proposed system detected malicious behavior from devices infected by the Mirai malware autonomously. In [35], security management architecture using NFV/SDN is presented for security and privacy in IoT. The contextual and monitoring information from the IoT environments is used to act according to the actual status of IoT networks, systems, and deployed security policies to enable self-healing and self-protection. The architecture enables monitoring, detecting, and triggering autonomous responses to mitigate DDoS and IoT malware attacks.

2) *Self-Organization:* Self-organization is the ability to achieve the desired global state based on the interaction between various components of the system without the aid of a centralized entity, which in some cases is the human operator. Self-organizing systems examine their environment, collaborate to form topologies, monitor environmental changes and respond without any external intervention. A self-organizing IoT system should be able to stabilize itself in response to external changes solely through the coordination of its constituent parts, which operate with no knowledge of any system-wide properties and only a partial, estimated view of the overall system. "Reduced determinism" is the primary issue with autonomy and self-organization in general. Because self-organization is inherently random due to the dynamic and ad-hoc nature of IoT networks, self-organizing systems frequently exhibit high unpredictability and uncontrollability. As the system grows in complexity and scale, it becomes difficult to manage via centralized control as the system loses predictability and control. Predictability and scalability have an inverse relationship. The relationship between determinism and scalability over the different levels of self-organizational systems is shown in Fig. 3. Typically, distributed algorithms and protocols are used by self-organizing systems in autonomic computing for IoT to allow devices to find and connect to one another, exchange information, and coordinate actions. From a communication perspective, a load-balancing scheme for self-organizing IoT is proposed in [37] to avoid saturation and improve the reliability of the wireless link. The links are managed dynamically considering the spatiotemporal variations of data and control the signalling overhead to reduce the reconfiguration. However, the system uses a centralized orchestrator, creating a bottleneck and making it unsuitable for large-scale networks. The use of game theory for decision making and reaching stable outcomes in terms of topology control or self-organization has been an active research area in recent years. A static game model of selfish IoT nodes with multiple interfaces that decide on communication with others is presented in [38]. The proposed game-theoretic model results in realistic topologies and stable multihop network structures without a centralized entity. In [39], various game models that can be applied in massive distributed IoT networks for resource management are discussed.

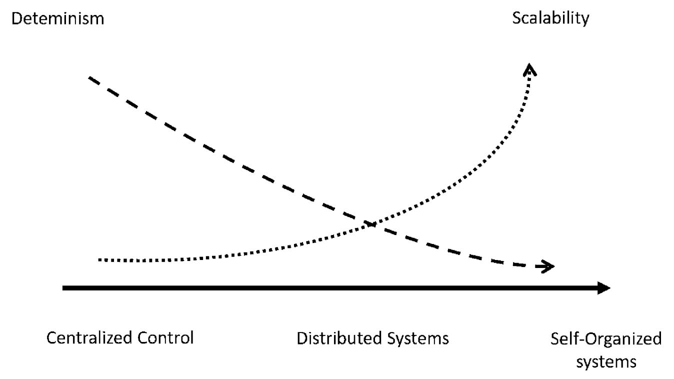

Fig. 3. Reduced determinism in self-organized systems [36].

3) *Self-Adaptation:* Self-adaptation allows monitoring of runtime and operating conditions of the components to assess if objectives are being achieved. A self-adaptive system continuously assesses the operating environment to ensure the goals are met and take corrective actions by adapting to any events such as system crashes. In contrast to self-organization, self-adaptation takes place locally, which is then reflected globally through self-organization. For instance, a self-organizing swarm of robots may also have self-adaptive control systems that allow them to modify their behavior in response to rapidly changing environmental factors. Components which self-adapt may necessitate frequent communication between agents and adapt accordingly. Adaptation necessitates constant monitoring of the environment and quick decisions to change the process to keep the operation running smoothly. Since most IoT devices are resource constrained, frequent communication will result in a short lifespan due to increased energy usage. Therefore, the adaptation can occur frequently or occasionally, depending on the requirement. Designing a system that can self-adapt depends on the available energy resources and is an important performance parameter for IoT energy efficiency. In autonomic IoT, the self-adaptation of IoT elements is important due to its highly distributed nature and operation of a large number of devices in unsupervised mode. This may involve the usage of a range of techniques, including dynamic resource allocation, load balancing, and adaptive control. A Quality-of-Service (QoS)-aware

TABLE III
EXAMPLE OF PARAMETERS THAT CAN BE SELF-CONFIGURED BY AN AUTONOMIC IoT SYSTEM

Category	**Parameters**
Identification	Device ID, Device Interfaces (Sensors/Actuators), Device OS
Device State	Device Status,duty-cycle, Device Memory, Device TX Power, Battery Remaining
Firmware/Software	Level 2 Stack info (e.g. RIME), Level 3 Stack Info (e.g. uIP, nanoStack, ISA100),
Communication	channel coding, max transmit power, Modulation scheme, Device Bit rate
Security	Public Keys, Authentication and encryption, integrity check

self-adaptive framework for critical IoT networks for e-health is presented in [40]. The proposed framework uses a discrete controller synthesis that relies on rules and labeled state transitions to monitor new contexts and monitoring requirements dynamically. A service-level agreement (SLA) is defined for each sensor and compared with the global SLA to set adaptation objectives. The gateway monitors the QoS of each sensor (gets the data from the device, adjusts their functional parameters, monitors their nonfunctional properties), and informs the discrete controller with events to decide on a self-adaption strategy. Arcaini et al. [41] proposed a concept for formal modeling, verifying, and validating a distributed self-adaptive system. The MAPE loop is adopted for self-adaptation in an abstract stateful language like abstract state machines (ASMs). In [42], an adaptive architecture based on the MAPE loop using fog and cloud computing for IoT-based health monitoring systems is presented. In [43], a self-adapting framework is proposed using a finite verification-machine-based model to perform runtime verification.

4) *Self-Optimization:* Typically, optimization refers to choosing the best option based on goals, decision-making guidelines, and constraints. Similarly, self-optimization refers to continuous monitoring and choosing optimal behaviors to tune resources automatically for optimizing system performance and resource utilization in real time. This could encompass both hardware and software components in order to maximize resource usage and satisfy design goals without the need for human intervention. Self-optimization may include allocating available resources to improve overall utilization, such as responding to dynamically changing workloads, allocating appropriate resources to meet specified QoS and discovering optimal routes from source to destination. Self-optimization is particularly important in the context of IoT due to the resource-constrained nature of IoT devices and the varying QoS requirements of IoT networks. A method to manage and optimize the QoS based on multiobjective optimization is proposed for IoT applications in [28]. The proposed method explores and learns the tradeoffs of different deployments to autonomously optimize the QoS and other quality attributes of IoT applications. In [44], energy optimization for energy harvesting in decentralized wireless sensor networks is presented using game theory and reinforcement learning. The game theory models the interaction among sensor network nodes and formulates utility for the energy balancing problem. Whereas reinforcement learning is used to address the time-varying availability of energy sources. By iterative learning, the sensors adapt their behaviors to the dynamic and unknown environment and thus optimize energy utilization. In [45], a method for self-optimizing topology using a backpropagation algorithm is presented to improve robustness against cyberattacks. The method uses a genetic algorithm to generate topology evolution data to train the back-propagation algorithm to achieve the desired level of accuracy for achieving optimum topology.

5) *Self-Configuration:* Self-configuration enables the system to automatically set up its components in an unknown environment. Self-configuring allows the system to respond to changes, such as the addition of new components (e.g., new nodes, hosts, routers, protocols and applications) or removing existing ones, as well as substantial changes in the operating characteristics of the system. It is highly inefficient to manually configure the system every time a change in the system behavior is needed. The delivery of new parameters or the distribution of services to devices may necessitate the need to change system behavior. For example, self-configuration can include schemes, such as device registration, adjusting device parameters, service, and device discovery [46]. Thus, self-configuration allows dynamic behavior without compromising the ease of management. Every part of the IoT system must exhibit a certain level of self-configuration. Some parameters and state variables used for self-configuration in IoT devices are listed in Table III. A fully autonomic gateway is presented in [47] to support IoT device registration and solve the heterogeneous network transmission problems. The proposed self-configurable autonomic gateway allows real-time detection and configuration over wireless networks. The gateways were tested using the AllJoyn framework for dynamic automatic updates of hardware changes, connection management of smart things, and discovery of home IoT devices. In [48], an adaptive self-configuring LORA network that dynamically adjusts the link parameters for scalable and efficient network operations is presented. The scheme increases the network delivery ratio while improving the reliability and energy efficiency of communications regardless of the network size. Focussing on CPS, Dai et al. [49] presented a case study for a baggage handling system capable of self-optimization and self-configuration by defining autonomic service management. The proposed

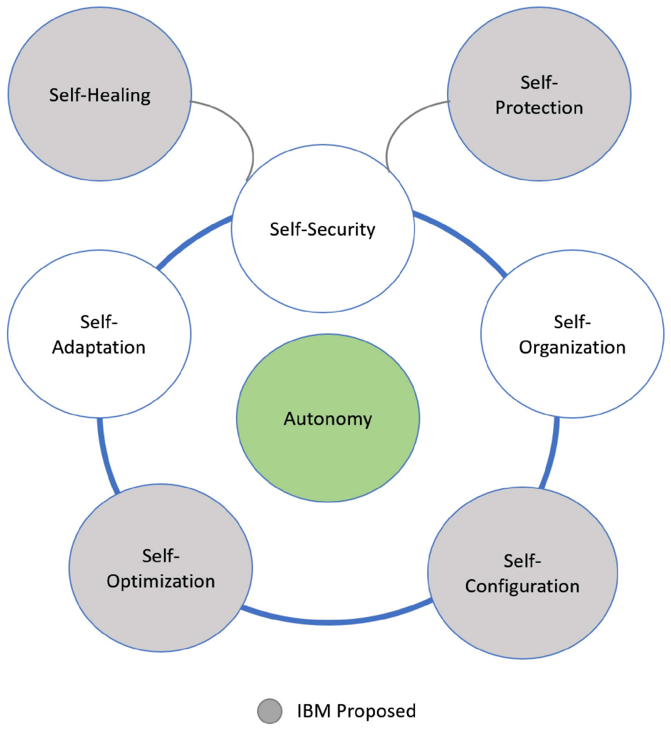

Fig. 4. Generic model for self-* IoT paradigm.

autonomic system allows for service orchestration using information and rules from an ontological knowledge base (OWL) base built using the MAPE loop. The rule-based autonomic service management provides device-level smart control using the knowledge base with rules defined using the semantic Web rule language (SWRL). However, the performance is reduced when using a distributed version where the size of the OWL files and the SQWRL engine time requirements can lead to inefficiency in the system.

In conclusion, autonomy IoT refers to implementing self-security, self-organization, self-adaptation, self-optimization, and self-configuration. Fig. 4 shows the core self-* behaviors for achieving autonomy. Only four of the self-* paradigms—1) self-healing; 2) self-protection; 3) self-optimization; and 4) self-configuration—were covered in the original discussion of the autonomic framework by IBM. Even though there are many self-* properties [50], [51], we identified the key self-* properties most relevant in the autonomic IoT context. We envision that autonomy in IoT can be characterized by five behaviors: 1) self-optimization; 2) self-configuration; 3) self-organization; 4) self-adaptation; and 5) self-security. The reason for additional self-* properties compared to the original IBM framework is to deal with the unique characteristics of IoT networks, which include high heterogeneity, complexity, and resource constraints. The choice of self-* properties varies and can be fine-tuned according to the application needs. Section V presents a framework to evaluate the self-* properties discussed in this section based on the various characteristics exhibited by each self-* property. Separation of self-adaptation and organization from self-configuration makes it easier to design a complex system in which each self-* property can be managed or developed independently (e.g., just like in the case of the OSI layer model). Furthermore, the self-* properties are not mutually exclusive and may have a dependency on each other. For instance, self-protecting actions may lead to self-healing actions. The self-healing processes may result in self-configuration, which could then result in self-optimization [52]. Self-adaptation covers all transient failures that can destabilize the overall operation and deal with the operation of individual devices. In contrast, self-organization is more focused on global interactions and cooperation between multiple elements (e.g., deals with joins and leaves that alter the topology of the network). Self-configuration is more closely related to the individual setup of parameter configuration of end devices or reacting to an external disruption to restore, maintain, or enhance functionality. A standalone IoT sensor might be able to self-configure. However, it might be unable to self-organize (due to the lack of connectivity). Self-configuration occurs in every self-organizing system, while the opposite is not true [51].

III. Enabling Technologies for Enabling Autonomy in IoT

Autonomic computing will transform IoT into an intelligent network that dynamically provisions and orchestrates the networking, computing, sensing, and communication resources required for a particular scenario. Autonomic computing in IoT necessitates a paradigm shift from traditional centralized architecture, where the network merely adapts its functions in response to specific environment states, to a distributed and decentralized architecture that can maintain its QoS under highly dynamic and complex environments. This section discusses the technologies that will enable the development of autonomic behavior in IoT systems. The development of the autonomic system will be made possible by combining these technologies. Moving control from a centralized entity to the edge to achieve greater autonomy is a critical consideration for these technologies. Since IoT uses distributed networks, such systems cannot be managed with human intervention at the runtime, making them challenging to manage and troubleshoot. These technologies will allow the IoT network to adapt its functions and maintain optimal resource usage and management when appropriately integrated.

A. Cloud Computing

IoT networks are complicated due to heterogeneity and the sheer volume of connected devices and services. Furthermore, due to limited computational capabilities, most IoT devices suffer runtime execution and processing delays. The developers are integrating IoT with the cloud to build intelligent IoT solutions that can function more effectively and efficiently by utilizing the scalability, flexibility, and affordability of cloud computing. The cloud assists the IoT networks by offloading the computational and storage requirements boosting execution speed and response time. With the help of cloud-based systems, administrators can regulate device behavior, update firmware, and carry out maintenance tasks from a single location. The cloud can also provide a centralized interface for managing and monitoring IoT devices. Additionally, utilizing autonomic computing principles with a centralized view of data, the cloud can act as a centralized autonomic manager. The aggregated view of data allows for appropriate corrective actions to be taken before any network issue arises to minimize downtime and optimize the operation of the entire network using analytics tools and ML algorithms.

To manage the unpredictable behavior of IoT devices in the case of potential DDoS attacks or device failure, autonomic computing and the cloud computing paradigm are proposed in [53]. In order to guarantee the availability of the affected service during anomaly, the proposed protocol transparently relocates the offered services of the vulnerable/failed devices to either another contextual agent or the cloud-platform-based IoT during runtime. This approach guarantees the availability of the services without interruptions and downtime and is suitable for real-time execution. The suggested technique integrates self-management abilities, including self-discovery, to create a contextual perspective and self-healing by selecting contextual agents which can carry out the necessary action as a contextual recovery plan.

Al-Shara et al. [54] proposed a generic solution for the autonomous runtime management of heterogeneous cloud systems to balance cost and revenue while meeting the constraints established by the SLAs. The optimal working configuration is chosen using a generic autonomic manager based on a constraint solver and evaluated on the OpenStack cloud platform. An autonomic resource provisioning approach based on the MAPE loop for cloud-based services to optimize cost and resource utilization is proposed in [55]. The hybrid resource provisioning service integrates reinforcement learning and autonomic computing principles. Real workload traces are used to evaluate the efficacy of the proposed method resulting in an overall cost reduction and increased resource utilization. A detailed review of autonomic approaches in the cloud is provided in [23].

B. Fog Computing

The traditional IoT designs have focused on connecting sensors and devices and sending data to some centralized or semi-centralized environment for decision making and processing. With fog computing [56], instead of taking a centralized decision, the processing and analytics power are brought closer to, or within, the devices themselves, thus reducing latency. Fog computing helps make an autonomous decision regarding where and when to deploy computational resources, storage, and control functions by creating awareness about objectives for a given context [57]. The awareness portion of fog computing changes the IoT devices from passive devices to smart active devices that can operate and respond to changes in the demands without waiting for instructions from a distant cloud. For example, sensors deployed in outdoor environments often require updates to resolve security-related issues. However, due to various factors, such as battery life, bandwidth, and signal strength, the centralized server may face several challenges in delivering the updates on time. This process increases the risk of a security attack. Whereas, in the case of fog computing, the backend can distribute the task of updating the nodes to selected fog nodes to deliver the updates to the nodes in the field swiftly.

Recently, researchers have experimented with autonomic computing and its application in fog computing. For example, autonomic controllers dynamically change the processing between fog and centralized servers to improve response time [58] has been proposed. In addition to network and computing, researchers have also attempted to use the autonomic management approach [59] to make buildings energy efficient by proactively adjusting duty cycles based on contextual data. An autonomic approach for a fully distributed service placement for fog computing was considered in [60] to avoid central coordination for virtual machine (VM) placement usually deployed in the cloud environment. This method improves system performance and optimizes network and system parameters comparable to a centralized solution. An autonomous resource provisioning system based on the MAPE loop and employing Bayesian learning is provided in [61] to control the workload fluctuations over time for IoT services. The suggested autonomic method manages the fog resources to take into account variations in workload for IoT services, preventing over/under provisioning issues while simultaneously satisfying QoS requirements. Compared to existing solutions, the autonomous resource provisioning framework reduces total cost and latency violation while enhancing fog node usage in a generic fog environment three-tier architecture. Ghobaei-Arani and Shahidinejad [62], and Jazayeri et al. [63] addressed a similar problem as in [61] for autonomous service placement framework in IoT network using metaheuristic technique and deep reinforcement learning, respectively. The proposed approaches result in efficient resource usage, increased service acceptance ratio, and reduced service delay and energy consumption.

C. Edge Computing and Edge Intelligence

Edge computing [64] is increasingly being used in the IoT for data processing, analytics, and decision making at the edge. It enables devices to analyze data locally and make decisions on the spot, reducing the amount of data that needs to be transmitted to the cloud. This not only reduces the latency in decision making but also saves on bandwidth costs and network resources. Edge computing allows full use of embedded computing capabilities to enable autonomy via distributed information processing [65]. Edge computing allows for the rapid configuration of IoT devices to accommodate the particular demands of users and changes in environmental conditions. Furthermore, the edge computing framework cooperates with the remote data center to enable intelligence in IoT devices (also referred to as edge intelligence) for any given application, e.g., smart factory and smart home.

Edge intelligence implements AI and ML technologies at the edge of a network closer to the source of data. It enables systems to manage themselves autonomously and make decisions without human intervention. Edge intelligence represents a major shift in how autonomic computing systems are designed and deployed, enabling real-time, data-driven decision making in a wide range of applications. Edge Intelligence is crucial to achieving the goal of a trillion connected devices by enabling IoT sensor nodes to function independently and sustainably in a setting where energy is limited and the supply is intermittent [66].

Bajaj et al. [67] presented offloading criteria where an IoT smart gateway acts as a mediator at the edge for autonomic decision making. They also highlight the approach taken to meet the performance requirements of IoT-enabled services. Edge computing is highly relevant in the modern energy utility industry [68] since the system can be made more efficient through distributed generation, storage, and consumption. An autonomic approach to Smart Grid demand management has also been previously studied in [69]. Data mining is crucial for the IoT, where billions of smart devices will produce rich data for creating new applications. However, traditional data mining approaches are inadequate for resource-constrained IoT applications. In order to overcome this issue, new data mining approaches designed for IoT and edge computing is addressed in [70], where centralized and distributed approximations of the K-Means clustering algorithm are implemented and evaluated on a real system. Using the EdgeCloudSimsimulator, the authors demonstrated that distributed clustering reduces computation, communication, and energy usage while retaining high levels of accuracy. Elgendy et al. [71] considered multiuser scenarios to address joint computation offloading and radio resources allocation to guarantee efficient utilization of shared resources for mobile users and IoT devices with limited computation power and energy. An offloading algorithm is developed to determine the optimal computation offloading decision for mobile users with an integrated security layer using the AES cryptographic technique to protect sensitive information from cyberattack. The proposed model can minimize the total overhead of time and energy, which can be further improved using a compression layer. Regarding resources and evaluation metrics, the requirements vary across cloud, fog, and edge computing. Researchers have presented a comprehensive review [72] on the differences and various optimization techniques relevant to edge computing. They also highlighted the need to utilize metrics to evaluate system performance based on individual applications and use cases.

D. Blockchain and Smart Contracts

The majority of IoT implementations use a centralized client–server architecture to connect to the cloud. Rapid IoT device growth will not work well for such an implementation. For example, the communication between IoT devices will have to go through the cloud, even in the vicinity. Such a model is prone to bottlenecks, downtime, and coordinated attacks that might affect the operation of the entire network. One solution is to use decentralized architecture for IoT to reduce infrastructure and maintenance costs and increase robustness by removing single points of failure. This introduces the need to include a decentralized mechanism that manages communication among devices implemented using technology like Blockchain. Blockchain is a shared, distributed database that many parties share. After receiving the approval from the participating nodes in the Blockchain network, the information is added in the form of blocks. The next evolution of Blockchain, which incorporates the concept of smart contracts, provides exciting IoT opportunities. For example, several important processes, such as updates, maintenance, and information sharing, can be automated using smart contracts according to the established and recorded rules in the Blockchain.

However, the benefits provided by Blockchain are not straightforward to be realized in IoT due to several limitations, such as scalability, processing power and time, battery, and storage. Therefore, the current Blockchain implementation must be adapted to the IoT ecosystem. Also, there is a need to develop an IoT architecture with Blockchain as an integral part. For example, The autonomous decentralized peer-to-peer telemetry (ADEPT) [73] presents a decentralized architecture to enable autonomic IoT systems. ADEPT integrates Telehash for peer-to-peer messaging, BitTorrent for distributed file sharing, and Ethereum Blockchain for autonomous device coordination. This architecture results in self-maintaining, self-servicing devices that can verify the trustworthiness of their peers and handle automatic and secure updates without needing a central broker.

IoT faces significant challenges in security due to the exponential growth in connected devices. Developing security mechanisms, such as authentication and authorization strongly relies on digital identity. Due to single point of failure and privacy concerns arising from current centralized identity management solutions based on external identity suppliers. IoT requires scalable device identity and authentication management to reduce the attack surface and offer protection against security threats like identity theft. To address this issue, an autonomic identity framework based on Blockchain is presented in [74]. The proposed framework uniquely identifies devices by autonomously extracting unique signatures based on their intrinsic digital qualities and relationship to their human owner (using a private key) that can operate in distributed and trustless situations. A Blockchain-based scalable identity management solution for IoT device authentication using a lightweight consensus-based identity authentication system is presented in [75].

IoT encompasses a large number of devices and services offered. The majority of IoT solutions only offer limited support for sharing IoT devices and costs incurred. In order to use IoT solutions, the users are required to set up and maintain the sensor that will gather the necessary data. In order to facilitate device and service discovery, an autonomic, global discovery and integration of IoT devices and services is proposed in [76]. The suggested framework enables pay-as-you-go cost sharing for IoT device costs while enabling IoT applications to find, integrate, and use IoT devices owned and managed by any

IoT provider. An IoT device metadata-focused Blockchain is created to store and manage all the data required for IoT device description, inquiry, integration, and payment.

E. Software Defined Radio and Network Function Virtualization

SDN is an architecture created to increase the flexibility and manageability of the network by decoupling the control plane from the data forwarding operation of the networking device. Whereas NFV enables service providers to increase service innovation and deployment agility by operating virtual network functions (VNFs) in VMs distributed over data-center infrastructures and swapping out specialized hardware appliances with network functions. SDN and NFV provide greater flexibility in the orchestration of networks and services due to dynamic instantiation and configuration of network functions and services in virtualized (fog/cloud) computing environments suitable for IoT applications. In order to deliver the right Quality of Experience (QoE) to end users, a relatively large number of unique stand-alone software modules and networking devices must be controlled for a diverse range of IoT applications. As a result, managing the IoT ecosystems operated by SDN and NFV will make it possible to implement autonomic management techniques. Since the human operator cannot manually manage a wide range of interconnected systems, this enormous and dynamic number of heterogeneous components creates a management issue for SDN and NFV that needs to be handled autonomously. Furthermore, securing IoT solely on human intervention makes it impossible to defend against an attack on sophisticated network structures quickly. Distributed auto-defence systems in autonomic communications (e.g., NFV security chains) constantly monitor system characteristics for attacks and identify the best course of action without significantly affecting overall performance, whereas the preset security configurations manage authentication, authorization, and access control.

An online heuristic algorithm called topology-aware placement of VNFs (TAP-VNFs) as a low-complexity solution for dynamic infrastructures to successfully determine the placement of VNF is proposed in [77]. The proposed work assesses the applicability of various VNF placement strategies for static (offline) and dynamic (online) scenarios. It offers a general formulation of network function placement based on the service function chaining concept. In terms of consolidation and aggregation ratios, the suggested approach outperforms the best available options, demonstrating a better fit for dynamic cloud-based environments. The outcomes demonstrate that TAP-VNF performs better than current methods based on conventional bin-packing schemes. Existing network and middleware infrastructure often fails to adapt dynamically and meet the requirements of evolving IoT applications, degrading QoS while being used. To address this issue, a self-adaptive communication infrastructure prototype at the middleware level of IoT systems to handle and manage floating QoS in dynamic and demanding contexts using edge computing, SDN, and NFV is presented in [78].

In order to enhance network management in 6LoWPAN IoT networks, a solution based on VNFs that can be deployed at the edge of IoT networks due to lightweight virtualization was presented in [79]. NFV allows VNFs to be scaled up or down at the network edge depending on the state of the network and system. In [35], a security management architecture was proposed, outlining the different planes of the architecture and the primary architectural flows to address security and privacy in NFV/SDN-enabled IoT scenarios. Along with the main IoT threads and attacks, potential NFV-SDN-based mechanisms for detection and defence were also presented. The suggested solution can automatically monitor, detect, and mitigate IoT attacks by enforcing appropriate security policies on time, considering the latency and delays experienced by IoT networks.

F. Artificial Intelligence

For widespread IoT adoption, the devices must be able to make decisions autonomically, requiring adding some intelligence across the entire IoT network. IoT devices can achieve this by deriving context through the implementation of AI techniques. Because autonomic computing is a component of AI, autonomic computing techniques have become integral to any AI-based development. AI can be used in autonomic systems planning and analysis phases, frequently set up as MAPE cycles. Once this is accomplished, human involvement will be restricted to setting up the system and creating policies and rules. The IoT system with AI capabilities will be able to learn and adapt to changes effectively and efficiently using the data generated by the devices. As a result, IoT networks can detect tampering, self-heal, identify impending failures, reduce downtime, and optimize process efficiency. For instance, ML techniques can be used to identify patterns in the workload and then use those patterns to improve resource management. Additionally, the autonomic manager could use ML models for predictions to achieve self-learning and reduce model uncertainty. Furthermore, AI brings a human element to IoT by applying data and context awareness to problem-solving [80]. For example, automatic software updates will be broadcast to all interconnected IoT devices before problems occur. For this to happen, context awareness is also critical, allowing machines and devices to infer meaning from data streams from disparate systems. Recently, autonomic computing has been increasingly applied in computer and networking paradigms like cloud, fog, edge and communication networks with the help of AI and ML. The application of autonomic computing approaches is significant when a system has a large number of configurable parameters that can be adjusted to keep the system running optimally. Cooperation and interaction between various components result in dynamic, complex, and heterogeneous networks. AI and ML techniques will be useful in managing such large-scale heterogeneous IoT networks by analyzing data and making decisions for the self-management IoT network with the desired level of QoS.

For the autonomic management of a large number of offloading requests in mobile edge/fog for IoT applications, a deep Q-learning-based compute offloading approach was proposed in [81]. The proposed scheme considers the resource demand, availability and network status mobility, and heterogeneity to make the optimal decision. The problem is modeled as a Markov decision process (MDP) and solved using reinforcement learning due to the randomness in resource availability and the wide range of possibilities for assigning those resources for offloading computation. Based on the experimental results, the proposed offloading mechanism performs better in terms of execution time, latency, and energy usage. With the recent advances and optimizations in deep neural networks (DNNs), applications like computer vision can be run on simple devices without requiring specialized hardware. However, limited device memory, computing power, and battery life make DNN implementation and satisfying QoS requirements difficult. Therefore, offloading of tasks is required in such scenarios. Therefore, a novel optimization method for parallel offloading large-scale DNN models in a local-edge-cloud collaborative environment with limited resources is presented in [82] to minimize the offloading failures due to intermittent wireless connectivity during DNN data transmission. A thorough discussion on using AI/ML with the autonomic principle is presented in [24] for next-generation computing.

G. *Ubiquitous and Reliable Communication*

The necessity of information exchange is vital to achieving autonomic behavior in the IoT because it enables the components of an autonomic system to communicate critical information and take the appropriate action based on contextual data. To realize this, ubiquitous and reliable communication technology is essential for all autonomic systems to integrate and cooperate effectively. In this regard, new features are being added to cellular networks to enable ubiquitous and pervasive communication supporting millions of IoT. Traditionally, the main focus of such networks was to provide higher data rates without considering the lower data rate requirements of IoT devices. To address this issue, the current and future generations of cellular networks, such as 5G and beyond 5G, are being designed to accommodate massive machine-type communication (mMTC). Furthermore, several other networks, classified as low-power wide-area networks (LPWANs), are being deployed to support the connectivity demands of the exponentially growing IoT devices.

Furthermore, communication networks should increase their ability to deal with unexpected changes, such as changes in topology, load, task, and the physical and logical properties of the networks that can be accessed. This gives rise to the concept of Autonomic Communications, which applies autonomic computing principles to communication systems and networks to provide end-to-end QoS using self-management [83]. In this regard, cognitive radio [84] is a term used to describe intelligent radios that can make decisions autonomously based on information gathered about the radio environment and learn/plan based on experience. Cognitive radios can communicate efficiently, avoid interference with (un)licensed radio spectrum, and modify their radio broadcast or reception characteristics. The cognition cycle is similar to the MAPE loop and is based on Observe-Orient-Plan-Decide-Act-Learn, which means that the cognitive radio system constantly examines its surroundings, makes plans, decides, and then acts. Autonomic communications are crucial in scenarios like smart city and smart transportation applications, where there are varying radio conditions and high dynamic numbers of interconnected sensors and actuators (due to node failures and duty cycling). For instance, smart objects can organize in ad-hoc networks to exchange information and coordinate tasks even when the topology is dynamic. Any intelligent communication device should support autonomic communication and self-management capabilities, which can optimize any network decision and task, such as packet scheduling.

Riker et al. [85] discussed the challenges and solution for self-adaptive communication in Dense IoT (DIoT) environments. The proposed solution is implemented as a software layer which uses a fuzzy logic controller to autonomously detect and react to low-performance situations to provide an energy-efficient and reliable group-oriented communication mechanism called Autonomic management of Group communication for IoT applications (AGREEN). The proposed solution results in an improvement in terms of the notifications interval, message loss, and energy consumption when compared to the standard version of the CoAP protocol. This is achieved by adapting the communication settings of a group of nodes based on group indicators, such as the amount and criticality of collected data, traffic loss rate, type of energy source, and rate of energy harvesting.

IV. Autonomy at Various IoT Layers

IoT is typically divided into three functional layers: 1) the device layer (also referred to as the M2M layer); 2) the network layer; and 3) the cloud layer [8]. Each layer offers a different set of features and services, as well as different limitations and challenges for IoT autonomy. The interfacing requirements, mobility requirements, sensing, and processing power in the components of each layer are used to define the capabilities of every layer. Devices with sensing and actuation capabilities are included in the device layer. Databases, analytics, and human–machine interfaces are included in the cloud layer, while middleware and data transport resources are included in the network layer. The term "interfacing" describes how easily different devices can communicate within the same layer. Mobility, on the other hand, is the dynamic nature of the devices, the flexibility to move, and the ability to switch network attachments. Due to heterogeneous devices, technologies and protocols, the lower layer interfaces are subject to more restrictions. On the other hand, the widespread use of TCP/IP and other similar protocols standardizes interfacing and makes its implementation at higher layers easier. Additionally, in the higher layers, devices are less mobile. Based on this finding, autonomic software can exist: 1) in the device layer as agents residing in the end devices; 2) in the network layer as a part of middleware; and 3) in the cloud layer with application protocols as cloud agents. Based on the autonomic IoT requirement discussed in terms of self-* paradigm and MAPE loop, supporting technology,

TABLE IV
SUMMARY OF RECENT WORK RELATED TO AUTONOMIC COMPUTING AND THEIR APPLICATION IN IoT

	IoT Layer			Enabling Technologies					Self-* Paradigm					Decision Making		
Reference	Device Layer	Network Layer	Application Layer	Cloud Computing	Artificial Intelligence	Blockchain	SDN/NFV	Edge/Fog	Self-Configuration	Self-Adaptation	Self-Security	Self-Organisation	Self-Optimisation	Distributed	Centralised	Cooperative
[47]	●							●	●			●	●		●	
[86]	●							●		●			●	●		●
[82]	●			●	●			●				●	●	●	●	●
[87]	●				●					●			●		●	
[88]		●		●	●		●	●		●			●		●	
[89]		●			●			●		●			●	●		●
[90]		●			●	●		●	●				●	●		
[91]	●	●			●	●		●			●			●		●
[92]	●	●			●	●					●			●		●
[93]		●					●				●				●	
[81]	●	●			●			●		●			●	●		●
[35]	●	●					●	●			●			●		
[54]			●	●					●	●					●	
[55]			●	●	●				●	●					●	
[94]		●	●	●	●		●				●			●		●

autonomic capabilities can be enabled at various layers of the IoT network.

Table IV is an illustrative example summarizing the discussion in the previous sections considering recent literature. The table groups the work based on the layer of the IoT network where autonomic computing capabilities were implemented, enabling technology used to implement self-* properties and the decision control implemented to achieve defined goals. From Table IV, it can be seen that there have been efforts to enable autonomic capabilities in the IoT ecosystem, focusing on various layers and using different technologies. However, several issues/challenges need to be addressed to achieve end-to-end autonomic capabilities due to the inherent complexity of the IoT ecosystem. To achieve full autonomy, IoT systems require cooperation across various layers, using multiple enabling technologies and cooperative decision-making capability (combination of both centralized and decentralized) to implement a complete set of Self-* paradigms.

V. Evaluation Framework

This section presents a set of criteria that can be used to evaluate the state of autonomic design employed within a system or a subsystem. To achieve self-* properties, the proposed criteria can serve as a guideline to classify the decision making of any system/scheme/approach in IoT as autonomic. The proposed criteria described in the subsequent section were chosen from the literature that defines the properties of autonomic systems [4], [26], [31], [36], [95], [96]. For example, the two criteria of "status monitoring" and "observation" may appear to be the same, and both do fulfil the "Monitor" function of the autonomic control loop. However, the objective and context of the monitored condition are different. Status monitoring is the process of monitoring the current state of its (internal) resources to ensure efficient operation. In contrast, observation is the process by which the system obtains input from the (external) environment to make a collective decision at the autonomic manager. For instance, an IoT device may constantly check its battery level but not know how long the network will last (which may be observed only by a gateway like the autonomic manager).

These evaluation criteria are qualitative parameters to determine the autonomic capability of the system. However, depending on the context of the system being evaluated, each evaluation criterion may have a unique subset of performance metrics. For the proposed integration criteria, for instance, a system must dynamically adapt to and comprehend the needs of other similar systems. Therefore, the autonomic systems may have to renegotiate configuration parameters to work together as a single entity to accomplish the specified goal. In a networking context, suitable QoS parameters (e.g., delay, throughput, packet loss, jitter, etc.) are examples of specific parameters for the integration criterion. For example, a system may tolerate higher delay but not low throughput, whereas another system may tolerate low QoS metrics for both. QoS are application specific and need to be negotiated during the runtime as an adaptation process that may require integration (or negotiation) with various components and assessed using accepted methods.

Furthermore, the presented criteria are for the overall evaluation of the IoT system, supporting the definition presented in [26], which states that an autonomic computing system should provide significantly more automation than the sum of its individually self-managed parts. An overall evaluation of IoT autonomy is more important than individual component evaluation. As a result, the evaluation criteria will be consistent across all layers.

A. Criteria of Evaluation

1) *Negotiation:* One of the important tasks in autonomic computing is to use the data gathered by continuous monitoring for making decisions. The gathered data can

be treated as a requirement and is used to negotiate and communicate requests to the devices. Negotiation is a process where two parties mutually agree to perform certain tasks within their capabilities. Negotiation may be performed directly by the devices or by an intelligent entity at a higher level in the system. Negotiation minimizes user intervention by making optimization decisions instead of waiting for input from the human. An IoT system capable of negotiation can dynamically change its state based on the gathered data without human intervention and can claim partial autonomy.

2) *Observation:* Autonomic elements must monitor the environment and determine the likelihood of various services because managed elements may not be capable of performing such observations. However, the managed elements can provide input for a collective decision by the autonomic manager. The observation parameter refers to the process by which the system gathers data from the environment to make decisions at the autonomic manager. The shared objective of autonomic IoT, which is to generate data and collect it from the system and the environment, is immediately met by this procedure. Observation parameters directly fulfil the goal of an autonomic IoT system by providing the capability to monitor. Furthermore, in certain scenarios, the environment may not be fully observable, and IoT devices must collaborate to share the observations to take optimal actions.
3) *Deliberation:* Deliberation ensures that all the observable events and actions are considered in the decision-making process. Interfaces do not interact with the environment directly. Instead, they help respond to failure by using predefined actions or compromising certain functionalities. Hence, deliberation allows for self-optimization and system-level adaptability. The success of a decision made depends on the quality and quantity of inputs. Fewer factors considered in decision making usually result in a suboptimal solution. Therefore, the list of factors considered in decision making should be exhaustive for autonomic decision making.
4) *Failure Recovery:* Every autonomic system needs to have a recovery mechanism that helps keep the system functional in case of anomaly or misuse detection. Failure recovery refers to the autonomic system's action to ensure system availability. It is comparatively easier to detect failure for failure recovery than to perform the exact recovery approach to restore the functionality. The scope of this evaluation criteria is more toward decision making in case of unusual events. Furthermore, failure recovery results in reduced user intervention contributing toward autonomy. An IoT system implementing a failure recovery mechanism also exhibits partial autonomy in security features.
5) *Execution:* The managed resources take certain actions in a given situation after mutual agreement between various entities involved. The autonomic manager supervises the action, which ensures that all the negotiated tasks and parameters are honored. One of the goals of autonomic computing is to use technology (or technologies) that can manage other technologies. To achieve this goal, the execution of the control process forms a core element of the autonomic MAPE loop.
6) *Prioritization:* This evaluation criterion uses policies in decision making. The prioritization may occur either in determining the data flow or how routing takes place in a particular scenario. For example, in [97], a method is proposed to maximize the lifespan of the node to make the network more efficient. This is achieved by temporarily disabling the nodes to save power using the interpolation algorithm. Prioritization can also allow the device management to take precedence over other actions if the resources are limited and availability is more critical than privacy. In such cases, access and identity management decisions can be overridden.
7) *Status Monitoring:* Status monitoring is an important feature in every autonomic system contributing to self-awareness. The autonomic system continuously monitors its elements and controls the resources to guarantee efficient operation. In the IoT context, this function can be regarded as a subset of self-configuration. The status monitoring and observation criteria for evaluating the autonomic scheme may look similar as both are confined to the scope of monitoring. However, "observation" refers to collecting data from the environment, whereas "status monitoring" is limited to the current state of internal resources to improve efficiency. Both criteria handle monitoring and gathering data for autonomic analysis.
8) *Arbitration:* Arbitration is the process of selecting a particular service, configuration, or function from the available set of alternatives. This directly relates to the goal of autonomic decision making in IoT. The main challenge in implementing arbitration capability in IoT systems is the heterogeneity of IoT devices. An IoT system with an arbitration feature can be self-sufficient in management, and human intervention can be minimized in selecting functions, services, or features.
9) *Integration:* Integration is important due to the heterogeneity in the protocols and devices. Integration is defined as a process through which the autonomic system can adapt itself to the requirement of another system without affecting its current operation. Integration achieves the high-level goal of the self-adaptation of functionality, optimization, and management of other technology. This function may be implemented in the autonomic manager as the managed resources may be unable to provide such functionality due to resource constraints. This functionality is usually implemented at the gateway or IoT middleware to allow seamless operation among heterogeneous devices and protocols.
10) *Filtration:* Filtration refers to using specific knowledge and understanding based on the available data for a given context. Filtration involves feature extraction, clustering, and classification that utilizes user-defined policies

TABLE V
PROPOSED EVALUATION CRITERIA AND ITS RELATION TO MAPE CONTROL LOOP

	Negotiation	**Observation**	**Deliberation**	**Failure Recovery**	**Execution**	**Prioritisation**	**Status Monitoring**	**Arbitration**	**Integration**	**Filtration**	**Reasoning and Learning**
Monitor	●	●	○	◐	○	○	●	○	○	●	○
Analyse	●	◐	●	●	○	●	○	●	◐	◐	●
Plan	●	○	●	○	○	◐	○	●	○	●	●
Execute	○	○	○	●	●	●	○	◐	●	○	○

●required, ◐optional, ○not needed

TABLE VI
PROPOSED EVALUATION CRITERIA AND ITS RELATION TO SELF-* PARADIGM

	Negotiation	**Observation**	**Deliberation**	**Failure Recovery**	**Execution**	**Prioritisation**	**Status Monitoring**	**Arbitration**	**Integration**	**Filtration**	**Reasoning and Learning**
Self-Security	◐	●	●	●	●	◐	●	◐	○	○	●
Self-Organisation	◐	●	●	◐	●	●	●	◐	●	●	●
Self-Adaptation	◐	●	◐	●	●	◐	●	◐	●	●	●
Self-Optimisation	◐	●	●	○	●	●	●	●	●	●	●
Self-Configuration	●	●	●	○	●	●	◐	●	◐	◐	●

●required, ◐optional, ○not needed

and filters unnecessary components to make unbiased decisions that may affect the operation of the network.

11) *Reasoning and Learning:* Context-based policies and decisions are a significant part of autonomic IoT. Based on available information, reasoning and learning can help an autonomic system to become self-sufficient to deal with heterogeneity and make decisions. ML techniques have reduced the need for manual searching and policy setting. An autonomic system using the learned model can find the best action for a given scenario. The advantages include application independence, strong validation, and improved decision making. An IoT system capable of reasoning and learning will guide the functions of the rest of the system and enable autonomy.

B. Evaluation of the MAPE Control Loop

The proposed evaluation criteria for an autonomic system must be compatible with the MAPE control discussed in Section II. Table V compares the four processes in the MAPE control loop with the proposed evaluation criteria. The monitor process in the MAPE loop represents the status observation, negotiation and filtration. Therefore, the monitor process is present at all layers of autonomic IoT. Likewise, the planning process represents arbitration, deliberation, negotiation, and filtration. The analyze process is more complex and represents arbitration, deliberation, negotiation, failure detection, reasoning, and prioritization. Finally, the execution process combines the criteria of prioritization, failure recovery, execution, and integration.

C. Evaluation of the Self-* Paradigm

The proposed scheme can also be used to evaluate the self-* paradigm to determine which autonomic components are manifested in individual self-* properties. Table VI assesses self-adaptation, self-configuration, self-organization, self-optimization, and self-security discussed in Section II to define autonomy with respect to the proposed evaluation criteria. As the definitions of self-* properties are relative, it is difficult to compare them with the proposed evaluation criteria. For example, self-security can be divided into self-healing and self-protection, which are evaluated differently. Thus, the evaluation criteria yield the best results when applied to a focused domain with precise requirements and features.

D. Application of Evaluation Criteria

In this section, we present the application of evaluation criteria using two relevant works, one focusing on IoT and the other being a generic network. We provide an example of self-organization to show readers how to apply the suggested standards for evaluating autonomy. A self-organizing network with optimization capabilities was proposed by Edwards et al. [98], in which specialized routing information is provided to the end devices via a gateway. The nodes modify their operation and other factors, like transmission power, frequency, and bandwidth, after receiving the information. Akgül and Canberk [96] considered a conflicting parameter to determine where the coverage zones of wireless devices overlap. This approach aids in expanding the coverage area, which improves event observation and even extends the lifespan of the network. Both studies under consideration list self-organization as one of their main objectives. Because the amount of detail is lower than a whole middleware or cloud-based analytic solution, these examples were chosen. We first look for a shared scope of work, such as interface capability, network structure, mobility, or energy efficiency. The focus in [98] research was mainly on networking technology and changing the configuration of device states. The research is focused explicitly on the self-organizing routing behavior of the network. The research in [96] utilizes a strategy to maximize network longevity, which lowers wireless coverage redundancy. The configuration parameter that is shared by both works is transmission power, which also has an impact on the wireless coverage area.

The features of deliberation, prioritization, status monitoring, integration filtration, and reasoning can be used as a starting point, as our focus is on evaluating self-organization-based works. As prerequisites to self-organization, Akgül and Canberk [96] discussed self-configuration, self-healing, and self-optimization. Both filtration and the negotiating criterion are not obvious. Because the working scheme does not negotiate any requirements with the environment, it does not filter any data for further decision making. However, the requirements of observation and status monitoring are met since the location and status information of devices is used as input to the self-configuration algorithm. The approach optimizes the network after specific occurrences. Therefore, the failure recovery criteria are also satisfied. One such occurrence is when the battery of the device is exhausted, and it disconnects from the network. The scheme excludes a semantic or context-based policy system due to its scope of work. There is no decision-making procedure in the work that considers all conceivable observable events and variables that could impact the radio model and the device coverage. In choosing sleep states, one can observe an arbitration requirement (sleep, passive, active). However, it cannot be inferred that the work can arbitrate disputes. To some extent, few options are reached after decision making. But the options are very limited, and the system is judged to lack the property of arbitration. In addition, the work lacks the property of integrating heterogeneous parts. This work assumes that all devices cover an equal area, which is not necessarily true. So, the performance of the algorithm will be severely impacted by every device with a varied coverage area. Because the configuration of selected preferred elements is altered, this task reviews execution and prioritization.

Similar to [96], the authors in [98] also covered the self-configuration, self-healing, and self-organization properties. This research primarily aims to reduce the number of hops and delays in retransmitted packets across an ad-hoc network. Also, the system keeps track of the number of hops, which means it keeps tabs on the condition of its components. The decision engine in this work, unlike earlier ad-hoc networks, is not wholly built within ad-hoc components. The application of the autonomic management and resource architecture, where decision-making functionality is distributed across the components, is suggested by delegating some of the functionality to centrally placed elements. We can consider this work to satisfy the execution and arbitration requirements of the configuration since the proposed approach can change transmitting power, frequency, and bandwidth operations. The ability of users and constituent elements to collaborate over an ad-hoc network, solving various infrastructure-based challenges, can be viewed as satisfying the criteria of negotiation and integration. The work also fails to meet the filtering requirements since no data has been filtered for subsequent decision making. Similar to [96], the evaluation criteria for filtering, arbitrating, and semantic reasoning are not fulfilled. The observation criteria is not fulfilled since the system does not gather information from the outside world to inform a group decision. Only internal status data is gathered, meeting the need for status monitoring. Since all observable (specified) occurrences are considered, the deliberation condition is satisfied. The requirement for failure recovery is also satisfied because reconfiguring the ad-hoc network makes it simple to fix a problem with one node.

The discussion presented in this section is summarized in Table VII, highlighting the characteristics of the proposed evaluation criteria implemented in the compared literature. In a similar manner, we can evaluate existing autonomic schemes/protocols against the proposed evaluation criteria to determine which self-* property is implemented at any given layer.

TABLE VII
APPLICATION OF THE PROPOSED EVALUATION CRITERIA

	Negotiation	**Observation**	**Deliberation**	**Failure Recovery**	**Execution**	**Prioritisation**	**Status Monitoring**	**Arbitration**	**Integration**	**Filtration**	**Reasoning and Learning**
[96]	x	✓	x	✓	✓	✓	✓	x	x	x	x
[98]	✓	x	✓	✓	✓	✓	✓	x	✓	x	x

✓implemented, x not implemented

VI. CHALLENGES AND FUTURE DIRECTIONS

Autonomic principles have been successfully applied to conventional computing and communication scenarios. Applying the autonomic principles in IoT will have a significant advantage in device, security, and network management. Direct migration and implementation are not possible due to the different size and functionality requirements of IoT and its constituent elements. Fortunately, the growing need and popularity of ubiquitous computing and the increasing IoT adoption assure a promising future for autonomic IoT. However, there are several issues and challenges that need to be addressed to apply autonomic computing principles from traditional computing to the IoT paradigm. This section presents the challenges that need to be addressed and suggests possible research directions.

A. Implementing Autonomic Complexity

Implementing autonomic principles on resource-constrained devices is a research challenge, especially trying to avoid complex protocols and maintain a simple modular design. The proper design combines many techniques and schemes with other common research ideas in IoT, such as energy conservation, interoperability, and AI. At the same time, the amalgamation and description of security requirements for cross-layer designs could be an exciting field of upcoming research. Self-adaption and autonomic design will bring an extra layer of complexity to the system implementations. Currently, designs are made simpler, lighter, and portable to meet the IoT device constraints. Such techniques must be more robust considering the computing and energy resources level. As an example, certificateless public-key algorithms are being applied to IoT. However, initial attempts at the design of certificateless mechanisms may not be suitable for IoT [99]. Autonomic-enabled IoT comprises resource-constrained, simple end devices. However, autonomic computing features and processes can result in overall complex behavior. Complex behavior and autonomic negotiation can also exist in sensors gathering data. The system decides on the characteristics of the environment based on the gathered data. Furthermore, each device or node in an intelligent IoT system should be network-aware by combining M2M communication with the transparent flow of data and avoiding complex protocols as much as possible. This simplicity is essential to maintaining and managing intelligent IoT.

Furthermore, issues of interoperability, common standards of management, and automation rise significantly in autonomic IoT, given the wide variety of hardware and software technologies. When choosing autonomic decision makers, the existence of devices from different manufacturers following the same standard might cause interoperability issues due to the presence of additional proprietary features. The traditional autonomic architecture follows a centralized pattern approach, with two major components: 1) the managed resource and 2) the autonomic manager. The autonomic features in the manager should be left untouched to protect the generality because the device constraints do not exist at the platform or server level. However, the managed resource requires several updated observations, negotiations, failure recovery, and deliberation. This is mainly due to the constrained nature of the end devices, and implementing full-fledged autonomic agents is not optimal. Therefore, choosing the position of the decision engine in an autonomic setup is challenging as it depends on several criteria and use cases. For example, we can position the decision making in the higher layers in a separate geographical location from the network itself. It is also possible to keep the decision making entirely in the device layer and fully assign it to the end devices. A third option can be locating the decision-making midway in the network layer as middleware (edge or fog). Another alternative is to distribute the decision engine across the layers. Each alternative has its set of advantages and disadvantages.

B. Failure Detection and Recovery

The physical deployment of end devices in wireless sensor networks and IoT varies from standard indoor and outdoor environments to rough military deployments. Devices may be exposed to harsh environmental conditions compared to traditional standard computing equipment. Therefore, failure detection becomes challenging in complex autonomic IoT systems with many possible states and actions. For example, sensors in smart grid transmission are subjected to extreme electromagnetic interference and weather conditions, which can lead to accidental damage and frequent failure. In multihop networks, losing individual nodes means losing parts of the network due to loss of functionality by routing, actuating, and sensing issues. To realize self-*, an autonomic-enabled IoT system must adapt to such changes, automatically recover from losses and, thus, be self-protected and capable of failure recovery. In the lower layers, especially at the end device level, failure recovery would refer to the system coping with damaged and malfunctioned sensors. For the complete IoT system, self-adaptation and failure recovery may mean selecting the optimal active response for deploying countermeasures while maintaining system functions. An example of this has been demonstrated in [100]. Failure recovery can be achieved by constantly monitoring the network routes and the status of individual devices. Designing autonomic systems with a built-in mechanism for failure recovery is challenging. The designers have to focus on the actual functionality and restore the functionality upon detection of a failure. Also, remedial actions may include self-recovery in the network by detecting partial failures other than when the system stops working.

C. Self-Healing for Reliability

Autonomic IoT systems may operate in extreme and remote environments requiring multihop communication technology to transmit and receive data. Consequently, studying self-healing and reliability requirements is essential, leading to higher availability despite adverse environmental conditions [101]. Self-healing makes the necessary adjustments to recover from faults so the system can function normally. Regardless of their seriousness, software, network, and hardware defects must not degrade the performance. However, such advanced functions may cost vital memory, processing power and energy resources in constrained devices. Thus, optimization and backward compatibility problems exist since the life cycle for such devices tends to be between 5-10 years. Additionally, developing an ecosystem with autonomic computing principles involves a change in the design philosophy to incorporate the MAPE look principle at the component, device, software frameworks, and system level.

D. Autonomic Intelligence and AI Integration

The use of AI/ML to enable the autonomic and self-management features in IoT systems is an important upcoming research area. Computing areas, such as Web services and data center management software increasingly feature self-managing AI and ML capabilities that let these systems automatically adjust to changing workloads. To manifest intelligent, autonomic behavior in an IoT system, it is essential for individual components, processes, networks, and devices to possess a level of autonomic behavior. Furthermore, cross-layer requirements should have a certain level of dynamism for successful management and decision making. For higher layers, an essential requirement for an intelligent autonomic IoT to exhibit self-* behavior is facilitating local interaction between the components. On the other hand, in the lower layers, software should allow modular handling of various communication modules. The communication layer (or network layer) requires additional features, such as security, multihop routing, and automatic power management. AI-based autonomic systems can use various data sources, including sensor data, to create fault models and make fault detection and maintenance predictive rather than reactive. Due to its flexibility and simplicity of adaptation to a changing environment, AI/ML-based adaptive scheduling is suitable for data-intensive applications. Additionally, it is possible to measure automatically how different QoS attributes affect system performance and adapt to meet the SLA. Reinforcement learning can be used to discover and rectify faults and configure and optimize the system in an unknown environment. This is particularly important as it is challenging to train the ML model on every possible scenario of a dynamic large-scale complex network of IoT. This can increase reliability, but it can also increase system complexity by increasing data processing, resulting in increased computational requirements.

Since the AI algorithms depend on the data, the quality of data may degrade decision making, for example, data being corrupted due to random fluctuations. To prevent this, no outside feedback or interaction should be allowed except for valid environmental interaction. Unfortunately, for a distributed system with control and services spread across all layers and components, this requirement poses a significant challenge. Distributed AI, such as federated learning algorithms, are becoming popular in addressing the data privacy issue and building prediction models with reduced computing requirements for edge devices. Designing and implementing an AI-enabled application is a considerable feat, and adding an extra overhead of autonomic computing is a research challenge that requires coordination and integration across IoT layers. Accuracy and outcome are the most important metrics to determine the success of AI in imparting autonomic computing. In supervised learning, for instance, the accuracy is highly dependent on the training data set. Without proper data, a biased/flawed AI will produce inaccurate or skewed results affecting the operation of the participating autonomic components. Furthermore, in edge computing, the implementation of AI functionality is open and is not considered a built-in capability which results in edge intelligence. Therefore, it is still unclear how and where AI capabilities should be built into edge systems to enable edge intelligence. In order to achieve this, specifications primarily for standardized APIs, software constructs, interoperability mechanisms, and supporting infrastructures need to be developed [102].

E. Security and Privacy

Wireless technologies are particularly vulnerable to security attacks, and security remains a severe challenge in autonomic IoT. In terms of confidentiality and integrity, safeguarding data across all layers is challenging, and developing resource-efficient security schemes in communication and data privacy is essential. A wide variety of attacks exist to target IoT networks and applications. The best defence is for IoT components to self-protect and even self-heal. With the introduction of Blockchain and smart contracts, security and privacy can be decentralized and autonomous. However, Blockchains are not straightforward to be realized in IoT due to several limitations and constitute an interesting research problem. In addition to the smart application and an autonomic structure, designing a secure architecture around the exact requirements is a big challenge. Providing privacy and authentication are two crucial requirements for the IoT. While privacy imposes strong regulations prohibiting any identification or disclosure of personal data, autonomous authentication requires an end device to reveal its identity to some extent. Although they are not opposites, authenticity and privacy security objectives frequently clash. Their relationship is more complicated because a system's authentication may be compromised more easily with private data that is made public. To develop an autonomic authentication system for the IoT, decisions about what constitutes a device identifier must be made to meet the fundamental authentication requirement. The privacy consequences of these choices must be considered and ensure that device identifiers cannot be connected to a specific user or any physical object throughout the entire IoT ecosystem. Security challenges remain a hot area for research, and protecting autonomic IoT will pose a significant challenge for researchers.

F. Trust Issues in IoT

In IoT, the devices need to exchange data to enable them to make decisions in a decentralized manner to implement self-* properties. However, in an IoT network where arbitrary nodes can join and leave the network at any time, some participant nodes may act maliciously and try to exploit the system. Therefore, the concept of trust is of prime importance in realizing full autonomy in IoT. The practical approaches toward trust computing work well when the system is predictable and static. Since modern IoT systems are very dynamic and display unpredictable behaviors, past experiences cannot be utilized to assess the trustworthiness of the system at any given instance. The performance of the overall network may be negatively impacted by trust management if the system changes too frequently while it is in use, such as when new nodes are added. Due to these restrictions, trust management systems (TMSs) find it challenging to adapt to the heterogeneous, autonomous IoT architectures that are prevalent today. Therefore, work is needed to combat various threats in the open distributed environments and new ways of computing trust to enable self-* properties in IoT.

G. Interoperability and Integration of Devices

To communicate with one another, heterogeneous systems, networks, and devices can exchange information with each other. This information is used to participate toward a common system goal or function. The IoT-aided autonomic system typically comprises a heterogeneous, large number of different types of IoT devices and gateways. These components in an autonomic setup vary in resources and operational techniques, such as memory characteristics, computational power, energy, and time sensitivity. On top of the hardware differences, various implemented software stacks follow various standardized communication stacks and protocols (for IP and non-IP-based devices). The lack of device interoperability leads to integration issues due to development constraints and proprietary technology stack. One common technique is converting proprietary technology stub networks into IP-based networks using standardized hardware I/O modules or protocol converters. Furthermore, at the physical level, the availability of a wide variety of wireless protocols (long-range and short-range) creates additional integration issues. On a system level, heterogeneous architectures also lead to interoperability issues. Thus, interoperability issues exist at different levels, from the communication to the physical and application layers. A universal and open approach to services-oriented communication, open architecture devices and semantics are needed to solve the interoperability issue in autonomic IoT systems.

For example, Jaleel et al. [103] proposed an autonomic interoperable manager (AIM) to address the interoperability issue among heterogeneous devices by implementing several self-* to ensure efficient data exchange in time-varying and heterogeneous IoT. AIM uses a service-oriented architecture (SOA) implemented at the fog node. The SOA facilitates the discovery and registration of IoT devices, validates incoming requests for protection against network attacks, manages active device groups, tracks resource utilization, and optimizes it using the MAPE loop.

Furthermore, multiple cross-domain interoperability layers exist, which are intradomain, interdomain, and extra-domain. To fully exploit the benefit of recent IoT developments, interdomain, cross-domain interoperability is required between various vertical IoT applications, e.g., the interoperability of autonomic IoT with smart-cities or smart homes technology stack. For cross-domain interoperability, integration with external IT and enterprise solutions is a significant challenge in interoperability, especially in the business rules domain.

H. Device Management

Platform-based solutions are now widespread for rapid deployment and software development in IoT system development. Sensor and application data in IoT are stored, processed, and transported through these platforms. Another advantage of using platforms is that they enable high levels of interoperability. Access to configuration management and service delivery poses a challenge to achieving self-sufficiency since platforms act as autonomic managers for lower level devices and managed elements for higher application software. Moreover, due to the decentralized and distributed nature of IoT networks, centralized management is not efficient and secure. Therefore, efficient solutions based on fog computing, AI, and Blockchain can be explored for device management.

I. Device Constraints

IoT devices are severely constrained when it comes to resources, such as battery and computational power. To optimize the two, the end devices are tasked with sensing, filtering and forwarding the sensor data to the higher layers, such as an IoT platform. It is necessary to lower the constraints and increase device complexity to allow autonomic self-configuration and actuation of these devices. With the higher computational ability and energy resources, future devices will support self-maintenance, fulfilling another autonomic system goal.

J. Energy Optimisation

The new generation of batteries claims up to over ten years of life [104] at reasonable usage, especially if wireless technologies, such as LORA, SIGFOX, and NB-IoT are used for long-range communication. The scope of optimization exists since there are still restrictions and physical constraints on power usage at the device level. Since most IoT end devices in the stub network operate on batteries, the impact of energy optimization is enormous. Therefore, new energy optimization techniques must be explored along with the technologies, such as efficient energy storage, energy generation, and energy harvesting techniques, that can be used in IoT networks. Furthermore, energy utilization can be reduced by performing local data analysis using edge intelligence rather than sending raw data to the cloud for analysis. For example, edge Intelligence in IoT devices can function with scavenged energy temporarily stored at the IoT device enabled by a new class of electronic device known as FerroElectronics [66].

K. External Factors

In addition to protecting the devices from cybersecurity attacks, it is also essential to protect the devices and sensors from environmental factors. For example, applying autonomic concepts in the smart grid requires sensors to be protected from severe electromagnetic interference (caused by power generation and transmission) via appropriate electromagnetic shielding. Therefore, methods for protection against environmental factors and extreme events can be explored to keep the IoT network functional.

L. Standardisation Requirement

To achieve cross-domain communication, interoperability standardisation is important. Industry and academic players have conducted various standardisation activities in IoT and wireless compatibility [105], [106]. However, there is no standardisation activity specifically for autonomic IoT systems. Therefore, work related to the standardisation of implementing autonomic computing in IoT is required.

VII. Outlook

The future Internet hosting a trillion heterogeneous nodes must be sufficiently smart to handle data flow from all devices and appropriately identify and address the problems. This calls for incorporating autonomic abilities in IoT to enable seamless interoperability and data exchange among heterogeneous devices and systems that can reduce the technical complexity requiring the involvement of the user while providing sophisticated services and applications. As shown in Fig. 5, building an end-to-end autonomic IoT system is a challenging and complex task that requires the implementation of several enabling technologies at each layer of the IoT network and the integration of those technologies across the IoT layers. Furthermore, each layer and the components therein should be able to make decisions independently to a certain extent with the help of the MAPE loop. Implementing the MAPE loop will help realize specific features of the self-* paradigm that will, in turn, result in the implementation of a complete self-* paradigm for an IoT application. This will improve the scalability, security, and reliability of IoT systems while also lowering the operational costs and complexity of managing large-scale IoT deployments by enabling IoT devices and systems to learn from their surroundings, predict future occurrences, and take proactive steps. With autonomic IoT, the role of the user would be limited to setting operational policies and required QoS. Although much effort has been made to implement autonomic features, it has been

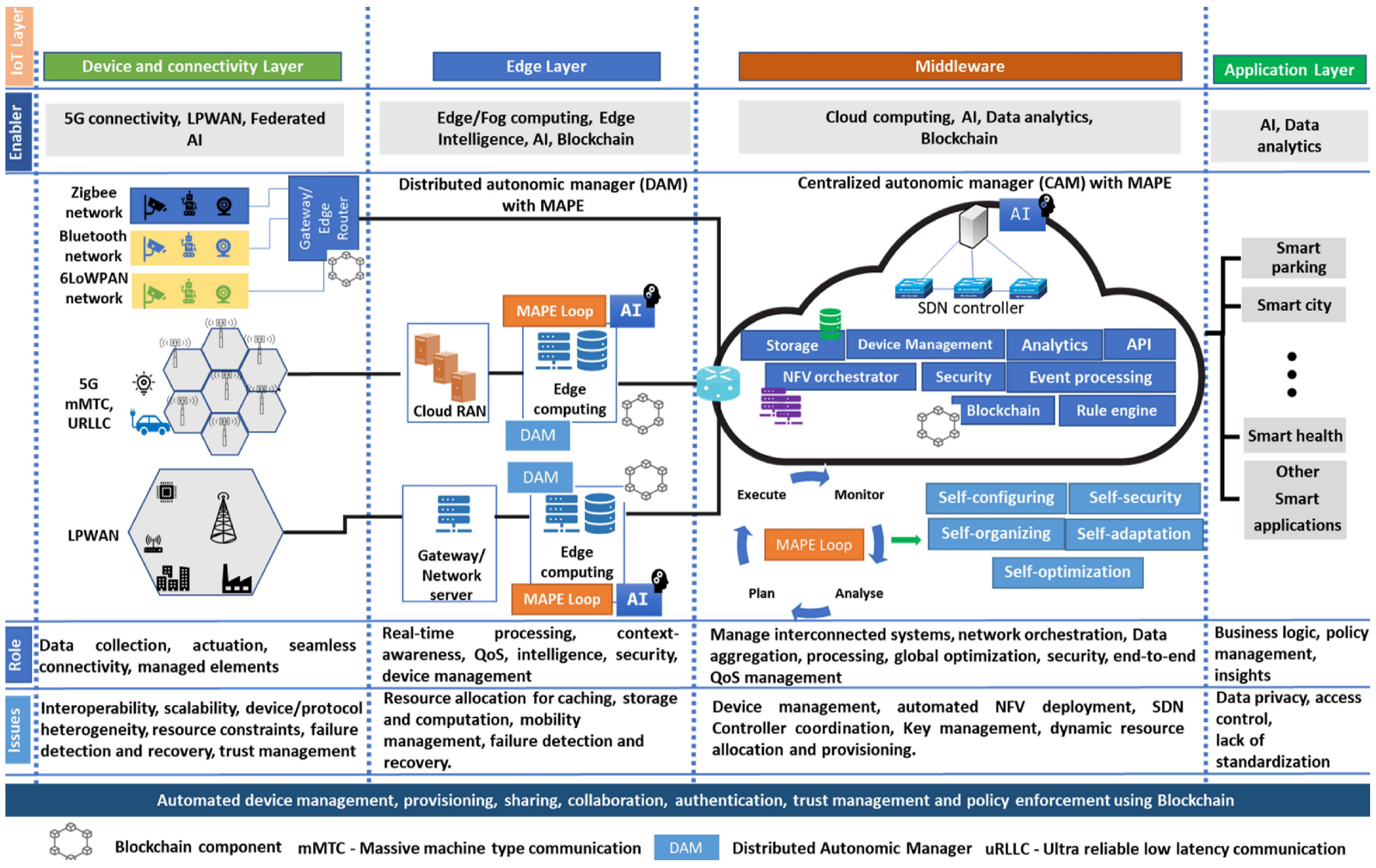
IoT Layer
Device and connectivity Layer
Edge Layer
Middleware
Application Layer
Enabler
5G connectivity, LPWAN, Federated AI
Edge/Fog computing, Edge Intelligence, AI, Blockchain
Cloud computing, AI, Data analytics, Blockchain
AI, Data analytics
Zigbee network
Bluetooth network
6LoWPAN network
Gateway/ Edge Router
5G mMTC, URLLC
LPWAN
Distributed autonomic manager (DAM) with MAPE
MAPE Loop
AI
Cloud RAN
Edge computing
DAM
Gateway/ Network server
Centralized autonomic manager (CAM) with MAPE
SDN controller
Storage
Device Management
Analytics
API
NFV orchestrator
Security
Event processing
Blockchain
Rule engine
Execute
Monitor
Plan
Analyse
Self-configuring
Self-security
Self-organizing
Self-adaptation
Self-optimization
Smart parking
Smart city
Smart health
Other Smart applications
Role
Data collection, actuation, seamless connectivity, managed elements
Real-time processing, context-awareness, QoS, intelligence, security, device management
Manage interconnected systems, network orchestration, Data aggregation, processing, global optimization, security, end-to-end QoS management
Business logic, policy management, insights
Issues
Interoperability, scalability, device/protocol heterogeneity, resource constraints, failure detection and recovery, trust management
Resource allocation for caching, storage and computation, mobility management, failure detection and recovery.
Device management, automated NFV deployment, SDN Controller coordination, Key management, dynamic resource allocation and provisioning.
Data privacy, access control, lack of standardization
Automated device management, provisioning, sharing, collaboration, authentication, trust management and policy enforcement using Blockchain
Blockchain component
mMTC - Massive machine type communication
DAM Distributed Autonomic Manager
uRLLC - Ultra reliable low latency communication

Fig. 5. Overview of envisioned implementation of autonomic capability in IoT using enabling technologies at each layer of IoT ecosystem along with their role and key research challenges.

mostly limited to enabling partial autonomy allowing for certain self-* features owing to several challenges and limitations that exist due to the nature of IoT devices.

Therefore, future research should attempt to achieve the goal of complete yet robust autonomic systems that looks at the integration of all the discussed enabling technologies systematically. Achieving full autonomy is the ultimate goal where a system would need to be able to complete all complex decisions without human intervention. The inclusion of an autonomic paradigm provides additional value through the development of self-managing, self-configuring, and self-optimizing networks. Autonomic computing is not restricted to traditional client–server architectures, managing computing resources and application software. It can also assist in device, network and resource management in a highly dynamic and distributed networked system comprising many smart devices. Therefore, the pursuit to include self-management and autonomic capabilities is expected to become a key driver in developing future IoT solutions.

VIII. Conclusion

To make the IoT system implement self-* properties, it is essential to introduce autonomic behavior in individual components in IoT. A system can be deemed intelligent if it has the power of reasoning and decision making after considering a multitude of parameters. Subsequently, an intelligent, autonomic IoT will be able to make decisions to maximize its operational efficiency by being aware of the operating environment. Intelligence will come from autonomic implementations in each layer of the IoT network. In light of the above observations, the major challenge is to combine traditional concepts of autonomic computing with the distributed, heterogeneous and constrained nature of IoT. To achieve this, self-* properties proposed by IBM can be expanded to more properties, such as self-organization and self-adaptation. This granular treatment of self-* properties allows the designing of the system to be more modular, where each self-* property can be developed independently (e.g., just like in the case of the OSI layer model) and integrated into the IoT network.

In this article, we have taken the first step to compile and specify high-level autonomic requirements and evaluation criteria to achieve autonomic intelligence. It is not required for all autonomic features to exist in a single system or component, such as an end device. Instead, the components of autonomic computing will work together to achieve autonomy in a particular functionality of any component. Furthermore, future IoT will support a wide range of devices and diverse applications requiring varying QoS. To manage the entire operation of the network, the IoT ecosystem as a whole, from the device layer to the application layer and all components in between, must be sufficiently intelligent to handle and act according to the situation. Therefore, IoT must incorporate autonomic capabilities to address device heterogeneity and network complexity in a dynamic environment. To that end, we explored enabling technologies that would aid in realizing the vision of autonomic computing IoT. Finally, challenges and future research directions were discussed that need to be explored in order to realize and enable the vision of autonomic computing in IoT. We believe this work will help to guide and influence future research in autonomic IoT by opening doors to the foundation of self-sufficiency.

References

[1] S. Turber, J. Vom Brocke, O. Gassmann, and E. Fleisch, "Designing business models in the era of Internet of Things: Towards a reference framework," in *Advancing the Impact of Design Science: Moving from Theory to Practice*. Cham, Switzerland: Springer Int., 2014, pp. 17–31.

[2] L. Atzori, A. Iera, and G. Morabito, "The Internet of Things: A survey," *Comput. Netw.*, vol. 54, pp. 2787–2805, Oct. 2010.

[3] D. Miorandi, S. Sicari, F. De Chlamtac, and I. Chlamtac, "Internet of Things: Vision, applications and research challenges," *Ad Hoc Netw.*, vol. 10, pp. 1497–1516, Sep. 2012.

[4] J. O. Kephart and D. M. Chess, "The vision of autonomic computing," *Computer*, vol. 36, no. 1, pp. 41–50, Jan. 2003.

[5] M. Tahir, Q. Mamoon Ashraf, and M. Dabbagh, "Towards enabling autonomic computing in IoT ecosystem," in *Proc. IEEE Int. Conf Depend. Auton. Secure Comput. Int. Conf. Pervasive Intell. Comput. Int. Conf Cloud Big Data Comput. Int. Conf. Cyber Sci. Technol. Congr. (DASC/PiCom/CBDCom/CyberSciTech)*, 2019, pp. 646–651.

[6] P. Lalanda, J. A. McCann, and A. Diaconescu, "Future of autonomic computing and conclusions," in *Autonomic Computing: Principles, Design and Implementation*. London, U.K.: Springer, 2013, pp. 263–278.

[7] M. C. Huebscher and J. A. McCann, "A survey of autonomic computing–degrees, models, and applications," *ACM Comput. Surveys*, vol. 40, no. 3, pp. 1–28, 2008.

[8] C. C. Aggarwal, N. Ashish, and A. Sheth, "The Internet of Things: A survey from the data-centric perspective," in *Managing and Mining Sensor Data*, C. C. Aggarwal, Ed. Boston, MA, USA: Springer, 2013, pp. 383–428.

[9] L. Da Xu, W. He, and S. Li, "Internet of Things in industries: A survey," *IEEE Trans. Ind. Informat.*, vol. 10, no. 4, pp. 2233–2243, Nov. 2014.

[10] H.-D. Ma, "Internet of Things: Objectives and scientific challenges," *J. Comput. Sci. Technol.*, vol. 26, no. 6, pp. 919–924, 2011.

[11] Q. M. Ashraf and M. H. Habaebi, "Autonomic schemes for threat mitigation in Internet of Things," *J. Netw. Comput. Appl.*, vol. 49, pp. 112–127, Mar. 2015.

[12] P. P. Ray, "A survey on Internet of Things architectures," *J. King Saud Univ.-Comput. Inf. Sci.*, vol. 30, no. 3, pp. 291–319, 2018.

[13] C. Savaglio and G. Fortino, "Autonomic and cognitive architectures for the Internet of Things," in *Proc. Int. Conf. Internet Distrib. Comput. Syst.*. 2015, pp. 39–47.

[14] M. Ammar, G. Russello, and B. Crispo, "Internet of Things: A survey on the security of IoT frameworks," *J. Inf. Secur. Appl.*, vol. 38, pp. 8–27, Feb. 2018.

[15] A. R. Sfar, E. Natalizio, Y. Challal, and Z. Chtourou, "A roadmap for security challenges in the Internet of Things," *Digit. Commun. Netw.*, vol. 4, no. 2, pp. 118–137, 2018.

[16] K. Sood, S. Yu, and Y. Xiang, "Software-defined wireless networking opportunities and challenges for Internet-of-Things: A review," *IEEE Internet Things J.*, vol. 3, no. 4, pp. 453–463, Aug. 2016.

[17] O. B. Sezer, E. Dogdu, and A. M. Ozbayoglu, "Context-aware computing, learning, and big data in Internet of Things: A survey," *IEEE Internet Things J.*, vol. 5, no. 1, pp. 1–27, Feb. 2018.

[18] K. Fizza et al., "A survey on evaluating the quality of autonomic Internet of Things applications," *IEEE Commun. Surveys Tuts.*, vol. 25, no. 1, pp. 567–590, 1st Quart., 2023.

[19] A. Čolaković and M. Hadžialić, "Internet of Things (IoT): A review of enabling technologies, challenges, and open research issues," *Comput. Netw.*, vol. 144, pp. 17–39, Oct. 2018.

[20] P. Dehraj and A. Sharma, "A review on architecture and models for autonomic software systems," *J. Supercomput.*, vol. 77, no. 1, pp. 388–417, 2021.

[21] A. H. Ngu, M. Gutierrez, V. Metsis, S. Nepal, and Q. Z. Sheng, "IoT middleware: A survey on issues and enabling technologies," *IEEE Internet Things J.*, vol. 4, no. 1, pp. 1–20, Feb. 2017.

[22] L. Lei, Y. Tan, K. Zheng, S. Liu, K. Zhang, and X. Shen, "Deep reinforcement learning for autonomous Internet of Things: Model, applications and challenges," *IEEE Commun. Surveys Tuts.*, vol. 22, no. 3, pp. 1722–1760, 3rd Quart., 2020.

[23] N. Agrawal, "Autonomic cloud computing based management and security solutions: State-of-the-art, challenges, and opportunities," *Trans. Emerg. Telecommun. Technol.*, vol. 32, no. 12, 2021, Art. no. e4349.

[24] S. S. Gill et al., "AI for next generation computing: Emerging trends and future directions," *Internet Things*, vol. 19, Aug. 2022, Art. no. 100514.

[25] R. F. Sari, L. Rosyidi, B. Susilo, and M. Asvial, "A comprehensive review on network protocol design for autonomic Internet of Things," *Information*, vol. 12, no. 8, p. 292, 2021.

[26] J. Koehler, C. Giblin, D. Gantenbein, and R. Hauser, "On autonomic computing architectures," IBM Res., Zurich Res. Lab., Rüschlikon, Switzerland, Rep. RZ3487, 2003.

[27] E. Vassev and M. Hinchey, "The ASSL formalism for real-time autonomic systems," in *Self-Organization in Embedded Real-Time Systems*. New York, NY, USA: Springer, 2013, pp. 151–177.

[28] A. Ramakrishnan, S. N. Z. Naqvi, Z. W. Bhatti, D. Preuveneers, and Y. Berbers, "Learning deployment trade-offs for self-optimization of Internet of Things applications," in *Proc. 10th Int. Conf. Auton. Comput. (ICAC)*, 2013, pp. 213–224.

[29] R. Mahmud, R. Kotagiri, and R. Buyya, "Fog computing: A taxonomy, survey and future directions," in *Internet of Everything*. Singapore, Springer, 2018, pp. 103–130.

[30] I. Chun, J. Park, W. Kim, W. Kang, H. Lee, and S. Park, "Autonomic computing technologies for cyber-physical systems," in *Proc. 12th Int. Conf. Adv. Commun. Technol. (ICACT)*, vol. 2, 2010, pp. 1009–1014.

[31] O. Babaoglu et al., *Self-Star Properties in Complex Information Systems: Conceptual and Practical Foundations*, vol. 3460. Berlin, Germany: Springer, 2005.

[32] M. C. Domingo, "An overview of the Internet of Underwater Things," *J. Netw. Comput. Appl.*, vol. 36, pp. 1879–1890, Nov. 2012.

[33] S. Marchal, M. Miettinen, T. D. Nguyen, A. Sadeghi, and N. Asokan, "AuDI: Toward autonomous IoT device-type identification using periodic communication," *IEEE J. Sel. Areas Commun.*, vol. 37, no. 6, pp. 1402–1412, Jun. 2019.

[34] T. D. Nguyen, S. Marchal, M. Miettinen, H. Fereidooni, N. Asokan, and A.-R. Sadeghi, "DïoT: A federated self-learning anomaly detection system for IoT," in *Proc. IEEE 39th Int. Conf. Distrib. Comput. Syst. (ICDCS)*, 2019, pp. 756–767.

[35] A. Molina Zarca et al., "Security management architecture for NFV/SDN-aware IoT systems," *IEEE Internet Things J.*, vol. 6, no. 5, pp. 8005–8020, Oct. 2019.

[36] F. Dressler, "A study of self-organization mechanisms in ad hoc and sensor networks," *Comput. Commun.*, vol. 31, pp. 3018–3029, Aug. 2008.

[37] M. C. Lucas-Estañ and J. Gozalvez, "Load balancing for reliable self-organizing Industrial IoT networks," *IEEE Trans. Ind. Informat.*, vol. 15, no. 9, pp. 5052–5063, Sep. 2019.

[38] V. Behzadan and B. Rekabdar, "A game-theoretic model for analysis and design of self-organization mechanisms in IoT," in *Proc. Int. Conf. Game Theory Netw.*, 2017, pp. 74–85.

[39] P. Semasinghe, S. Maghsudi, and E. Hossain, "Game theoretic mechanisms for resource management in massive wireless IoT systems," *IEEE Commun. Mag.*, vol. 55, no. 2, pp. 121–127, Feb. 2017.

[40] A. Gatouillat, Y. Badr, and B. Massot, "QoS-driven self-adaptation for critical IoT-based systems," in *Proc. Int. Conf. Service-Orient. Comput.*, 2017, pp. 93–105.

[41] P. Arcaini, E. Riccobene, and P. Scandurra, "Modeling and analyzing MAPE-K feedback loops for self-adaptation," in *Proc. IEEE/ACM 10th Int. Symp. Softw. Eng. Adapt. Self-Manag. Syst.*, 2015, pp. 13–23.

[42] I. Azimi et al., "HiCH: Hierarchical fog-assisted computing architecture for healthcare IoT," *ACM Trans. Embedded Comput. Syst.*, vol. 16, no. 5s, pp. 1–20, 2017.

[43] E. Lee, Y.-G. Kim, Y.-D. Seo, K. Seol, and D.-K. Baik, "RINGA: Design and verification of finite state machine for self-adaptive software at runtime," *Inf. Softw. Technol.*, vol. 93, pp. 200–222, Jan. 2018.

[44] J. Zheng, Y. Cai, X. Shen, Z. Zheng, and W. Yang, "Green energy optimization in energy harvesting wireless sensor networks," *IEEE Commun. Mag.*, vol. 53, no. 11, pp. 150–157, Nov. 2015.

[45] N. Chen, T. Qiu, X. Zhou, K. Li, and M. Atiquzzaman, "An intelligent robust networking mechanism for the Internet of Things," *IEEE Commun. Mag.*, vol. 57, no. 11, pp. 91–95, Nov. 2019.

[46] Q. M. Ashraf, M. H. Habaebi, G. R. Sinniah, and J. Chebil, "Broadcast based registration technique for heterogenous nodes in the IoT," in *Proc. Int. Conf. Control Eng. Inf. Technol. (CEIT)*, 2014, pp. 45–50.

[47] B. Kang, D. Kim, and H. Choo, "Internet of everything: A large-scale autonomic IoT gateway," *IEEE Trans. Multi-Scale Comput. Syst.*, vol. 3, no. 3, pp. 206–214, Jul.-Sep. 2017.

[48] M. Slabicki, G. Premsankar, and M. Di Francesco, "Adaptive configuration of LoRa networks for dense IoT deployments," in *Proc. IEEE/IFIP Netw. Operat. Manage. Symp.*, 2018, pp. 1–9.

[49] W. Dai, V. N. Dubinin, J. H. Christensen, V. Vyatkin, and X. Guan, "Toward self-manageable and adaptive industrial cyber-physical systems with knowledge-driven autonomic service management," *IEEE Trans. Ind. Informat.*, vol. 13, no. 2, pp. 725–736, Apr. 2017.

[50] R. Sterritt and M. Hinchey, "SPAACE IV: Self-properties for an autonomous & autonomic computing environment–part IV a newish hope," in *Proc. 7th IEEE Int. Conf. Workshops Eng. Autonom. Auton. Syst.*, 2010, pp. 119–125.

[51] A. Berns and S. Ghosh, "Dissecting self-* properties," in *Proc. 3rd IEEE Int. Conf. Self-Adapt. Self-Organiz. Syst.*, 2009, pp. 10–19.

[52] W. F. Truszkowski, M. G. Hinchey, J. L. Rash, and C. A. Rouff, "Autonomous and autonomic systems: A paradigm for future space exploration missions," *IEEE Trans. Syst., Man, Cybern. C, Appl. Rev.*, vol. 36, no. 3, pp. 279–291, May 2006.

[53] B. Maati and D. E. Saidouni, "Ciotas protocol: Cloudiot available services protocol through autonomic computing against distributed denial of services attacks," *J. Ambient Intell. Humanized Comput.*, pp. 1–30, Oct. 2020.

[54] Z. Al-Shara, F. Alvares, H. Bruneliere, J. Lejeune, C. Prud Homme, and T. Ledoux, "Come4acloud: An end-to-end framework for autonomic cloud systems," *Future Gener. Comput. Syst.*, vol. 86, pp. 339–354, Sep. 2018.

[55] M. Ghobaei-Arani, S. Jabbehdari, and M. A. Pourmina, "An autonomic resource provisioning approach for service-based cloud applications: A hybrid approach," *Future Gener. Comput. Syst.*, vol. 78, pp. 191–210, Jan. 2018.

[56] F. Bonomi, R. Milito, J. Zhu, and S. Addepalli, "Fog computing and its role in the Internet of Things," in *Proc. 1st MCC Workshop Mobile Cloud Comput.*, 2012, pp. 13–16.

[57] O. Vermesan et al., "Internet of things cognitive transformation technology research trends and applications," *Cognitive Hyperconnected Digital Transformation*. Denmark, U.K.: River Publ., 2017, p. 79.

[58] U. Tadakamalla and D. A. Menascé, "Autonomic resource management for fog computing," *IEEE Trans. Cloud Comput.*, vol. 10, no. 4, pp. 2334–2350, Oct.-Dec. 2022.

[59] H. V. Sampaio, C. B. Westphall, F. Koch, R. do Nascimento Boing, and R. N. Santa Cruz, "Autonomic energy management with fog computing," *Comput. Elect. Eng.*, vol. 93, Jul. 2021, Art. no. 107246.

[60] P. Kayal and J. Liebeherr, "Autonomic service placement in fog computing," in *Proc. IEEE 20th Int. Symp. World Wireless Mobile Multimedia Netw. (WoWMoM)*, 2019, pp. 1–9.

[61] M. Etemadi, M. Ghobaei-Arani, and A. Shahidinejad, "Resource provisioning for IoT services in the fog computing environment: An autonomic approach," *Comput. Commun.*, vol. 161, pp. 109–131, Sep. 2020.

[62] M. Ghobaei-Arani and A. Shahidinejad, "A cost-efficient IoT service placement approach using whale optimization algorithm in fog computing environment," *Expert Syst. Appl.*, vol. 200, Aug. 2022, Art. no. 117012.

[63] F. Jazayeri, A. Shahidinejad, and M. Ghobaei-Arani, "Autonomous computation offloading and auto-scaling the in the mobile fog computing: A deep reinforcement learning-based approach," *J. Ambient Intell. Humanized Comput.*, vol. 12, pp. 8265–8284, Aug. 2021.

[64] W. Shi, J. Cao, Q. Zhang, Y. Li, and L. Xu, "Edge computing: Vision and challenges," *IEEE Internet Things J.*, vol. 3, no. 5, pp. 637–646, Oct. 2016.

[65] B. Chen, J. Wan, A. Celesti, D. Li, H. Abbas, and Q. Zhang, "Edge computing in IoT-based manufacturing," *IEEE Commun. Mag.*, vol. 56, no. 9, pp. 103–109, Sep. 2018.

[66] A. Keshavarzi and W. van den Hoek, "Edge intelligence—On the challenging road to a trillion smart connected IoT devices," *IEEE Des. Test.*, vol. 36, no. 2, pp. 41–64, Apr. 2019.

[67] K. Bajaj, B. Sharma, and R. Singh, "Implementation analysis of IoT-based offloading frameworks on cloud/edge computing for sensor generated big data," *Complex Intell. Syst.*, vol. 8, pp. 3641–3658, Oct. 2021.

[68] I. Sittón-Candanedo, R. S. Alonso, S. García, A. B. Gil, and S. Rodríguez-González, "A review on edge computing in smart energy by means of a systematic mapping study," *Electronics*, vol. 9, no. 1, p. 48, 2020.

[69] Q. M. Ashraf et al., "Autonomic Internet of Things for enforced demand management in smart grid," *Amer. J. Data Mining Knowl. Discov.*, vol. 2, no. 2, pp. 69–75, 2017.

[70] C. Savaglio, P. Gerace, G. Di Fatta, and G. Fortino, "Data mining at the IoT edge," in *Proc. 28th Int. Conf. Comput. Commun. Netw. (ICCCN)*, 2019, pp. 1–6.

[71] I. A. Elgendy, W. Zhang, Y.-C. Tian, and K. Li, "Resource allocation and computation offloading with data security for mobile edge computing," *Future Gener. Comput. Syst.*, vol. 100, pp. 531–541, Nov. 2019.

[72] M. S. Aslanpour, S. S. Gill, and A. N. Toosi, "Performance evaluation metrics for cloud, fog and edge computing: A review, taxonomy, benchmarks and standards for future research," *Internet Things*, vol. 12, Dec. 2020, Art. no. 100273.

[73] P. Veena, S. Panikkar, S. Nair, and P. Brody, "Empowering the edge-practical insights on a decentralized Internet of Things," in *Empowering Edge-Practical Insights a Decentralized Internet of Things*, vol. 17. Armonk, NY, USA: IBM Inst. Bus. Value, 2015.

[74] X. Zhu, Y. Badr, J. Pacheco, and S. Hariri, "Autonomic identity framework for the Internet of Things," in *Proc. Int. Conf. Cloud Auton. Comput. (ICCAC)*, 2017, pp. 69–79.

[75] M. Mukhandi, F. Damião, J. Granjal, and J. P. Vilela, "Blockchain-based device identity management with consensus authentication for IoT devices," in *Proc. IEEE 19th Annu. Consum. Commun. Netw. Conf. (CCNC)*, 2022, pp. 433–436.

[76] A. Dawod, D. Georgakopoulos, P. P. Jayaraman, A. Nirmalathas, and U. Parampalli, "IoT device integration and payment via an autonomic blockchain-based service for IoT device sharing," *Sensors*, vol. 22, no. 4, p. 1344, 2022.

[77] L. Ochoa-Aday, C. Cervelló-Pastor, A. Fernández-Fernández, and P. Grosso, "An Online algorithm for dynamic NFV placement in cloud-based autonomous response networks," *Symmetry*, vol. 10, no. 5, p. 163, 2018.

[78] C. A. Ouedraogo, E.-F. Bonfoh, S. Medjiah, C. Chassot, and S. Yangui, "A prototype for dynamic provisioning of QoS-oriented virtualized network functions in the Internet of Things," in *Proc. 4th IEEE Conf. Netw. Softw. Workshops (NetSoft)*, 2018, pp. 323–325.

[79] B. R. Al-Kaseem and H. S. Al-Raweshidy, "SD-NFV as an energy efficient approach for M2M networks using cloud-based 6LoWPAN testbed," *IEEE Internet Things J.*, vol. 4, no. 5, pp. 1787–1797, Oct. 2017.

[80] S. Sarma and J. S. Siegel, *Industrial Intelligence: AI's Implications on Security, Seamlessness and Services for the IIoT*, Ind. Internet Consortium J. Innov., Boston, MA, USA, 2017.

[81] M. G. R. Alam, M. M. Hassan, M. Z. Uddin, A. Almogren, and G. Fortino, "Autonomic computation offloading in mobile edge for IoT applications," *Future Gener. Comput. Syst.*, vol. 90, pp. 149–157, Jan. 2019.

[82] M. Xue, H. Wu, G. Peng, and K. Wolter, "DDPQN: An efficient DNN offloading strategy in local-edge-cloud collaborative environments," *IEEE Trans. Services Comput.*, vol. 15, no. 2, pp. 640–655, Mar./Apr. 2022.

[83] S. Dobson et al., "A survey of autonomic communications," *ACM Trans. Auton. Adapt. Syst.*, vol. 1, no. 2, pp. 223–259, 2006.

[84] J. Mitola, "Cognitive radio architecture evolution," *Proc. IEEE*, vol. 97, no. 4, pp. 626–641, Apr. 2009.

[85] A. Riker, R. Mota, D. Rosário, V. Pereira, and M. Curado, "Autonomic management of group communication for Internet of Things applications," *Int. J. Commun. Syst.*, vol. 35, no. 11, 2022, Art. no. e5200.

[86] Y. Sahni, J. Cao, S. Zhang, and L. Yang, "Edge mesh: A new paradigm to enable distributed intelligence in Internet of Things," *IEEE Access*, vol. 5, pp. 16441–16458, 2017.

[87] J. Zhu, Y. Song, D. Jiang, and H. Song, "A new deep-Q-learning-based transmission scheduling mechanism for the cognitive Internet of Things," *IEEE Internet Things J.*, vol. 5, no. 4, pp. 2375–2385, Aug. 2018.

[88] J. Wang, L. Zhao, J. Liu, and N. Kato, "Smart resource allocation for mobile edge computing: A deep reinforcement learning approach," *IEEE Trans. Emerg. topics Comput.*, vol. 9, no. 3, pp. 1529–1541, Jul.–Sep. 2021.

[89] J. Chen, S. Chen, Q. Wang, B. Cao, G. Feng, and J. Hu, "iRAF: A deep reinforcement learning approach for collaborative mobile edge computing IoT networks," *IEEE Internet Things J.*, vol. 6, no. 4, pp. 7011–7024, Aug. 2019.

[90] X. Qiu, L. Liu, W. Chen, Z. Hong, and Z. Zheng, "Online deep reinforcement learning for computation offloading in blockchain-empowered mobile edge computing," *IEEE Trans. Veh. Technol.*, vol. 68, no. 8, pp. 8050–8062, Aug. 2019.

[91] Y. Zhao et al., "Privacy-preserving blockchain-based federated learning for IoT devices," *IEEE Internet Things J.*, vol. 8, no. 3, pp. 1817–1829, Feb. 2021.

[92] Y. Lu, X. Huang, Y. Dai, S. Maharjan, and Y. Zhang, "Blockchain and federated learning for privacy-preserved data sharing in industrial IoT," *IEEE Trans. Ind. Informat.*, vol. 16, no. 6, pp. 4177–4186, Jun. 2020.

[93] A. M. Zarca, J. B. Bernabe, A. Skarmeta, and J. M. Alcaraz Calero, "Virtual IoT HoneyNets to mitigate cyberattacks in SDN/NFV-enabled IoT networks," *IEEE J. Sel. Areas Commun.*, vol. 38, no. 6, pp. 1262–1277, Jun. 2020.

[94] C. Benzaid, T. Taleb, and J. Song, "AI-based autonomic & scalable security management architecture for secure network slicing in B5G," *IEEE Netw.*, vol. 36, no. 6, pp. 165–174, Nov./Dec. 2022.

[95] R. Poor, C. Bowman, and C. B. Auburn, "Self-healing networks," *ACM Queue*, vol. 1, no. 3, pp. 52–59, 2003.

[96] Ö. U. Akgül and B. Canberk, "Self-Organized things (SoT): An energy efficient next generation network management," *Comput. Commun.*, vol. 74, pp. 52–62, Jan. 2016.

[97] R. Tynan, G. M. O'Hare, and A. Ruzzelli, "Autonomic wireless sensor network topology control," in *Proc. IEEE Int. Conf. Netw. Sens. Control*, 2007, pp. 7–13.

[98] C. P. Edwards, D. G. Leeper, R. I. Foster, R. O. Waddoups, and S. M. Daniel "Self-organizing network with decision engine and method," U.S. Patent 6 870 816 B1, Mar. 2005.

[99] W. Shi, N. Kumar, P. Gong, N. Chilamkurti, and H. Chang, "On the security of a certificateless Online/offline signcryption for Internet of Things," *Peer Peer Netw. Appl.*, vol. 8, no. 5, pp. 881–885, 2015.

[100] Q. Chen, S. Abdelwahed, and A. Erradi, "A model-based validated autonomic approach to self-protect computing systems," *IEEE Internet Things J.*, vol. 1, no. 5, pp. 446–460, Oct. 2014.

[101] Q. Ou, Y. Zhen, X. Li, Y. Zhang, and L. Zeng, "Application of Internet of Things in smart grid power transmission," in *Proc. 3rd FTRA Int. Conf. Mobile Ubiquitous Intell. Comput.*, 2012, pp. 96–100.

[102] V. Barbuto, C. Savaglio, M. Chen, and G. Fortino, "Disclosing edge intelligence: A systematic meta-survey," *Big Data Cogn. Comput.*, vol. 7, no. 1, p. 44, 2023.

[103] A. Jaleel, T. Mahmood, A. Tahir, S. Aslam, and U. U. Fayyaz, "Autonomic interoperability manager: A service-oriented architecture for full-stack interoperability in the Internet-of-Things," *ICT Exp.*, vol. 8, no. 4, pp. 507–512, 2022.

[104] J. Steiner, M. Blakeley, and A. Miller, "Estimating the battery life of a wireless occupancy sensor," U.S. Dept. Energy, Lutron Electron. Co., Inc., Coopersburg, PA, USA, Rep. 367–2437, 2014.

[105] M. R. Palattella et al., "Standardized protocol stack for the Internet of (Important) Things," *IEEE Commun. Surveys Tuts.*, vol. 15, no. 3, pp. 1389–1406, 3rd Quart., 2013.

[106] Z. Sheng, S. Yang, Y. Yu, A. V. Vasilakos, J. A. Mccann, and K. K. Leung, "A survey on the ietf protocol suite for the Internet of Things: Standards, challenges, and opportunities," *IEEE Wireless Commun.*, vol. 20, no. 6, pp. 91–98, Dec. 2013.

Qazi Mamoon Ashraf received the Ph.D. degree in computer engineering from University Islam Antarabangsa, Kuala Lumpur, Malaysia, in 2016.

He worked as a Research Assistant with the Wireless Communication Division, MIMOS Malaysia, Kuala Lumpur, and later joined Telekom Malaysia Research and Development, Cyberjaya, Malaysia, as a Researcher with the IoT Unit, where he currently serves as a Product Manager for the Research and Innovation Division. His research interests include product management, IoT, smart wearables, autonomic computing, ubiquitous networks, and secure M2M communication.

Mohammad Tahir (Senior Member, IEEE) received the M.Sc. and Ph.D. degrees in electrical and computer engineering from the Department of Electrical and Computer Engineering, International Islamic University Malaysia, Kuala Lumpur, Malaysia, in 2011 and 2016, respectively.

He is currently working as a Lecturer with the Cybersecurity And Communication Engineering Laboratory, University of Turku, Turku, Finland. Prior to joining academics, he worked with the Research and Development Division of industry for seven years on several projects related to the Internet of Things and cognitive radio. His research interests include 5G and beyond, Internet of Things, game theory, and applied ML for wireless networks, wireless security, and autonomic computing.

Mohamed Hadi Habaebi (Senior Member, IEEE) received the degree from the Civil Aviation and Meteorology High Institute, Tripoli, Libya, in 1991, the M.Sc. degree in electrical engineering from Universiti Teknologi Malaysia, Skudai, Malaysia, in 1994, and the Ph.D. degree in computer and communication system engineering from University Putra Malaysia, Serdang, Malaysia, in 2001.

He is currently a full-time Professor and the Post Graduate Academic Advisor with the Department of Electrical and Computer Engineering, International Islamic University Malaysia, Kuala Lumpur, Malaysia, where he heads the research works on the Internet of Things. He has supervised many Ph.D. and M.Sc. students and published more than 120 articles and papers. He is actively publishing in M2M communication protocols, wireless sensor and actuator networks, cognitive radio, small antenna system and radio propagation, and wireless communications and network performance evaluation.

Dr. Habaebi is an Active Member of IEEE and an active reviewer of many international journals and sits on the Editorial Boards of many international journals.

Jouni Isoaho received the M.Sc. (Tech) degree in electrical engineering, and the Lic.Tech. and Dr.Tech. degrees in information technology from Tampere University of Technology, Tampere, Finland, in 1989, 1992, and 1995, respectively.

Since 1999, he has been a Professor with the University of Turku, Turku, Finland. The core of his research is communication and cybersecurity technologies. His current ongoing research activities include autonomous systems technology, security and communications, human and societal cybersecurity, and smart technology and digitalization.