Edge Computing, Fog and MIST Architecture Analysis, Application, and Challenges

by

Maryam Karimi Makiabadi

MINT 709 Capstone Project Report submitted in partial fulfillment of the requirements for the degree of

Master of Internetworking

Department of Computing Science
University of Alberta

Abstract

With the advent of IoT and next-generation internet which contains a multitude of geographically distributed devices (approx. 50 billion devices by 2020 [1]) continuously generating data streams, the centralized cloud computing structure is becoming inefficient for processing and analyzing an extraordinary volume of data collected from IoT devices. Therefore, we need new data processing architectures that can handle the challenges of heterogeneity, distribution, latency, and bandwidth along the IoT-cloud continuum. Fog computing [2] emerged as a new computing paradigm that changed the distributed computing landscape by extending the computing power and data analytics to the proximity of users from the core to the edge of the network. It is potentially built with various components and resource types distributed across the IoT-Fog-Cloud continuum.

In this project, we first introduce the notions of cloud computing, Fog Computing, Edge Computing, and Mist Computing. Then, compare these architectures with other related computing paradigms. Finally, we conclude the project by addressing Challenges and future directions.

Acknowledgements

I would like to thank the following people, without whom I would not have been able to complete this Capstone project.

First of all, I would like to express my sincere gratitude to *Mr. Juned Noonari* for the thoughtful comments and recommendations during this project. Further, I am delighted to thank my program coordinator *Mr. Shahnawaz Mir* for his guidance, overall insights in this field ,and support to opt for a project of my interest. I am also so grateful to *Prof. Mike MacGregor* for providing me such a great opportunity to start this project. Last but not least, my biggest thanks to my family and friends for all the support they have shown me during this journey.

Table of Contents

List of Tables

List of Figures

1 Introduction

2 Cloud Computing

The key aspects of Cloud Computing were documented in the definition provided by the National Institute of Standard and Technologies(NIST)[3]: "Cloud computing is a model for enabling ubiquitous, convenient, on-demand network access to a shared pool of configurable computing resources (e.g., networks, servers,storage, applications, and services) that can be rapidly provisioned and released with minimal management effort or service provider interaction". While the principal concept behind cloud computing wasn't new, In recent years, cloud computing has attracted great attention and frameworks like Amazon Web Services, Microsoft Azure and Google Cloud Platform have gained a lot of popularity among cloud consumers.

Large businesses (such as Amazon, Google, Facebook, etc.) have generally adopted this model for offering services over the Internet, gaining economic and technological benefits. National Institute of Standards and Technology (NIST) introduced cloud computing with five essential characteristics: (i) on-demand self-service, (ii) rapid elasticity or expansion, (iii) broad network access, (iv) resource pooling, and (v) measured services. Furthermore, cloud computing can be defined as a flexible and scalable platform to provide virtualized resources to end-users through the Internet. Figure 1 shows the conventional cloud computing structure. Data producers generate raw data and transfer it to the cloud.

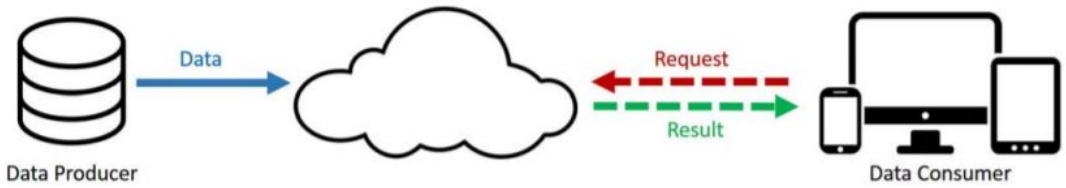

Figure 1: Cloud computing paradigm [4]

2.1 Cloud Computing Characteristics

Cloud computing has some essential characteristics that are summarized below:

- **On-demand self-service:** computer capabilities can be supported automatically when needed without needing any human contact between customers and service providers.

- **Broad network access:** Computing capabilities are available across the network and available for a wide variety of client platforms (e.g. workstations, laptops and mobile devices) through many mechanisms.

- **Resource pooling:** To satisfy multiple users, computing resources are pooled, dynamically allocated, and distributed according to customer demand. Besides, the services of the provider are independent of location, i.e. the customer has no knowledge or influence of their exact location.

- **Rapid elasticity:** It is possible to flexibly have and release computing capabilities to scale in and out according to demand. The user thus has the perception of limitless, and often necessary, computing capabilities.

- **Measured service:** It is possible to track and disclose the use of resources according to the type of service provided. In charge per use, or pay per user, services, this is especially important because it gives great clarity between the provider and the customer of the service.

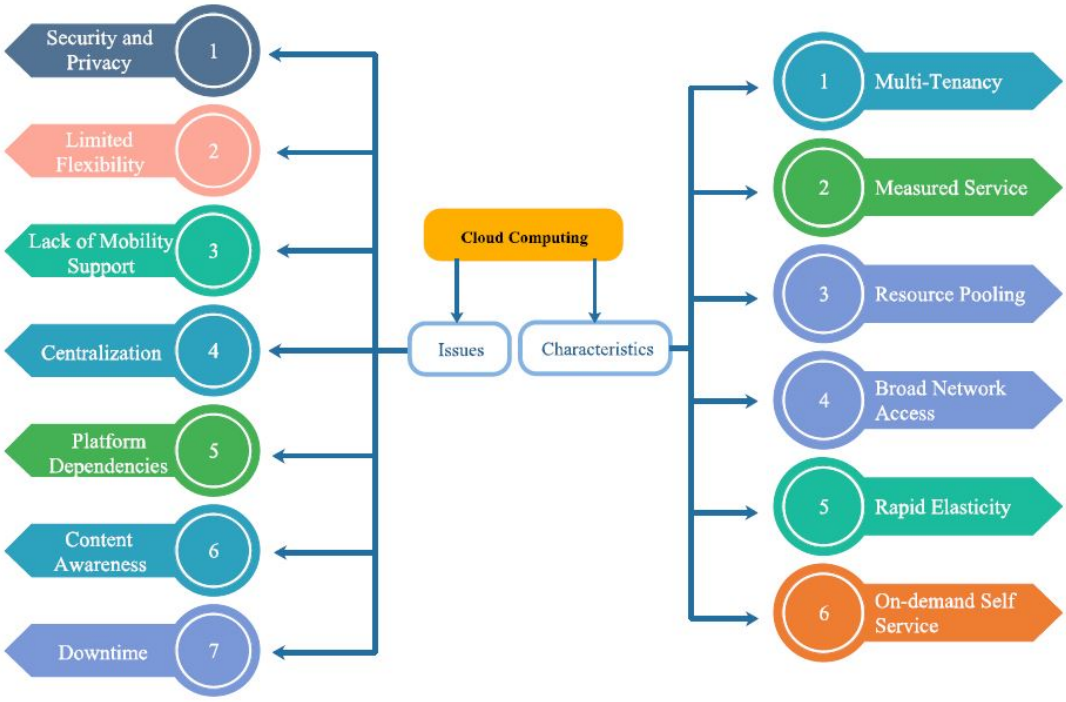

Figure 2: Cloud Computing Characteristics and issues [4]

2.2 Cloud Computing Architecture

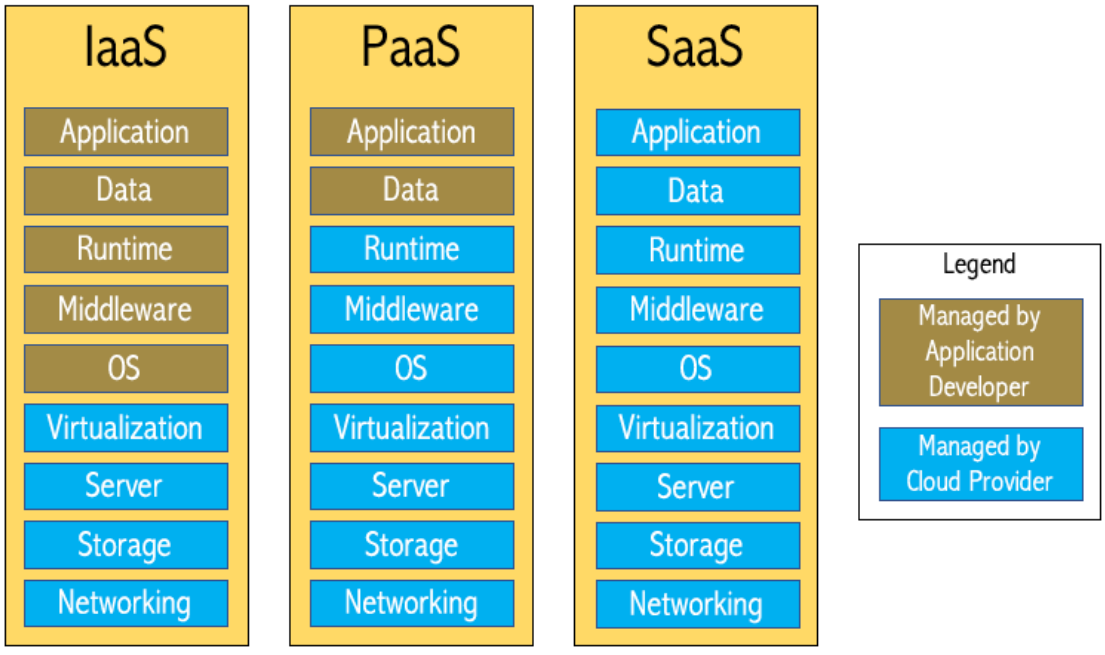

Figure 3: Different Cloud Service Model [5]

The architecture of the Cloud can be divided into four layers: datacenter (hardware), infrastructure, platform, and application [13]. Each one can be seen as a service for the upper layer and as a consumer for the lower layer. In practice, Cloud services can be categorized into three main services. namely: (a) Infrastructure-as-a-Service (IaaS), which enables the provision of computing, storage, and networking elements (e.g. Amazon Elastic Compute Cloud (EC2)), (b) Platform-as-a-Service (PaaS), which offers an integrated platform to deploy, test and verify custom applications (e.g. Google App Engine), and (c) Software-as-a-Service (SaaS), which supports software distribution with specific requirements (e.g. SalesForce). Figure 3depicts the layered organization of the cloud stack from physical infrastructure to applications. These levels of abstraction can also be seen as layered architecture in which Higher-level services may be composed of underlying layer services[6]

2.3 Infrastructure as a Service

Facilitate users to manage, storage, networks, process, and other significant computing resources so they can publish and run applications that include operating systems and/or apps. The user does not manage or operate the cloud infrastructure for the hardware but has control over operating environments, deployed software, and storage and select components for networking.[7] Maintaining the IT infrastructure on-premise is expensive and labor intensive. It also needs large initial investment in physical hardware, and then you will need to hire external IT contractors to support the hardware and keep it up to date and running.With IaaS, you can buy what you need, and buy more as your company grows. Iaas provides flexibility, scalability and can be replaced if you need them without losing money on your initial investment. Another advantage of IaaS is that it puts control of the infrastructure back in your hands. You do not need to trust an external IT contractor, if you wish, you can access and monitor IaaS platforms yourself and supports companies of all shapes and sizes as it provides full control of your infrastructure and works on a pay-as-you-use

model, making it ideal for most budgets.

A good example of IaaS is AWS Elastic Compute Cloud (EC2). It offers the flexible infrastructure for businesses wishing to host cloud-based content. EC2 users do not own the physical servers. AWS provides virtual servers, So users only pay for the server use, saving them the cost (and related ongoing maintenance) of investing in physical hardware.

2.4 Platform as a service

In addition to infrastructure-oriented clouds that provide raw computing and storage services, another approach is to offer a higher level of abstraction to make a cloud easily programmable, known as Platform as a service (PaaS). The provider provides the virtual environments and application programming interfaces (APIs) framework that can be used in cloud application development. Of course, the users of this class are developers who use specific APIs on the cloud platform to create, test, deploy, and integrate their apps. One example is Google App Engine[1], which offers runtime environments for Python and Java, as well as APIs for applications to communicate with the runtime environment. Microsoft Azure[2]can also be considered as a platform service providing an API and enabling developers to run their application in the Microsoft Azure environment.

Development of a cloud platform application is somewhat analogous to creating a software application for the old-fashioned web servers, in the sense that developers write and deploy codes to a remote server. The result is a web-based application for end-users. However, the PaaS model is different in that it can include additional services, such as scalability, monitoring, and load balancing, to simplify application development, deployment, and execution. The developers may also incorporate other services offered by the PaaS into their application, such as authentication services, e-mail services, and user interface components.

[1]Google App Engine. http://code.google.com/appengine
[2]Microsoft Azure. http://www.microsoft.com/azure

The cloud software that is designed for the cloud platform, in turn, usually has a shorter time to market. Some research initiatives have also emerged to promote a more detailed understanding of PaaS, such as AppScale[8]. Another aspect that characterizes PaaS services is the provision of information metering and billing APIs. Metering and billing help developers to more readily establish a business model focused on usage through their application. Such support helps to incorporate and maintain relationships between end-users, developers, PaaS, and any lower-level providers, while at the same time allowing developers and providers to gain economic benefit.

2.5 Software as a Service

The first layer is the cloud application layer which resides on top of the cloud stack. The cloud application layer is the layer that is most accessible and visible to cloud end-users. Users typically access the services that this layer provides via the browser through web portals, and are often expected to pay fees for their use. This model has recently proved attractive to many users, as it alleviates the burden of software maintenance for customers and simplifies development and testing for providers[6]. Arguably the cloud application layer has allowed a new class of end-user devices to develop in the form of "netbook" computers, which are less costly end-user devices that rely on network connectivity and functionality cloud applications. Netbook computers also have limited capacity for processing, with little to no disc drive-based storage, depending on cloud services to meet both needs.

As for cloud services providers, this model simplifies their working to code updating and testing while protects their intellectual property. Since a cloud framework is installed on the computing infrastructure of the provider (instead of on the desktop computers of the customers), the application developers can roll smaller patches into the system and add new functionality without disrupting customers with requests for changes or service packs. In this model, the application development and testing are potentially less difficult, as the implementation area, i.e. the data center of the

provider, is limited. Also about profit margin of the company, this model provides a constant sales flow for the software provider which may be much more efficient in the long run. This SaaS model holds some beneficial advantages for cloud service users and provider [9].

2.6 Relationship between Cloud Computing Actors

Figure 4 provides an overview of the reference architecture for NIST cloud computing, which describes the main actors, their operations, and cloud computing functions. The diagram represents a high-level generic architecture and is intended to promote the comprehension of cloud computing specifications, uses, characteristics, and standards [10].

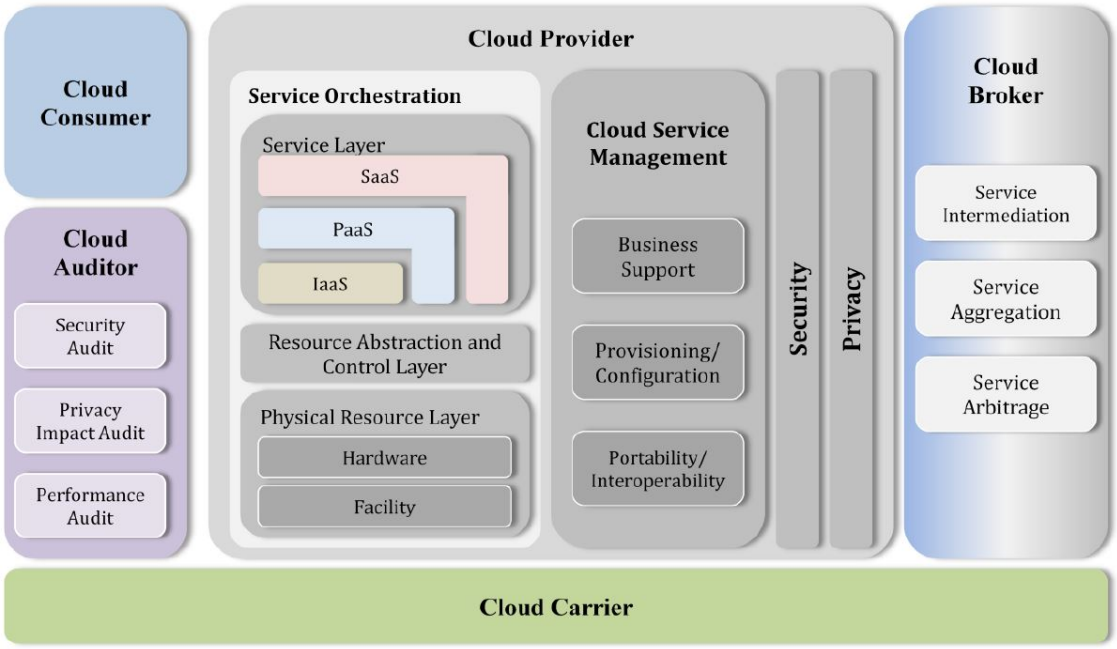

Figure 4: The Conceptual Reference Model [10]

As shown in Figure 1, five major actors are identified by the NIST cloud computing reference architecture: cloud consumer, cloud provider, cloud carrier, cloud auditor, and cloud broker. Each actor is an entity that participates in a transaction or process and/or performs cloud computing tasks (a person or an organization).

The actors that are defined in the NIST reference architecture are summarized in the following.

Actor	Definition
Cloud Consumer	A person or organization that maintains a business relationship with, and uses service from, Cloud Providers.
Cloud Provider	A person, organization, or entity responsible for making a service available to interested parties.
Cloud Auditor	A party that can conduct independent assessment of cloud services, information system operations, performance and security of the cloud implementation
Cloud Broker	An entity that manages the use, performance and delivery of cloud services, and negotiates relationships between Cloud Providers and Cloud Consumers
Cloud Carrier	An intermediary that provides connectivity and transport of cloud services from Cloud Providers to Cloud Consumers

Table 1: Actors in Cloud Computing [10]

The relationships between the actors are shown in Figure 5. A cloud user can request cloud services directly or through a cloud broker from a cloud provider. A cloud auditor carries out independent audits and can contact others to collect the information needed.

- **Example Usage Scenario 1** Instead of contacting a cloud provider directly, a cloud customer can request service from a cloud broker. By combining multiple services or by enhancing an existing service, the cloud broker can create a new service. The real cloud providers are invisible to the cloud user in this case, and the cloud customer communicates directly with the cloud broker.[10]

- **Example Usage Scenario 2** Cloud carriers offer cloud services from cloud providers to cloud users for networking and transportation. A cloud provider participates in and arranges two unique service level agreements (SLAs), one with a cloud carrier (e.g. SLA2) and one with a cloud customer (e.g. SLA1), as shown in Figure 7.

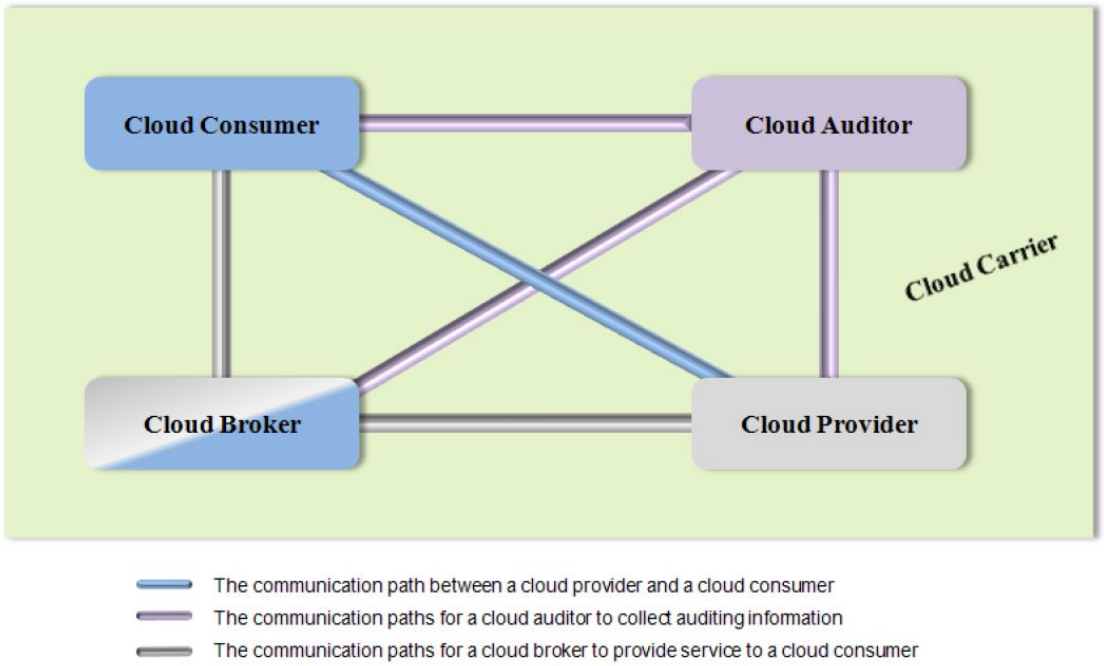

Figure 5: Interactions between the Actors in Cloud Computing [10]

Figure 6: Usage Scenario for Cloud Brokers [10]

A cloud provider arranges Service Level Agreements (SLAs) with a cloud carrier and which request dedicated and encrypted connections to ensure that cloud services are accessed at a consistent level in compliance with cloud customer contractual obligations.

In this case, the provider can specify its capacity, flexibility, and functionality requirements for SLA2 to provide the critical SLA1 requirements.

- **Example Usage Scenario 3** For a cloud service, a cloud auditor performs independent audits of the execution of the cloud service's operation and security. Interactions with both the Cloud Customer and the Cloud Provider could be

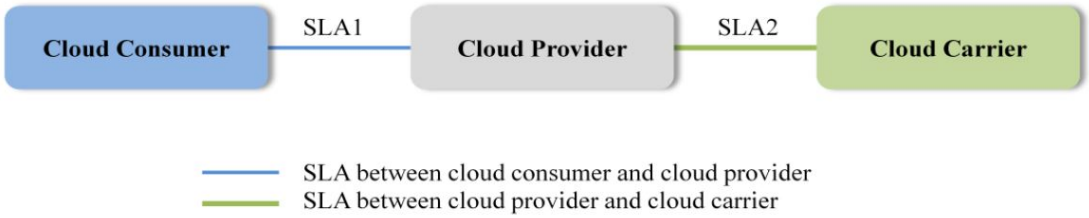

Figure 7: Usage Scenario for Cloud Carriers [10]

included in the audit [10].

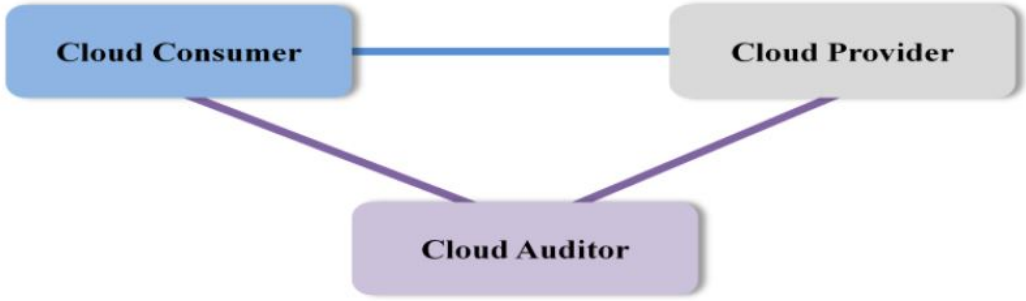

Figure 8: Usage Scenario for Cloud Auditors [10]

2.7 Cloud Consumer

The main stakeholder for the cloud computing service is the cloud customer. A cloud customer is an individual or company that maintains a contractual relationship with a cloud provider and uses the service. A cloud customer browses a cloud provider's service catalog, requests the required service, creates service contracts with the cloud provider, and uses the service. The cloud consumer will be billed for the service provided and payments must be structured accordingly. To define the technical performance requirements fulfilled by a cloud provider, cloud consumers need SLAs. SLAs may cover terms relating to service quality, protection, performance failure remedies.

A cloud provider can also list a set of commitments not expressly made to customers in the SLAs, i.e. restrictions and responsibilities that must be agreed upon by cloud consumers. A cloud user can freely select a better-priced cloud provider with more favorable terms and conditions. The pricing policy and SLAs of a cloud provider

are typically non-negotiable unless the customer plans heavy use and may negotiate better contracts.

Activities and consumption scenarios can be different across cloud users, depending on the services requested. Some instances of cloud services available to a cloud customer are presented in Figure 9 [10].

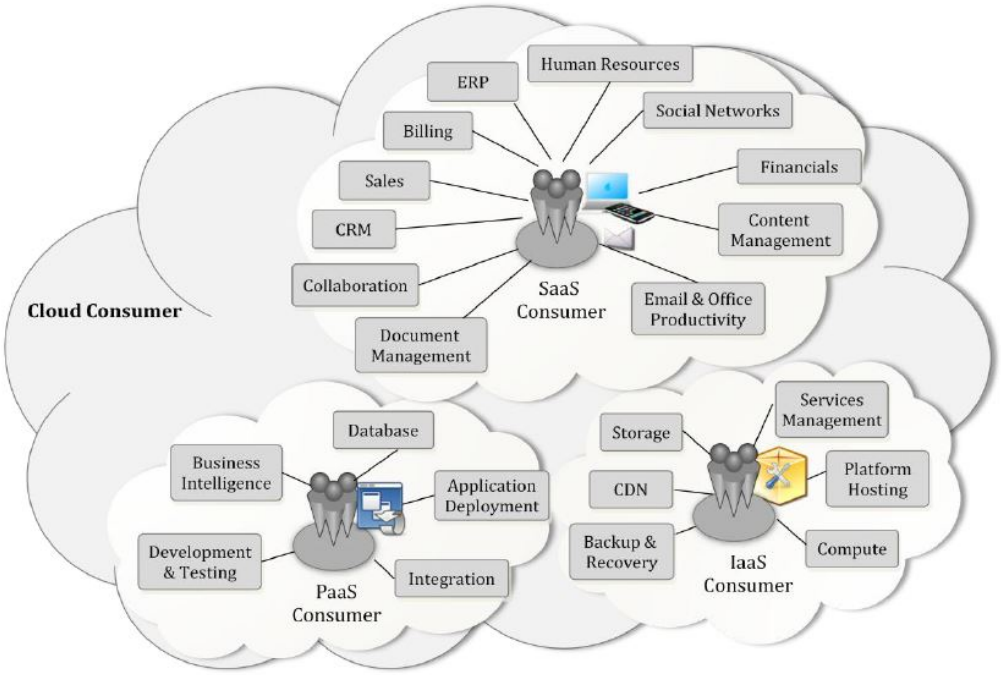

Figure 9: Usage Scenario for Cloud Auditors [10]

- **Consumers of SAAS:** SaaS applications in the cloud and made available to SaaS customers through a network. SaaS customers may be organizations that have access to software applications for their members, end-users who directly use software applications, or software application administrators who configure end-user applications. SaaS customers may be billed Based on the number of end-users, the time of usage, the network bandwidth used, the amount of data stored, or the length of data stored.

- **Consumers of PAAS:** The tools and execution services offered by cloud

providers to build, test, deploy and manage applications hosted in a cloud environment can be used by PaaS cloud consumers. Application developers developing and implementing application software, Pass consumers can be application testers running and evaluating applications in cloud-based environments, application deployers publishing applications in the cloud, and application managers configuring and tracking the performance of applications on a network. Depending on the processing, database storage, and network resources consumed by the PaaS application, and the length of platform use, PaaS consumers can be paid.

- **Consumers of IaaS:** IaaS users have access to virtual machines, network-accessible storage, components of the network infrastructure, and other essential computing tools on which they can install and run arbitrary applications. System developers, system administrators, and IT managers who are involved in the creation, implementation, management, and monitoring of services for IT infrastructure operations may be IaaS consumers. Consumers of IaaS are provided with the ability to access these computing resources and are paid according to the amount or length of the resources consumed, such as CPU hours used by virtual machines, data storage volume, and duration, bandwidth used by the network, number of IP addresses used for certain intervals [10].

2.8 Cloud Provider

A cloud provider is an individual, an organization; it is the entity responsible for supplying interested parties with a service. A Cloud Provider acquires and maintains the computing infrastructure needed to deliver the services, operates the services' cloud applications, and arranges for Cloud Users to deliver the cloud services through network access.

- **For Software as a Service:** The cloud provider deploys, configures, manages,

and upgrades the operation of software applications on cloud infrastructure for Software as a Service so that the services are supplied to cloud customers at the expected service levels. In managing and controlling applications and infrastructure, the SaaS provider bears most of the obligations, while cloud consumers have minimal administrative control of the applications.

- **For Platform as a Service:** For PaaS, the Cloud Provider manages the platform computing infrastructure and operates the cloud software that provides the platform components, such as the execution stack of runtime software, databases, and other middleware components. Usually, the PaaS Cloud Provider also supports the PaaS Cloud Consumer's growth, deployment ,and management process by offering tools such as integrated development environments (IDEs), cloud product development versions, software development kits (SDKs), deployment and management tools. The PaaS Cloud User has power over the applications and likely some hosting environment settings, but has no or restricted access to the network, servers, operating systems (OS), or storage infrastructure underlying the platform.

- **For Infrastructure as a Service:** The Cloud Provider obtains the physical computing services that underlie the service for IaaS, including servers, networks, storage, and infrastructure hosting. Via a series of service interfaces and computer resource abstractions, such as virtual machines and virtual network interfaces, the Cloud Provider runs the cloud software required to make computing services accessible to the IaaS Cloud User. In exchange, the IaaS Cloud Consumer uses these computing tools, such as a virtual machine, for their basic computing needs. In contrast with SaaS and PaaS Cloud Users, the IaaS Cloud Consumer has access to more fundamental types of computing resources and thus, the more software components in an application stack, like the OS and network, have more power.

On the other hand, the IaaS Cloud Provider has control over the physical hardware and cloud software that facilitates the provisioning of these infrastructure resources, such as physical servers, network equipment, storage devices, host operating systems and virtualization hypervisors.

As shown in Figure 10, The activities of a cloud provider can be represented in five key areas. A cloud provider performs its activities in the fields of service deployment, service orchestration, cloud service management, security, and privacy [10].

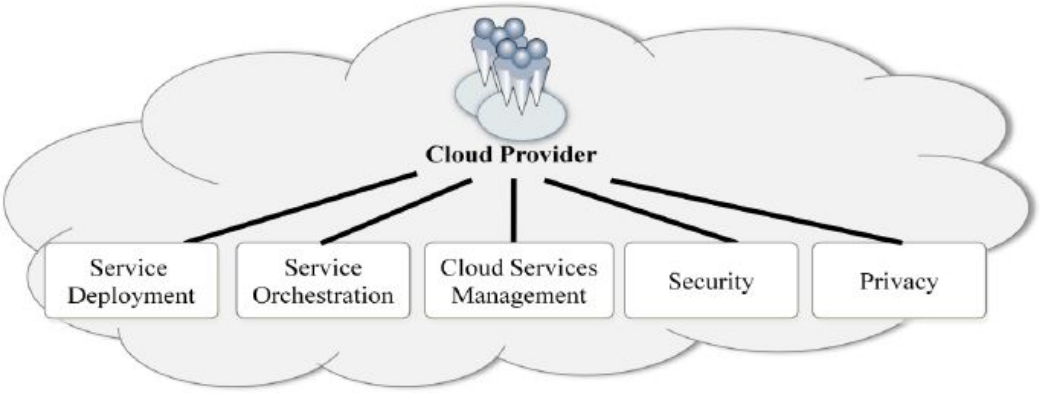

Figure 10: Cloud Provider - Major Activities [10]

2.9 Cloud Auditor

A cloud auditor is a party that can conduct an independent analysis of controls on cloud services to express an opinion about them. Audits are performed to verify conformance to standards through a review of objective evidence. The services offered by a cloud provider can be evaluated by a cloud auditor in terms of security controls, privacy effects, performance, etc.

2.10 Cloud Broker

When cloud infrastructure progresses, it can be too difficult for cloud users to handle the convergence of cloud services. Instead of directly contacting a cloud provider, a

cloud user can request cloud services from a cloud broker. A cloud broker is an agency that regulates the use, quality, and delivery of cloud services and negotiates relationships between cloud providers and cloud customers. A cloud broker can typically provide services in three categories:

- **Service Intermediation:** By enhancing certain basic features and delivering value-added services to cloud customers, a cloud broker improves a provided service. The improvement can be managing access to cloud services, identity management, performance reporting, enhanced security, etc.

- **Service Aggregation:** Multiple services are merged and incorporated into one or more new services by a cloud broker. The broker offers data integration and guarantees the safe movement of data between the cloud customer and several cloud providers.

- **Service Arbitrage:** Service arbitrage, except that the services being aggregated are not set, is similar to service aggregation. Arbitrage of services means that a broker can pick services from various agencies. For example, a cloud broker may use a credit rating service to calculate and pick an agency with the best score.

2.11 Cloud Carrier

A cloud carrier serves as an intermediary that offers cloud services between cloud users and cloud providers with connectivity and transport. Via networks, telecommunications, and other connected devices, cloud carriers provide customers with access. Cloud users, for instance, may obtain cloud services, through network access devices, such as computers, laptops, mobile phones, mobile Internet devices (MIDs), etc [11] Network and telecommunication carriers or transport agents usually provide the delivery of cloud services [12]. If a transport agent refers to a business entity that offers storage media such as high-capacity hard drives to be physically transported.

Note that with a cloud carrier, a cloud provider can set up SLAs to provide services consistent with the level of SLAs provided to cloud customers, and may enable the cloud carrier to provide dedicated and safe links between cloud customers and cloud providers.

2.12 Scope of Control between Provider and Consumer

In a cloud environment, the Cloud Provider and Cloud Customer share resource control. As Figure 11 shows, Different service models influence the control of computing resources by an enterprise and therefore what can be achieved in a cloud environment. Using a classic software stack notation comprised of the program, middleware, and OS layers, the figure illustrates these distinctions. This study of the delineation of controls over the application stack helps to understand the roles of the cloud application management parties involved.

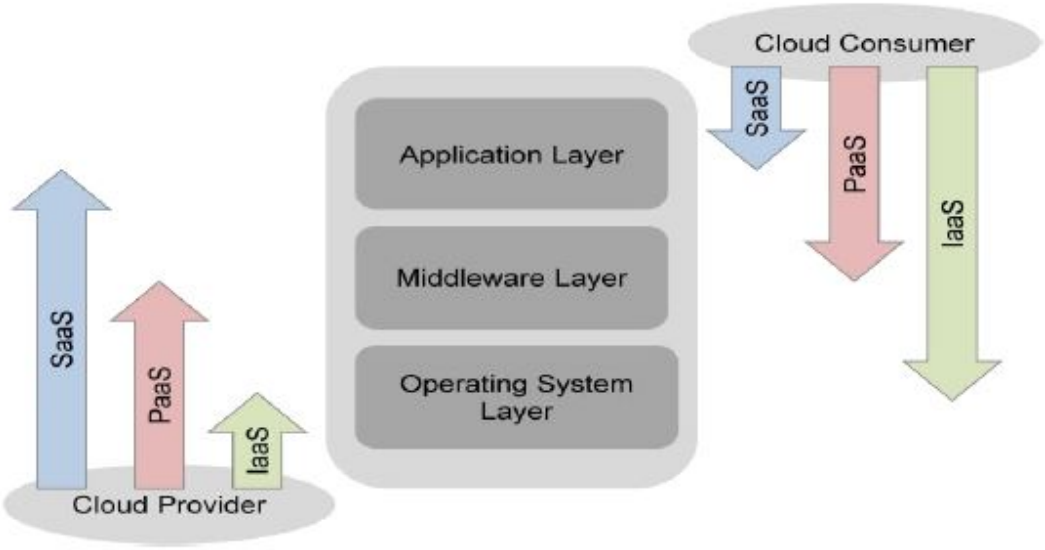

Figure 11: Scope of Controls between Provider and Consumer [10]

- **Application Layer:** Computer applications aimed at end-users or programs are included in the application layer. SaaS customers use the software, or PaaS customers, IaaS customers, and SaaS providers install/manage/maintain them.

- **Middleware layer:** The middleware layer includes software building blocks for the creation of application software in the cloud (e.g. libraries, databases, and Java virtual machines). PaaS customers, installed/managed/maintained by IaaS customers or PaaS providers, and hidden from SaaS customers, use the middleware.

- **Operating system Layer:** The operating system and drivers are included in the OS layer and are shielded from SaaS consumers and PaaS consumers. An IaaS cloud enables one or more virtualized guest OSs to operate on a single physical host. In general, customers have wide freedom to choose which OS to host from all the operating systems that the cloud provider can support. IaaS customers can assume full responsibility for the guest OS, while the host OS is managed by the IaaS provider.

2.13 Deployment Models

There are four types of cloud deployments: (i) Private cloud, (ii) Public cloud, (iii) Hybrid cloud, and (iv) Community cloud.

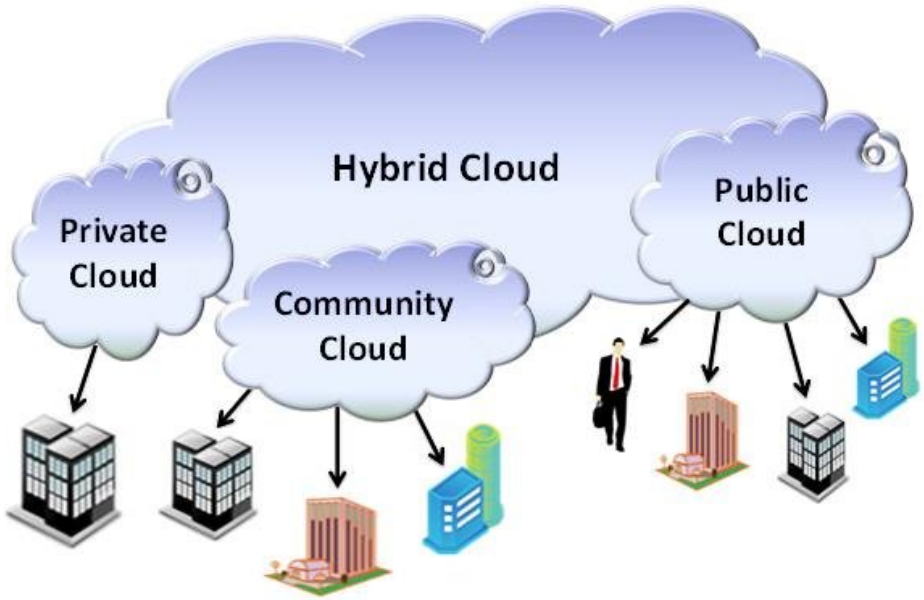

Figure 12: Cloud Computing Deployment Model[3]

2.14 Private Cloud

The private cloud is physically situated at the on-site datacenter of your enterprise or may be hosted by a third-party service provider. But in a private cloud, infrastructure is still managed on a private network and the hardware and software are dedicated only to your organization. Private clouds have the following advantages:

- **More flexibility:** Which means that organizations can customize their cloud environment based on specific needs.
- **High scalability:** Private clouds still offer the scalability and efficiency of a public cloud.
- **Improved security:** Resources are not shared with anyone so it is possible to reach higher standards of control and security.

[3]https://www.webbazar.co.in

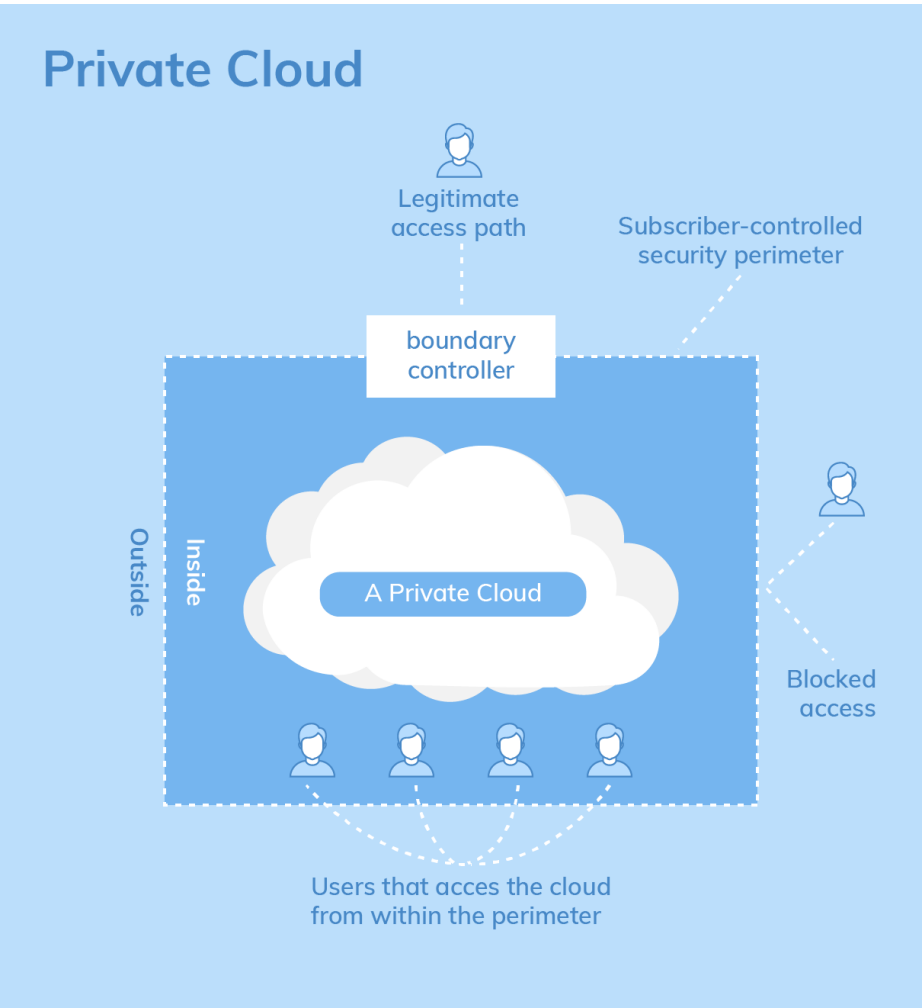

Figure 13: Private Cloud Model[4]

2.15 Public Cloud

The most popular method of deploying cloud computing is using public clouds. A third-party cloud service provider manages and maintains the cloud services (such as servers and storage) and is distributed over the Internet. An example of a public cloud is Microsoft Azure. For a public cloud, the cloud provider owns and maintains all hardware, software, and infrastructure. You share the same infrastructure, storage, and network equipment with other companies or cloud-based "tenants" in the public cloud. You access services and manage your account through a web browser. Public

[4]www.sam-solutions.com

clouds have the following advantages:

- **Lower Cost:** There's no need to buy hardware or software and you just pay for the service you use.
- **Unlimited Scalability:** To satisfy your business needs, on-demand resources are available.
- **No Maintenance:** Service Providers are responsible for providing maintenance.

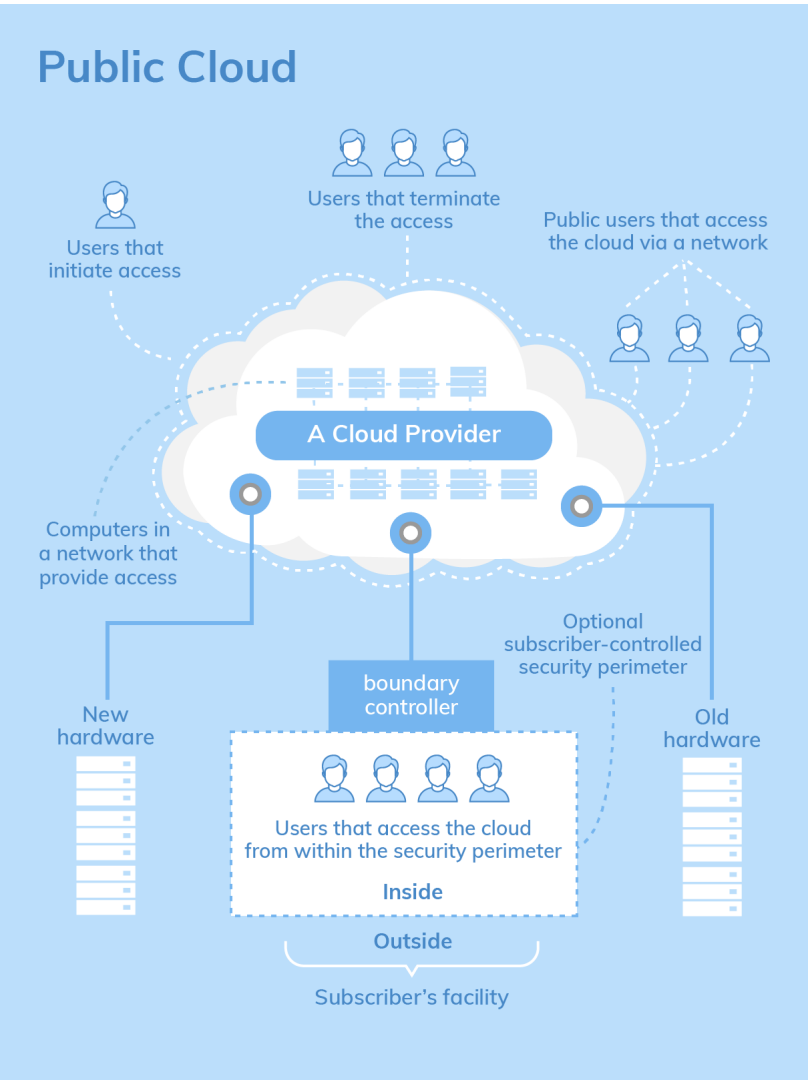

Figure 14: Public Cloud Model[5]

[5] www.sam-solutions.com

2.16 Hybrid Cloud

Hybrid clouds are sometimes referred to as "the best of both worlds". In other words, hybrid clouds combine private clouds, with public clouds so organizations can take advantages of both. Hybrid Cloud offers more flexibility and deployment options by allowing data and applications to move between private and public clouds. You can use either private and public cloud-based on your needs. For instance, leverage public cloud for high-volume, lower security applications such as web-based email, and for sensitive, business-critical operations like financial reporting use private cloud approach. Hybrid clouds have the following advantages:

- **Flexibility:** Which means you can take advantage of the public cloud when you need more resources.
- **Cost-effectiveness:** You can scale your resources only when you need and pay for extra computing power only.
- **Control:** Your organization can maintain and monitor private infrastructure for sensitive applications.

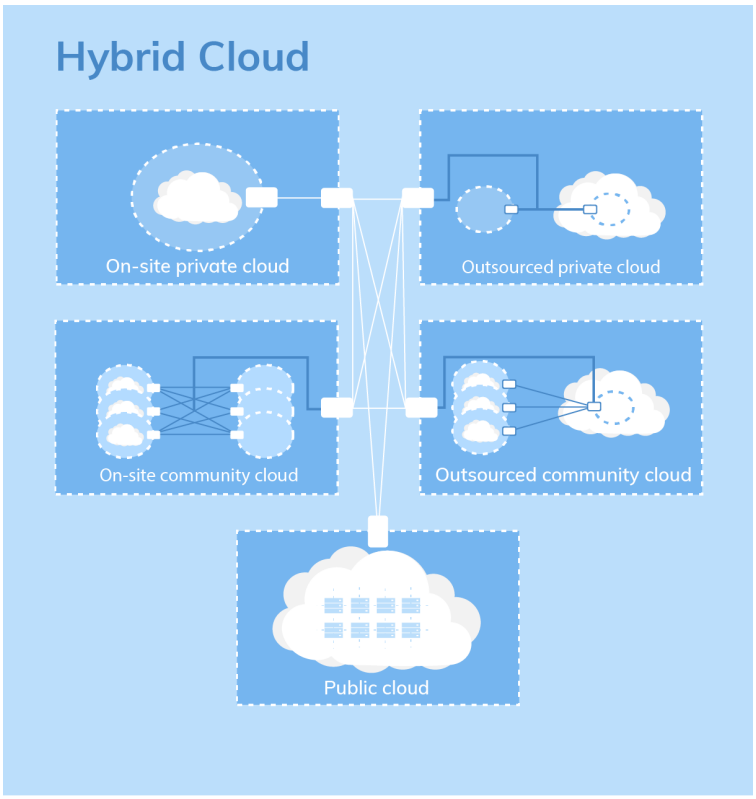

Figure 15: Hybrid Cloud Model[6]

2.17 Community Cloud

Community clouds are indeed distributed systems that are made up of different cloud services to address specific requirements of communities, or industries. The users of a particular community cloud fall into a well-identified community, having the same interests or needs; they can be government bodies, industries, or even simple users, but all of them concentrate on the same issues for their interaction with the cloud. This is different from public clouds, in which multiple users with different services are covered. Also, Community clouds differs thoroughly from private clouds, in which services are basically provided for an organization that owns the cloud. To summarize, Community clouds provide the following advantages:

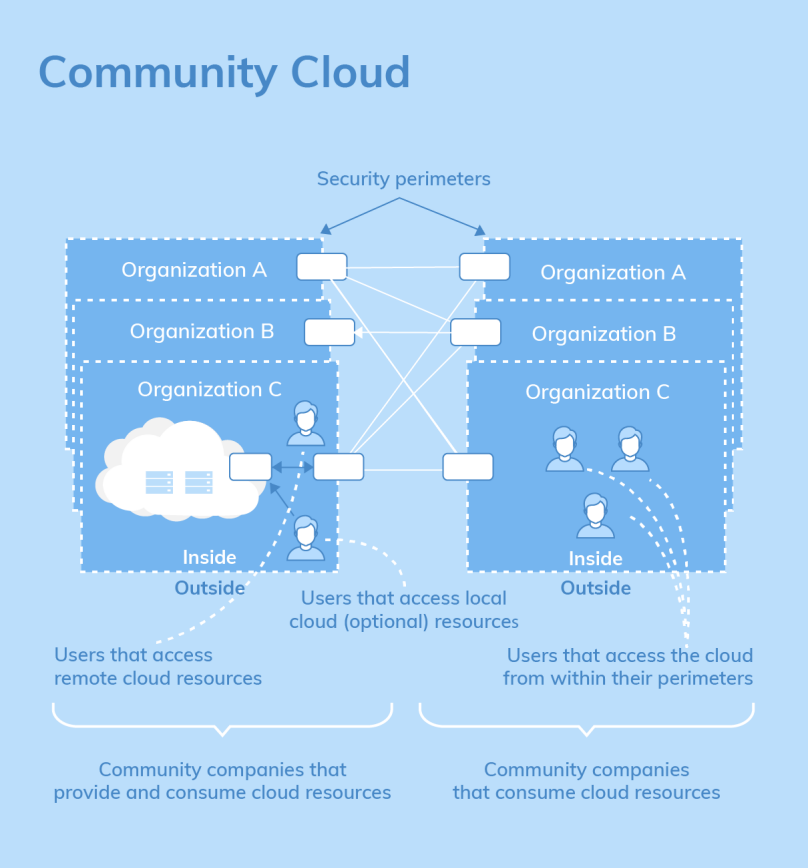

Figure 16: Community Cloud Model[7]

- **Graceful Failures:** There is no single point of failure since there is no single service provider in control of infrastructure.
- **Convenience and control:** There is no conflict between convenience and control within a community cloud since the cloud is shared and controlled by the community that makes all the decisions through a collective democratic process
- **Community:** Being based on a collective that offers resources and services, the infrastructure is more scalable because the system can grow simply by expanding its user base.

[7] www.sam-solutions.com

The Comparison of Top Cloud Deployment Models

Top cloud deployment models can be compared as shown in the following table:

	Public	Private	Community	Hybrid
Ease of setup and use	Easy	Requires IT proficiency	Requires IT proficiency	Requires IT proficiency
Data security and privacy	Low	High	Comparatively high	High
Data control	Little to none	High	Comparatively high	Comparatively high
Reliability	Low	High	Comparatively high	High
Scalability and flexibility	High	High	Fixed capacity	High
Cost-effectiveness	The cheapest	Cost-intensive,the most expensive model	Cost is shared among community members	Cheaper than a private model but more costly than a public one

Table 2: Comparison between Cloud Deployment Models [5]

2.18 Cloud Computing Use Cases

Cloud computing technology is increasingly important in the industry since it has brought several solutions that make doing business much easier. It has increased performance, dynamic resource allocation capabilities and provides a convincing opportunity for organizations to outsource their IT infrastructure under the pay-per-use model provided by many public cloud providers. here are several potential use cases for the cloud that businesses of every size and industry need to consider. We have mentioned some use cases of cloud computing below that could help any company.

2.19 Disaster Recovery as a Service (DRaaS)

Disaster recovery offers to secure and protect IT resources, ensuring high availability, and having a plan to return to operations which is critical to every business today. DR

created to prevent businesses from failure in the event of a disaster. You can easily provide a backup on the cloud to protect your data, in case your on-premise servers fail. In this situation, cloud can fully build and manage your required infrastructure if a fail-over happens. Therefore, disaster recovery offers some beneficial features such as, flexibility for data recall, Secure data transfer and storage, Ease of deployment, and Rapid recovery [8].

2.20 Scaling resources

Every business will expand at some point, they need to increase or capably reallocate your resources. Cloud computing offers scaling resources which makes it easy for users to scale their resources up or down depending on business needs(season, projects, growth, and more). By implementing cloud scalability, you can enable resources to grow as your traffic or organization grows.There are two cloud scaling strategies: vertically or horizontally. When you scale vertically it means scaling up or down, and when you scale horizontally, it means scaling out or in [9].

Cloud Vertical Scaling

It refers to adding to an existing server more CPU, memory, I/O resources, or replacing one server with a more strong server. Vertical scaling of Amazon Web Services(AWS)and vertical scaling of Microsoft Azure can be done by adjusting instance sizes or by buying a new,more efficient appliance in a data center and discarding the old one.

Cloud Horizontal Scaling

It refers to the provision of extra servers to meet your needs, often separating workloads between servers to reduce the number of requests received by any particular

[8]https://solutionsreview.com/
[9]https://rapidscale.net/

server. This would involve adding additional instances in a cloud-based setting instead of switching to a larger instance size.

2.21 Hosting applications and services

Application Hosting Services refers to providing a place for a company to host its applications and services for the public, instead of relying on physical distribution. This is a simple use case that most companies already use, but for any organization that provides software and service computing platforms that allow the distribution of software through the Internet, this software as a service (SaaS) use case is important to consider. Application Hosting Services can provide an operational platform for virtually any type of software application. Content management applications, web development applications, database applications, and email management applications serve as common examples of on-demand software that may be hosted via the Internet.

2.22 Big Data Analytics

Big data analytics use compute-intensive data mining algorithms that need efficient high-performance processors to produce timely results. Cloud computing infrastructures can serve as an efficient framework for addressing the needs of big data analytics applications for both computational and data storage. Advanced data mining techniques and related methods can help extract data from massive, complex datasets that are helpful in many business and scientific applications, including tax collection, retail sales, social studies, bio-sciences, and high-energy physics [13].

2.23 Cloud computing basic components

Cloud computing consists of several components. Here, we will address the basic components on which cloud computing has been deployed. These components consist of a broad variety of services that we can use all over the internet.

2.24 Virtualization

It plays an important role in the implementation of the cloud. It is the strategic aspect of the cloud, which enables multiple users to use physical resources. It produces the virtual instance of a resource or computer such as an operating system, servers, network resources, and storage devices where the resources are used in more than one execution environment in the framework [14].

2.25 Multi-tenancy

Multi-tenant environments may have multiple clients or users who may not see or exchange information with each other; but may share resources or applications in an execution environment, even though they do not belong to the same entity. Multi-tenancy results in the efficient use of hardware and data storage mechanisms [15].

2.26 Cloud storage

It is a component, which maintained, managed, and backed up remotely and it is made available over the network where the users can access data [16].

2.27 The hypervisor

It enables a single hardware host to run several virtual machines(VMs). It manages and tracks the different operating systems that run on a shared physical device [17].

2.28 Cloud Network

A typical data center has hundreds or thousands of servers. To efficiently build and manage the storages the cloud needs a secure network infrastructure which is called cloud networking. It requires an internet connection and similar to a virtual private network which allows the user to securely access printers, applications, files, etc [18].

2.29 Cloud Computing Challenges

Cloud Computing is a disruptive technology with profound implications for the delivery of Internet services as well as for the IT sector as a whole. However, several technical and business-related issues are still the main concerns.

- **Security and Privacy:**

  Security (e.g., data security and integrity, network security), privacy (e.g., data confidentiality), and service-level agreements are the main concerns in cloud computing. As resources are all distributed among different cloud regions, data privacy and security are more prone to get compromised, which makes companies worried more about data, especially when it comes to sensitive data.

2.30 Privacy

Because the Cloud computing system normally provides services (e.g. DaaS, SaaS, IPMaaS, PaaS, etc.) to its users on the other side of the Internet, the confidential information of individual users and companies is stored and handled By the service providers and thus contribute to privacy issues. In computer literature, privacy concerns have existed for a long time, and several laws have been published to protect the individual privacy of users as well as business secrets. These actions however, are obsolete and inapplicable to new situations in which a new partnership between consumers and providers (i.e. three parties) occurs. This section illustrates a few privacy acts which are not applicable in the new environment.

2.31 Legal Issues

Cloud computing platforms (including applications and services hosted on them) have important implications for personal information privacy as well as business

and government information confidentiality. That is because any information, including email, word processing papers, spreadsheets, videos, health records, photos, tax or other financial information, business plans, accounting information, advertising campaigns, address books, and more are moved from local computers to Cloud Storage systems in the Cloud Computing era. Besides, all of a user's content initially stored on a local computer can be migrated to a single cloud provider or even to multiple cloud providers. Whenever information is exchanged in the cloud by a person, a corporation, a government agency, or another organization, privacy or confidentiality issues can arise.

Besides, in cloud computing systems, the relationship between users and providers is more complex than that of other types of web services. Includes three roles: Cloud provider, XaaS provider/Cloud user, and XaaS user as illustrates in Figure 17, where X could be D (Data), S (Software), P (Platform), I (Infrastructure), and so on.

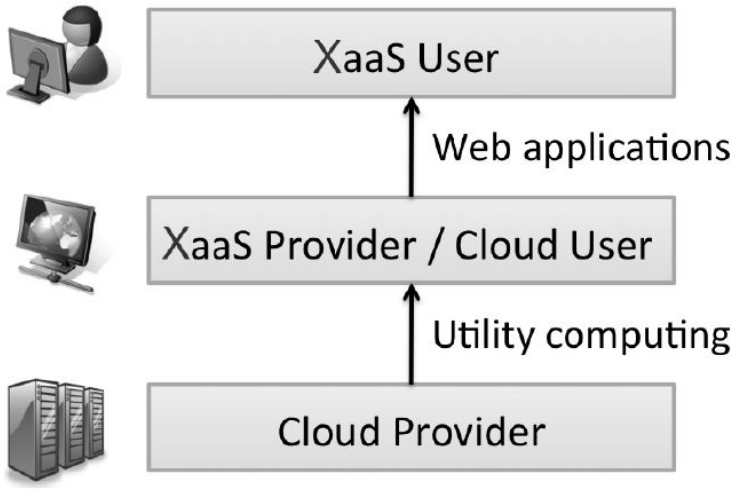

Figure 17: Users and Providers of Cloud Computing [19]

XaaS user is a customer or potential customer of a Cloud Computing service who can be an individual, business, government agency, or any entity. XaaS

provider is the organization that offers the Cloud Computing service and a user of Cloud Computing system. A Cloud provider is defined as an organization that provides the Cloud Computing system, it can be an individual, a company, or other business, a non-profit organization, a government agency or any other entity. it is worth noticing that a Cloud service provider is one type of the third party that maintains information about, or on behalf of, another entity.

In this new environment, several privacy acts are not applicable because of the presence of a third party i.e., cloud service provider/cloud user). Besides, certain privacy-related acts were released several years ago and originally covered privacy between only two parties. This means that data stored by a third party may have less or weaker privacy rights than data kept by the owner of the data. It could be simpler for other departments and organizations or even governments to collect information from a third party.

2.32 Multi-Location Issues

The primary job of the Cloud is to offer huge computer resources to users, including infrastructure, platforms, services (e.g., storage, computing power, and so on). A company must trust the provider of the Clouds system and store its private information in the Cloud system Which means that the company's data is stored in the machine of someone else. Consequently, if data are stored in someone else's devices, many things can go wrong. For example, the Cloud service provider may go out of business or may decide to hold the data hostage if there is a dispute. Moreover, large Cloud system vendors have their Cloud mirror sites in many other countries. For instance, in multi-locations, Amazon has its EC2, and currently one in The USA and the rest of Europe. Google App Engine is also based in several different zones, it has 36 data centers worldwide), such as the USA, China, and so on. A few problems which result in storing

private data in multi-locations are listed here:

- **Multi-location of the private data:** It is very risky if the organization stores its private information in the third Computer of the Party. In this way, the private data of the corporation resides on someone else's computer, and in someone else's facility. Many things, then will go wrong. Firstly, the provider of cloud services can go out of business. Secondly, if there is a conflict, the cloud service provider can decide to keep the data hostage. Third, an organization needs to understand which country It will be hosting its data. This is because the location of the information specifically influences the option of legislation to be applied to its private data. For example, if the data resides in China, access to that private data may be protected by Chinese law.
- **Multi-location of the service provider:** The Cloud service client (e.g. business user or private user) also needs to make sure how their declared services are handled by the Cloud service provider. The Cloud service client is thus able to maintain a direct link with the provider and to manage its own private data.
- **Data combination and commingling:** The Cloud Computing client (e.g., business user or private user) has to make sure that whether its private data is stored separately from others or not. If they are combined or commingled with those of other customer's data, then it is much more vulnerable. For instance, viruses might be transmitted from one client to others. If another client is hacked, the attack might affect the availability or integrity of the data of other companies located in the same environment.
- **Restrictions on techniques and logistics:** The Cloud service provider may find it very difficult or even impossible to ensure the locations where the data of the Cloud Storage client will be stored. For instance, Amazon

has data centers across the world, the data of the client is automatically placed across them unless Amazon uses dedicated servers for specific customers or clients. It may also be essential for the Cloud service provider to tackle logistics. Providers of cloud computing must delegate data storage or other services to third parties.

- **lack of standard APIs:**

  Standard APIs allow customers to easily extract code and data and transfer it from one side to the other [20].

- **Latency and backhaul overload:**

  As more customers tend to use services offered by cloud, huge amount of data should be transferred between the cloud and the endpoints which results in more bandwidth consumption and adds additional latency to the computations, which in turn causes congestion in the backhaul infrastructure.

Taking all deficiencies into account, especially with the advent of IoT devices, the cloud itself is not capable of connecting millions of *things* that are spread over large geographically distributed areas. Thus, alternative solutions should be provided.

3 Edge Computing

Edge computing is another paradigm that enhances the management, storage, and processing power for data that are generated by connected devices. edge computing is located at the edge of the network close to IoT devices, but not on the IoT devices. but as close as one hop to them. OpenEdge Computing describes edge computing as computing performed by small data centers that are close to users at the edge of the network. The initial vision for edge computing is to have open standards and ubiquitous ways of computing and storage resources close to the user.

In the current landscape of IoT devices, edge computing is a crucial computing paradigm; it integrates IoT devices with the cloud by intelligently filtering, preprocessing, and aggregating IoT data through cloud services deployed near IoT devices. Privacy, latency, and networking are several problems that edge computing is well suited to tackle. Because of its user proximity, edge computing latency is usually lower than in MCC and cloud computing if adequate local computing power is provided; edge computing latency may be slower than cloud or MCC if the local computing unit is not powerful enough. In edge computing, service availability is also better because connected devices do not have to wait for a highly centralized network to deliver a service, nor are connected devices restricted by traditional mobile computing resources. Edge computing has limited data centers as opposed to MACC, whereas MACC does not fundamentally require data centres. As a consequence, there is higher service availability for edge computing.

Although fog computing and edge computing paradigms both emphasize on pushing computation and storage to the edge of the network and to the proximity of end-devices, they are not pointing to the same concept. The OpenFog Consortium states that edge computing is often mistakenly called fog computing; Fog computing is hierarchical and offers computation, networking, and storage anywhere across the cloud-to-things continuum; while edge computing paradigm is limited to computing at the edge.

The two-way computing streams in edge computing are shown in figure 18. In the concept of edge computing, things not only consume data but also generate data by taking part in the processing. Apart from requesting services and content, Edge devices will perform computational tasks from the cloud. An edge node can perform data storage, computation of the load, processing, and caching. Also, the edge system can distribute requests and provide users with service on behalf of the cloud. Edge devices must be well configured to satisfy privacy criteria, reliability measures, and security issues in such scenarios [21].

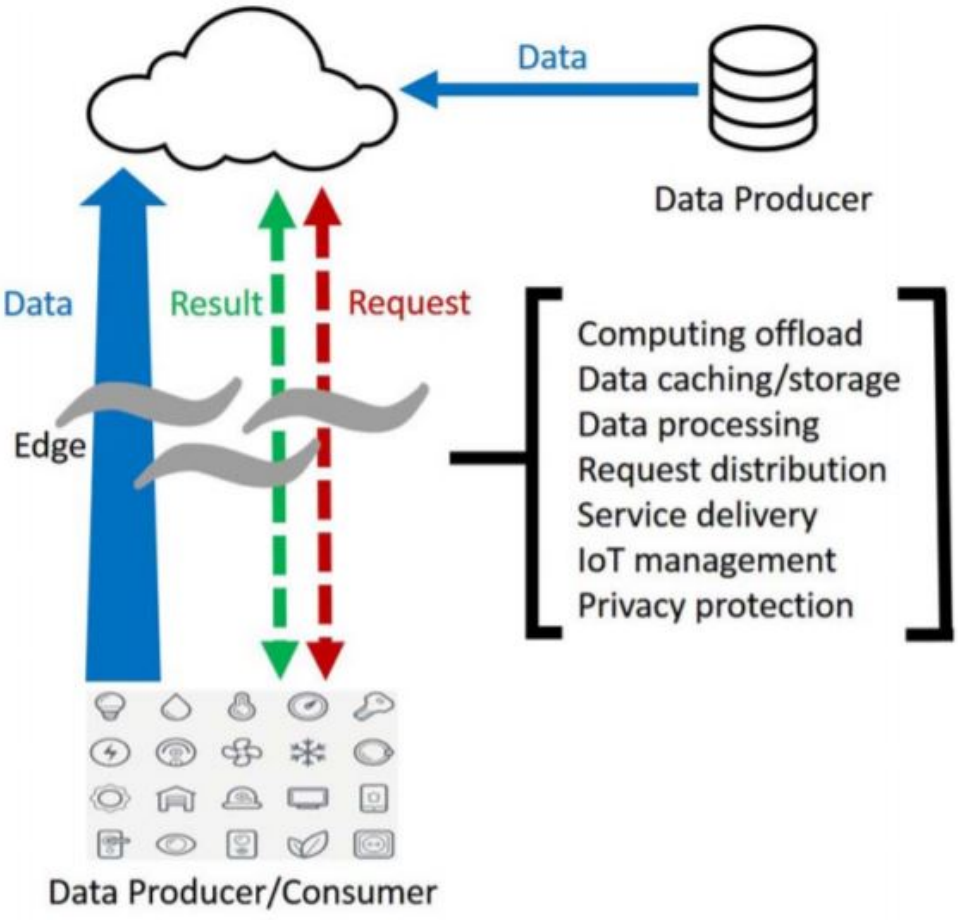

Figure 18: Edge computing paradigm [22]

Where is Edge?

It should be noted that in certain articles, edge computing, cloudlets, fog computing, and mist computing are used similarly, as they all have "edge" as a common word. The term edge used by the telecommunications industry usually refers to 4G/5G base stations, RANs, and ISP (Internet Service Provider) access/edge networks. However, the term edge that has recently been used in the IoT landscape refers to the local network where IoT devices and sensors reside.

In other words, the edge is the immediate first hop, such as WiFi access points or gateways, from the IoT devices (not the IoT nodes themselves). This computational paradigm is known as mist computing if the computation is performed on IoT devices

themselves. General Electric states that fog computing focuses on edge device interactions (e.g., RANs, base stations, or edge routers), while edge computing focuses on the wired device technology (e.g., WiFi access points).

3.1 Why Do We Need Edge Computing

- **Push From Cloud Services:** As the computational capacity on the cloud outclasses the strength of the things at the edge, placing all the computing activities on the cloud is an effective form of data processing. However the bandwidth of the network has come to a standstill compared to the increasingly developing processing speed of data. With the rising amount of information produced at the edge, data transmission speed is becoming the bottleneck for the cloud-based computing paradigm. For example, a Boeing 787 can produce about 5 gigabytes of data per second, but the bandwidth between the aircraft and the ground-based satellite or base station is not wide enough to transmit data. As another example consider an autonomous vehicle. The car produces one gigabyte of data every second and needs real-time processing for the vehicle to make the right decisions. If all of the information needs to be transmitted for processing to the cloud, the response time will be too long. Not to mention that its capacity to accommodate a large number of vehicles in one region would challenge existing network bandwidth and reliability. In this case, for shorter response time, more effective processing, and lower network burden, the data needs to be processed at the edge.

- **Pull From IoT:** Nearly all forms of electrical devices will become part of the IoT and will play the role of both data manufacturers and users, such as sensors for air quality, LED bars, streetlights, and even an Internet-connected microwave oven. It is fair to infer that in a few years the number of items at the edge of the network will grow to more than billions. This means that most

[9]https://www.ge.com/ digital/blog/what-edge-computing

of the IoT-generated information will never be transmitted to the cloud, but will be consumed at the edge of the network.

- **Change From Data Consumer to Producer:** In the paradigm of cloud computing, the end devices at the edge typically play as a consumer of data, for instance, Watching a YouTube video on your smartphone, people are still creating data from their mobile devices today. More function placement at the edge is necessary for the shift from data user to data producer/consumer. For instance, it is very common for people to take pictures or record videos today and then share the data through a cloud service such as YouTube, Facebook, Twitter, or Instagram. Also, YouTube users upload 72 hours of new video content every single minute; Facebook users share almost 2.5 million pieces of content; Twitter users tweet almost 300 000 times; Instagram users post nearly 220 000 new photos. The picture or video clip may be very big and it would take up a lot of bandwidth for uploading. In this case, before uploading to the cloud, the video clip should be demoted and changed to the required resolution at the edge.

3.2 Edge Computing Applications

Edge computing has several potential applications which are listed below.

3.3 Cloud Offloading

Most computations on the cloud computing paradigm occur in the cloud side, meaning that data and requests are processed in the centralized cloud. However, Such a computing model can suffer from longer latencies which weakens the user experience. Edge computing has some computing capabilities, and this offers an ability to discharge part of the workload from the cloud. In the conventional content delivery network, only the data on the edge servers is cached.

This is focused on the fact that data is generated on the Internet by the service

provider, which has been valid for the past decades. In the IoT, the data is generated and consumed at the edge. In the edge computing model, therefore, not only information but also data-applied operations should be cached at the edge. Online shopping services are one possible application that might benefit from edge computing. A customer can also manipulate the shopping cart. By default, all these changes will be made in the cloud on his or her shopping cart, and then the new shopping cart display will be updated on the computer of the customer.

Depending on network speed and the load level of servers, this phase can take a long time. Due to the comparatively low capacity of a mobile network, it maybe even longer for mobile devices. As mobile device shopping is becoming more and more common, it is important to enhance the user experience, particularly concerning latency. If the shopping cart update is unloaded from the cloud in such a situation, The latency would be significantly reduced for servers with edge nodes. As mentioned, the shopping cart details and related operations of the users (e.g., add an object, change an item, remove an item) can both be cached at the edge node. Upon the user request reaching the edge node, the new shopping cart view can be created immediately. At the edge node, the data should be synchronized. However, this can be performed in the background and should be coordinated with the cloud.

3.4 Smart Home

IoT and edge computing have helped the home environment in recent years. Several devices, such as smart lamps, smart TV and robot vacuum, have been produced and are available on the market. For smart home, however, simply attaching a Wi-Fi module to the current electrical unit and linking it to the cloud is not enough. In addition to the connected device, cheap wireless sensors and controllers should be deployed to the room, pipe, and even floor and wall in a smart home environment. These items will report an impressive amount of data and this data should be mainly consumed in the home for the consideration of data transportation pressure and

privacy security. This role makes the paradigm of cloud computing unsuitable for a smart home [22].

Edge computing, however, is considered suitable for creating a smart home: with an edge gateway running a especialized edge operating system (edgeOS) in the home, stuff can be easily linked and handled in the home, data can be processed locally to release the Internet bandwidth burden, and the service can also be installed on the edgeOS for better management, and distribution.

Figure 19 illustrates the structure in the smart home world of a design of edgeOS. Via different networking methods, such as Wi-Fi, BlueTooth, ZigBee, or a cellular network, EdgeOS needs to collect data from mobile devices and all sorts of things. In the data abstraction layer, data from various sources need to be fused and massaged. The service management layer is on top of the data abstraction layer. This layer will be assisted by criteria like separation, extensibility, isolation, and reliability.

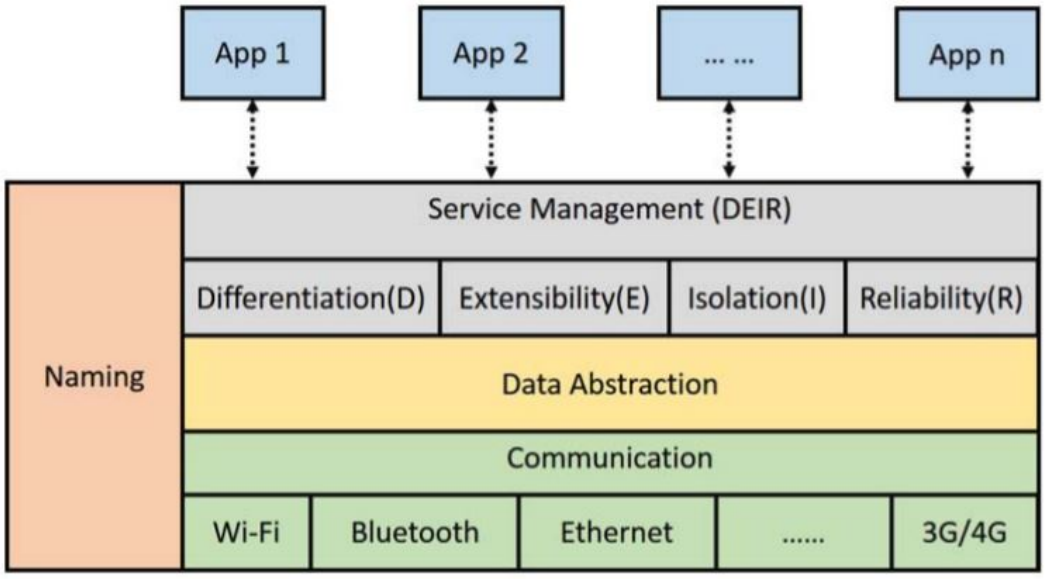

Figure 19: Structure of edgeOS in the smart home environment [22]

3.5 Smart City

From a single home to a neighborhood, or even a city scale, the edge computing model can be flexibly extended. Edge computing argues that computing can take place as close to the source of data as possible. A request could be created from the top of the computing paradigm with this design and be handled at the edge. Considering the following features, Edge computing may be a perfect tool for smart cities [22].

- **Large Data Quantity:** It is impractical to create centralized cloud data centres to control all of the data because the traffic burden will be too high. In this case, by processing the data at the edge of the network, edge computing may be an effective solution.

- **Low Latency:** Edge computing is also an effective paradigm for applications that require predictable and low latency, such as health emergencies or public safety, as it can save data processing time and simplify the network Architecture. Decision and diagnosis may be taken as well as spread from the edge of the network, which is more effective than central cloud data collection and decision-making.

- **Location Awareness:** Edge computing exceeds cloud computing because of location recognition for geographic-based applications such as transportation and utility management. Data could be gathered and analyzed based on geographic position in edge computing without being transported to the cloud.

3.6 Collaborative Edge

Arguably, the cloud has become the de facto computing network for academia and industry to process big data. A primary promise behind cloud computing is that the data should already be kept or transmitted to the cloud and finally processed in the cloud. However, due to privacy issues and the enormous expense of data transmission, data held by stakeholders is hardly exchanged with each other. Therefore, The

probability of cooperation between multiple stakeholders is reduced. The Edge may also be part of the conceptual definition as a physical small data centre that links cloud and end-users with data processing capabilities.

Connected health is one of the exciting applications shortly, as shown in Figure 20.

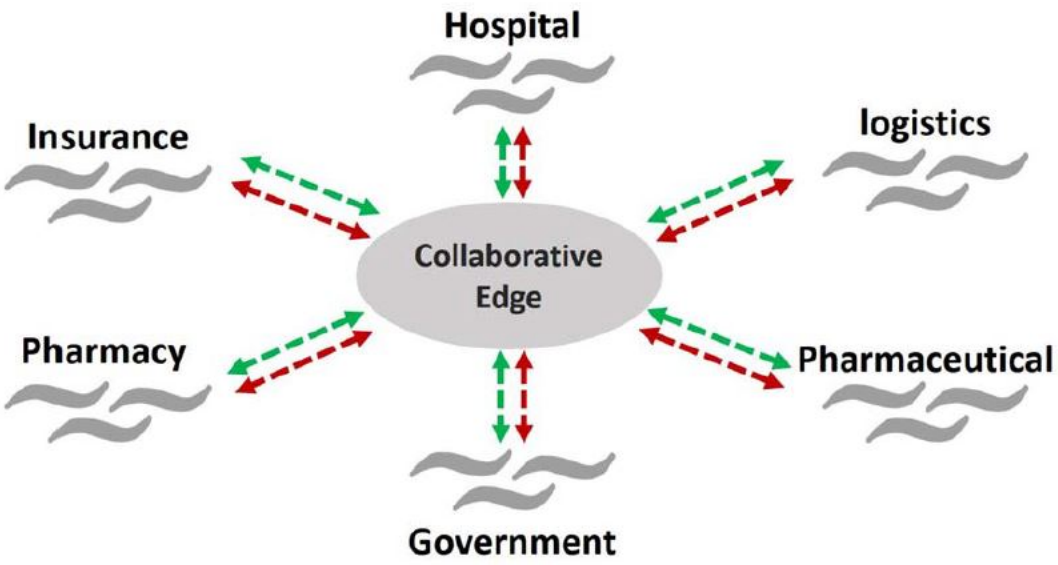

Figure 20: Connected Health [22]

The demand for geographically dispersed applications for data processing, i.e. healthcare, involves data sharing and collaboration between businesses in A multiplicity of domains. The collaborative edge will fuse geographically dispersed data by building virtual shared data views to overcome this challenge. The end-user can leverage shared data by predefined service interface. This public interface helps the end user to compose complex services. These public services are delivered by collaborative edge participants and computation happens only in the data facility of the participant in such a way that data privacy and integrity can be assured.

Connected health care is a good case to show the possible advantages of collaborative edge, for instance, a flu outbreak is a good case study. Patients go to hospitals, and the patients' electronic medical record (EMR) will be revised. For this

flu epidemic, the hospital summarises and shares details such as the average cost, the symptoms, and the population, etc. To get the pills from a pharmacy, a patient would technically obey the prescription. One possibility is that the treatment was not followed by a patient. The hospital must then assume responsibility for re-hospitalization when it can obtain confirmation that the pills were not taken by the patient. Now the pharmacy is able to provide a patient's purchasing record to the hospital through collaborative edge, which greatly enhances healthcare transparency.

The pharmacies can be able to use the collaborative edge which was provided by the hospital to retrieve the population of an outbreak. An obvious advantage is that pharmacies have sufficient inventory to make a lot more profits. Behind the purchase of drugs. The pharmacy would use data supplied by pharmaceutical companies to collect the locations, prices, and inventories of all drug stores. It also sends the logistics companies an application for a transport price question. Then by solving the total cost optimization problem according to retrieved details, the pharmacy may make an order plan.

Several flu prescription orders from pharmacies are also issued by pharmaceutical companies. At this point, the production plan can be rescheduled by a pharmaceutical company and the inventories of the warehouses can be rebalanced. In the meantime as our government representative in this situation, the centers for disease control and prevention monitor the flu population growing at a broad pace. As a result, the range of places will raise a flu warning for the people in the areas involved. Besides, further measures can be taken to avoid the spread of flu outbreaks. After the flu epidemic, the insurance companies have to pay the bill depending on the contract for the patients. The insurance providers will analyze the percentage of people during the epidemic who have the flu.

To change the policy price for the next year, the proportion and the cost of care for flu are important factors. Moreover, if the patient wishes to share it the insurance providers will also have a customized healthcare package based on their EMR. Most

participants will benefit from the collaborative advantage in terms of reducing operating costs and improving profitability in this simple example. Some of them may be pure contributors to the healthcare sector, such as hospitals in this case because they are the key collector of information in this community.[22]

3.7 Challenges and opportunities of Edge Computing

In this section, some challenges are discussed in more detail and introduced some potential solutions and opportunities, including programmability, naming, data abstraction, service management, privacy and security, and optimization metrics. [22]

3.8 Programmability

Users program their code into cloud computing and deploy it on the cloud. The cloud provider must determine where the processing in the cloud is conducted. This is the advantage of cloud computing that makes the infrastructure transparent for the customer since users have partial knowledge of the operation of the program. Since the application only runs in the cloud, therefore, the application usually written in one programming language and compiled for a particular target platform.

However, the edge nodes are most likely heterogeneous platforms then the computation is offloaded from the cloud. The runtime of these nodes varies from each other in this situation, and the programmer faces considerable difficulties in writing an application that can be implemented in the framework of edge computing.

To address the programmability of edge computing, it is suggested that the computing stream definition, defined as a set of functions/computing, be applied to the data along the path of data propagation. The functions/computing may be an application's entire or partial features, and the computing can take place anywhere on the path as long as the application specifies where the computing should be done. The computing stream is software-defined computing flow which processed in a distributed and efficient way on data generating devices, edge nodes, and cloud environment. A

lot of computing can be done at the edge instead of the centric cloud as described in edge computing. In this case, the processing stream will assist the user to decide what functions/computing should be performed and how after the computation occurred at the edge, the data is propagated.

3.9 Naming

A significant assumption in edge computing is that the number of things is enormously high. There are a lot of applications running at the top of the edge nodes, and each application has its structure regarding how the service is provided. The naming scheme in edge computing is very important for programming, addressing, identification of things, and data communication, like other computer systems. A proper naming mechanism for the model of edge computing has not yet been designed and standardized. Edge practitioners typically need to learn different contact and network experiences Protocols to interact with their system's heterogeneous stuff.

The edge computing naming scheme needs to manage the mobility of things, extremely dynamic network topology, protection of privacy and security, as well as scalability targeting the incredibly large amount of unreliable things. Most of the present networks are very well satisfied by traditional naming mechanisms such as DNS and uniform resource identifiers. They are not flexible enough, however, to serve the dynamic edge network, as most of the things at the edge can sometimes be highly mobile and resource-restricted. Besides, the IP-based naming scheme may be too strong for some resource-constrained things at the edge of the network to help considering its complexity and overhead.

For edge computing, new naming structures such as Named Data Networking (NDN) and Mobility First also be implemented.

- **Named Data Networking:** NDN gives the content/data-centric network a hierarchically ordered name, and it is human-friendly for service management and offers edge scalability. To fit into other communication protocols such

as BlueTooth or ZigBee, and so on, it would require an extra proxy, though. Another NDN-related problem is confidentiality, as it is very difficult to separate hardware information from service providers.

- **MobileFirst:** To provide better mobility support, Mobile First should distinguish the name from the network address, and if extended to edge networks where things are highly mobile, it will be very useful. Nevertheless, MobileFirst must be used as a global unique identifier (GUID) for naming, and this is not needed in the aggregation of related fixed details. At the edge of the network infrastructure, such as the home environment. The complexity in service management is another downside of MobileFirst for the edge, as GUID is not human friendly.

Letting the edgeOS allocate network address to each thing may be a solution for a relative small and fixed edge such as home environment. Each thing could have a unique human-friendly name in one system that describes the following information: location (where), role (who), and description of data (what), this thing will be allocated to the edgeOS identifier and network address, as illustrates in Figure 21.

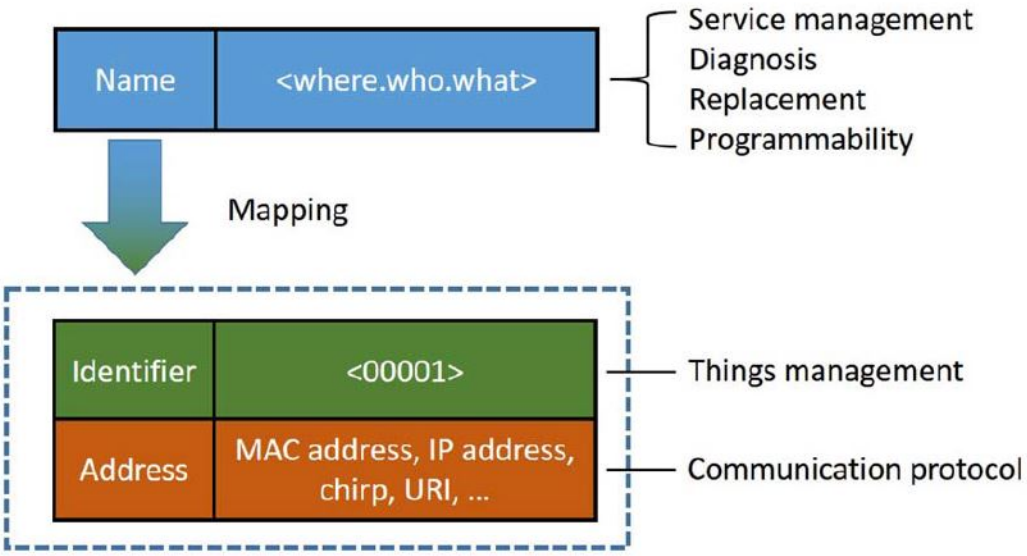

Figure 21: Naming Mechanism in edgeOS [22]

The human-friendly name is special to each thing and will be used for service management, diagnosis of things, and replacement of components. This naming system for the customer and service provider allows simple management. For example, the user will receive an edgeOS message such as "Bulb 3 (what) ceiling light (who) failed in the living room (where)" and then the user will replace the failed bulb directly without looking for an error code or reconfiguring the new bulb's network address. In addition, this naming mechanism gives the service greater programmability.

In the meantime, vendors and service providers are blocked from accessing hardware information to help protect data privacy and security. From a human-friendly name, unique identifier and network address could be mapped. The Identifier in edgeOS will be used to handle items. To help various communication protocols such as BlueTooth, ZigBee or WiFi, and so on, network addresses such as IP addresses or MAC addresses will be used.

3.10 Data Abstraction

By communicating via the air position indicators from the service management layer, various applications may run on the edgeOS that consumes data or provides service. In the wireless method, data abstraction was well explored and researched. The sensor network and the model of cloud computing. In edge computing, this problem becomes more troublesome. With IoT, the network will have a large number of data generators, and here is an example of a smart home environment. Nearly all of the things in a smart home will send details to the edgeOS, not to mention the vast number of things installed all over the home.

Most of the things at the edge of the network, report sensitive data to the gateway periodically. For instance, The temperature might be recorded every minute by the thermometer, but this data would most likely only be consumed by the actual consumer several times a day. Based on this evidence, human interaction in edge computing should be reduced and all data should be consumed/processed by the edge node and communicate proactively with users. In this case data, such as noise/low-quality elimination, event detection, and privacy protection, should be preprocessed at the gateway stage, and so on.

For potential service provisioning, processed data will be sent to the upper layer. In this phase, there will be many challenges. As illustrates in Figure 22, data comes from different formats and reported from different things. Raw data should be blinded from raw data for the benefit of privacy and security applications operating on the gateway. Besides, from an integrated data table, they can derive the information they are interested in. The table with id, time, name, data can be easily specified therefore that the data of any edge thing can be fitted in.

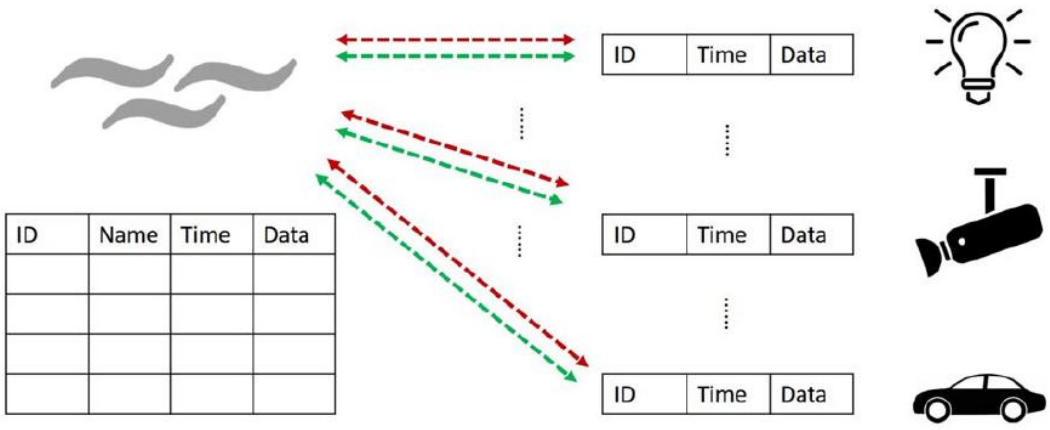

Figure 22: Data abstraction issue for edge computing [22]

Second, the degree of data abstraction is often hard to determine. Some software or utilities may not be able to gain enough information if too much raw data is filtered out. To retain a big amount of raw data, however, there will be a data storage problem. Finally, due to the low-precision sensor, danger setting, and poor wireless communication, data reported by items at the edge may often be unreliable. In this scenario, the abstraction of useful information from unreliable data sources is still a problem for developers of IoT applications and systems.

The applicable operations on the things are another problem with data abstraction. Data collection is to support the application, and to complete those services the user needs, the application should be able to monitor (e.g. read from and write to the things). The data abstraction layer, integrating data representation and operations, will serve as a public interface for all edgeOS-related things. Besides, both data representation, and permitted operations could differ a lot due to the heterogeneity of the things, which also raises the barrier of data abstraction.

3.11 Privacy and Security

Usage privacy and data security are the most critical services to be delivered at the edge of the network. A lot of private data can be learned from the sensed use of data

if a home is deployed with IoT. For instance, one can easily speculate whether the house is vacant or not by reading the electricity or water use. In this situation, the issue is how to help the service without damaging privacy. Before processing, some of the private information may be removed from the data, such as masking all the faces in the video. It may be a good way to preserve privacy and data protection by keeping computing at the edge of the data resource, which means in the home. To protect the protection of data and the privacy of users at the edge of the network, some challenges remain accessible.

The first important thing is the awareness of privacy and security to the end-users. Therefore, all stakeholders such as service providers, system and application developers, and end-users have to be aware that the privacy of end users would dangerous without notice at the edge of the network. for instance, health monitors, Ip cameras, and WiFi toys can be connected by others if there is no proper protection.

Second, the possession of the data gathered at the edge of things. Much as what happens with mobile apps, on the service provider side, the end-user data gathered by things will be processed and analyzed. It would however be a safer option for privacy security to leave the data at the edge where it is processed and let the user completely own the data. End-user data obtained at the edge of the network should be processed at the edge, similar to health record data, and the user should be able to decide whether service providers should use the data. To better preserve user privacy, extremely private data may also be omitted during the authorization process.

Third, powerful mechanisms to protect data privacy and protection at the edge of the network are lacking.Some of the products are extremely resource-constrained, so the existing methods might not be possible to deploy items for security defense, since they are starving for money. Some frameworks, such as Open mHealth, are proposed to standardize and store health data for privacy protection, but there are still more resources lacking to manage different edge computing data attributes. [22]

4 Internet of Things

Cloud computing has been the prevalent standard for the past decade. Computing, control, and data storage have been centralized and transferred into the Cloud, according to this trend. On the other hand, The Internet of Things (IoT) is now becoming widespread. There were approximately 20 billion IoT-linked devices in 2017 and this amount will rise to approximately 30 billion in 2020 and more than double by 2025.[23]. Due to its limitations, the evolving IoT poses many new problems that cloud computing has a hard time facing.

4.1 IoT Definition

The term "Internet of Things" was originally coined in 1999 by Kevin Ashton, executive director of the Auto-ID Center at Massachusetts Institute of Technology (MIT), and then it has assumed several slightly different meanings.In this work, the definition is given by the International and Telecommunication Union (ITU) is assumed: Internet of Things is "a global infrastructure for the information society, enabling advanced services by interconnecting (physical and virtual) things based on existing and evolving interoperable information and communication technologies (ICT)". In this context, a thing is intended as "an object of the physical world (physical things) or the information world (virtual things), which is capable of being identified and integrated into communication networks", while a device is "a piece of equipment with the mandatory capabilities of communication and optional capabilities of sensing, actuation, data capture, data storage, and data processing". In the other words, the Internet of Things is a set of Internet-connected computing devices (namely things) aimed at providing services for all forms of applications, while security requirements are met [23].

4.2 IoT Architecture

Many different IoT architectural models can be found, but, the most commonly used is based on three architectural levels: Perception (or Sensing) layer, Network (or Transmission) layer,the Application layer. The three-layer IoT architectural reference model is depicted in Figure 23. Each architectural layer is characterized by the devices that belong to it and by the functions performed.

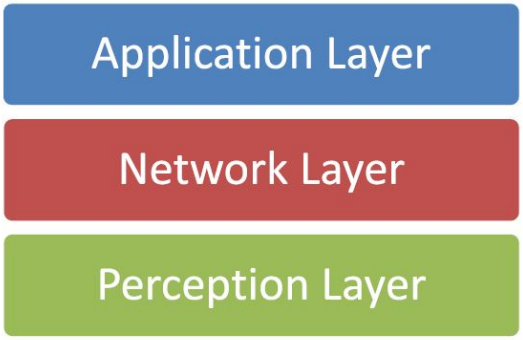

Figure 23: Three-layer IoT architectural model [23]

- **Perception layer:** The purpose of the Perception layer is to acquire environmental data (such as light, temperature, pressure, humidity, etc.) with the aid of sensors and actuators. Basically, before transmitting it to the network layer, the main objective of this layer is the identification and collection of information.

- **network layer:** The network layer is the middle layer and its aims to provide data routing and transmission functions to the correct destination. Therefore, the primary objective of this layer is to transmit data effectively within heterogeneous networks and without losing information. In this layer, Internet gateways, switches, routers, and other network devices run.

- **Application layer:** It is the highest layer that uses the data collected from the bottom layers to implement various services and applications. Typically, this layer contains the user interface, data model-related formulas, business logic, and anything required for the particular IoT service or application.

4.3 IoT Characteristics

The main features of the Internet of Things are summarized below:

- **Interconnectivity:** It is possible to interconnect anything in IoT with the global connectivity and information infrastructure.
- **Things-related services:** Within the limitations of things, such as security and semantic compatibility between physical and virtual things, IoT can provide thing-related services.
- **Heterogeneity:** IoT equipment may be based on various networks and/or hardware platforms. Besides, they can connect with various service platforms and/or Via various networks, devices.
- **Constrained resources:** IoT typically includes devices with energetic and computational constraints.
- **Dynamic changes and uncontrolled environment:** In IoT, the state of the devices (e.g., sleeping/awake, connected/disconnected) and context (e.g., position, speed) dynamically change. Therefore, IoT devices are part of an unregulated ecosystem characterized by unstable environments and interactions between them. Due to both unstable network access and complex system state shifts, devices are unreliable. Furthermore, the number of devices can change dynamically.
- **Huge scale:** The number of devices that need to be handled and that need to connect is immense and will be much greater in the future. Besides, the proportion of device-triggered communications will continue to rise to the detriment of human-triggered communications. The management and analysis of data created by such devices to exchange information with each other will be even more important [23].

4.4 IoT Challenges

IoT has brought some challenges which are driving the increasing interest for Edge and Fog computing, as solutions to such difficulties. some of these challenges are listed below:

- **Low Latency:** Low latency and jitter (within a few milliseconds) are often needed for both industrial control systems and IoT applications. The Cloud computing model is certainly not within the scope of this criteria.

- **High Network Bandwidth:** A large amount of data is increasingly produced by the rising number of connected IoT devices. It takes an extremely large network to transfer all this information to the cloud. Bandwidth and often useless or not allowed (e.g. due to concerns about data privacy). Thus, without involving the cloud, the data generated at the edge of the network often needs to be stored and processed locally.

- **Limited Resources:** Several IoT devices have very limited resources (such as sensors, drones, vehicles, etc.).This implies that they are unable to communicate directly with the Cloud, as these connexions often involve either complex protocols or computational intensity. As a result, devices with resource constraints must depend on an intermediate interface layer to link to the cloud.

- **IT and OT Convergence:** Industrial networks have recently undergone the integration of Operational Technology (OT) [10]and Information Technology (IT) [11] with the advent of Industry 4.0.This trend brings new goals and organizational requirements for the business. In modern cyber-physical systems, the incessant and secure operation is always a priority, because an offline system may cause a remarkable business loss or an unnecessary client inconvenience. As a result, it is a challenge to upgrade hardware and software in such systems. The consequence

[10]Hardware and software systems used to monitor and control physical processe
[11]Hardware and software systems used to process, transmit, and store business data.

is the need for a new architecture that, over time, eliminates the need for system updates.

- **Intermittent Connectivity:** Some IoT devices have intermittent network access (e.g., vehicles and drones). As a result of this, it is difficult to provide such devices with uninterrupted cloud services. To minimize or solve the problem, it is therefore important to rely on an intermediate layer of devices.

- **Geographical Distribution:** In wide geographical areas, the large number of IoT devices needing computer and storage facilities are distributed. It is therefore difficult to find a place for the cloud infrastructure that enables all IoT application specifications to be met. To bridge this distance, an intermediate layer of devices is useful.

- **Context Awareness:** Many IoT applications, such as vehicular networks and augmented reality, need local context information (e.g. user location, network conditions) to be accessed and processed. Due to the physical distance between IoT devices and central computing, this provision does not involve a centralized approach to cloud computing.

- **Security and Privacy:** Emerging issues in IoT security and privacy are unique. Today, cybersecurity solutions are aimed at protecting businesses and consumers through firewalls, intrusion prevention systems (IPSs), and intrusion detection systems (IDSs) that provide perimeter-based protection. Unfortunately, this paradigm is no longer sufficient to solve the new security problems raised by the IoT.

  It needs fundamental changes to current paradigms to overcome these challenges. That is where Edge and Fog computing comes in, offering a new technical trend designed to establish the missing link in the continuum of Cloud-to-Things.[23]

5 Mist and other related Computing Paradigms

Computing technology has reached a new era with the advent of cloud computing. This common computing model is currently nurtured as a utility by many computer service providers such as Google, Amazon, IBM, Microsoft, etc. They have allowed cloud-based services such as Infrastructure as a Service (IaaS), Platform as a Service (PaaS), Software as a Service(SaaS), etc. To manage various enterprise and educational issues simultaneously.Most cloud data centers, however, are geographically centralized and far from the vicinity of the end devices/users. As a result, requests for real-time and latency-sensitive computing resources to be addressed by remote cloud data centres frequently suffer from significant round-trip delays, network congestion, loss of service quality, etc. In addition to centralized cloud computing, a new paradigm called "Edge computing" has been proposed to address these issues.[24] The basic concept of Edge computing is to have computing facilities Closer to the data source. More specifically, Edge Computing allows edge network data processing. The edge network consists of end devices (e.g. cell phones, smart objects, etc.), edge devices (e.g. border routers, set-top boxes, bridges, base stations, wireless access points, etc.), edge servers, etc., and it is possible to equip these components with the necessary functionality to help edge computing. Edge computing offers quicker responses to requests for computing resources and most frequently resists the sending of bulk raw data to the core network. In general, however, Edge computing does not spontaneously associate IaaS, PaaS, SaaS, and other cloud-based services and focuses more on the end device side [25]. Several computing paradigms have already been introduced in computation technology, such as, Cyber Foraging, Cloudlet, Mobile computing(MC), Mobile Cloud Computing(MCC), Mobile-Edge Computing(MEC), Edge Computing, Mobile ad hoc cloud computing (MACC), Among them Mobile Edge Computing (MEC), Mobile Cloud Computing (MCC) are considered as the potential extensions of Cloud and Edge computing. In the next section the related

computing paradigms are discussed in the order of their trend and show how some paradigms resulted in the emergence of others.

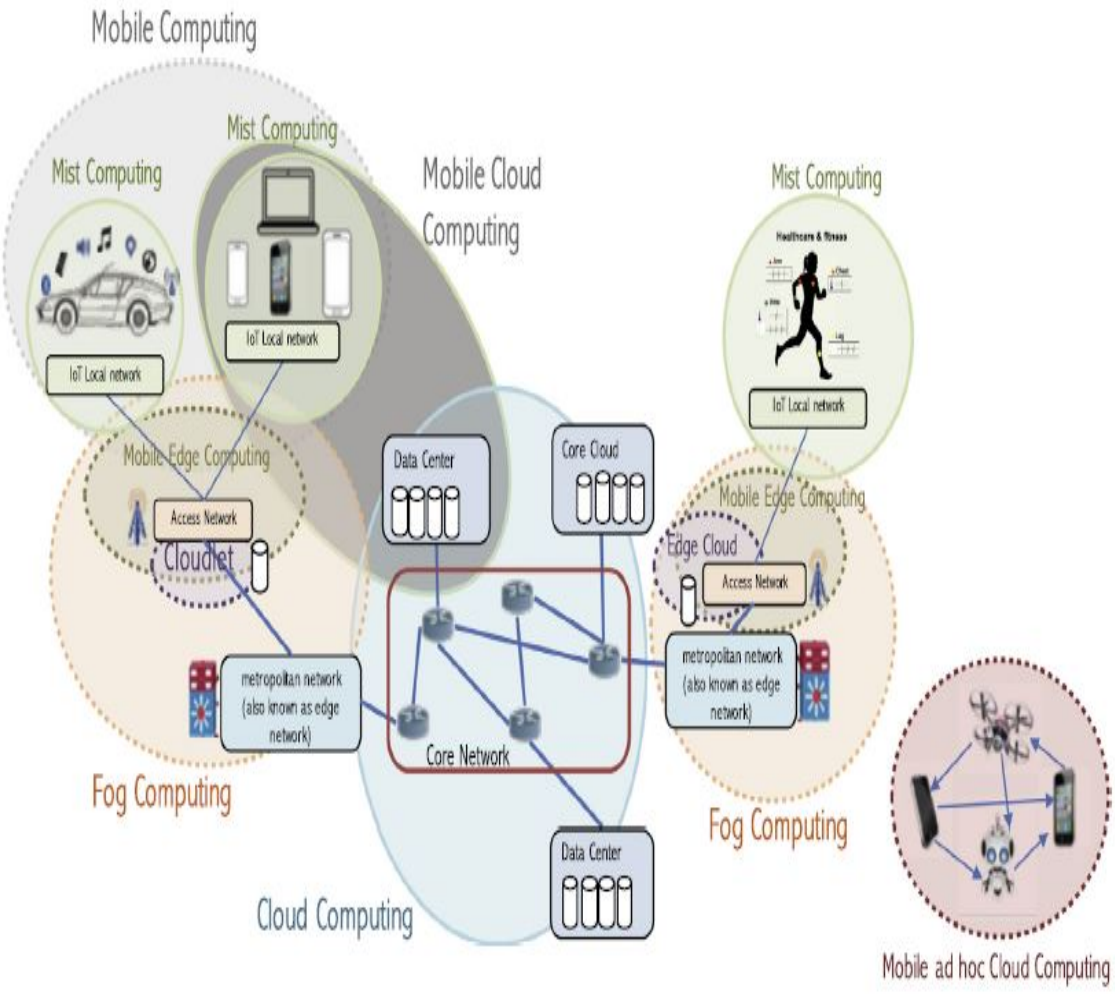

Figure 24: Comparison of the infrastructure of fog computing and its related computing paradigms from the networking perspective [25]

5.1 Cyber Foraging

Cyber foraging is one of the first edge computing ideas, although it has now been replaced by more recent concepts such as cloudlet, MEC, and fog. It was introduced in 2001 by Satyanarayanan [26] and was further refined in 2002 by Balan et al [27]. In cyber foraging, Resource-limited mobile devices exploit the capacities of nearby servers, connected to the Internet through high-bandwidth networks. Computing and

data staging is done by these servers called surrogates. The process of prefetching distant data to nearby surrogates is data staging. For example, when a mobile device has to process a compute-intensive part request, such as face recognition, which needs to access a large amount of data for face matching, it captures raw images and offloads the complex processing to surrogate. The surrogate performs the processes of face recognition and matching using For a database. This surrogate may install the database on its local disc and, on behalf of the mobile device, perform all or some part of the processing. It then delivers the result to the mobile device with low latency, because it is close to the device. If the mobile computer fails to locate a nearby surrogate server, It may provide the end-user with a degraded service due to its limited capabilities [28].

5.2 Cloudlet

The cloudlet definition was next proposed in 2009 by Satyanarayanan et al.[38]. This is referred to in [29] as cloudlet-based cyber foraging. Cloudlets reuse modern methods of cloud computing, such as virtual machine (VM) based-virtualization. They are resource-rich servers or server clusters located near mobile devices in a single-hop environment. They run one or more VMs in which mobile devices can offload components for expensive computation. Back to the application for face recognition, the face detection and matching processes will be done on VMs rather than actual machines using cloudlets.

Virtualization allows Cloudlets to dynamically expand and shrink. This ultimately contributes to scalability about the service demands of mobile users. Besides, the VM distinguishes the guest software environment from the cloudlet's host software environment, which in turn improves the probability of mobile users finding a compatible cloudlet anywhere in the world to unload their computer-intensive requests. Using cloudlets, resource-poor mobile devices offload their intensive computations (e.g., face recognition) to the cloudlets they use, thereby guaranteeing real-time interactive re-

sponses. It can link to a distant cloud if the mobile device moves away from the cloudlet, thereby providing a degraded service. Although cloudlets reflect the middle level of a three-tier hierarchy (i.e., mobile device-cloudlet-cloud), there is no clear emphasis on cloud interactions in the current concept of cloudlets. Cloudlets may also act on the edge as a complete cloud. Even when they are completely isolated from the cloud, Since VM provisioning of the cloudlets is done without cloud interference, it can exist as a standalone environment.

5.3 Mobile Computing

The advancement in fog and cloud computing is affected by the groundwork developed by mobile computing. mobile computing is done via mobiles, portable devices. computing can be used to create pervasive context-aware applications, such as location-based reminders. Mobile devices, which can be connected via Bluetooth , WiFi, ZigBee, and other cellular protocols, are the only form of hardware that mobile computing requires. In comparison, fog and cloud computing need virtualization capabilities of more resource-rich hardware. Security in mobile computing must be done on the mobile device itself. However, fog and cloud computing, mobile computing is more resource-constrained, but in recent years, advancements in mobile hardware and wireless protocols have significantly reduced this gap.

Mobile computing's strength comes from the distributed architecture. This architecture benefits from distributed applications since mobile machines do not require a centralised location to function. However, mobile computing comes with many disadvantages, such as weak resource limitations, Communication latency, and the need for mobile customers to adapt to changing environments effectively. Such disadvantages also make mobile computing unsuitable for current applications that need low latency or robustness, or that need to produce, process, and store large volumes of data on computers.[5]

5.4 Mobile Cloud Computing

As cloud computing and mobile computing matured, This combination resulted in mobile cloud computing (MCC), which is defined as a platform that offers unrestricted features, mobility, and storage via heterogeneous network access. This technology also provides scalable computing services by following the pay-per-use model. In mobile cloud computing, resource-contained mobile devices can utilize resource-rich cloud services.MCC transfers the majority of computation from mobile devices to the cloud.

MCC provides to run computation-intensive applications and to boost the battery life of mobile devices.MCC shares a combination of capabilities and characteristics in cloud computing and mobile computing which leads to the high availability of computing resources compared to mobile computing. This enables high-computation technologies, such as mobile augmented reality, to emerge. Also, in MCC, the availability of cloud-based applications are far greater than that of mobile computing. MCC depends on the cloud for running high-computation services like cloud computing and fog computing. MCC computing can also be operated by mobile devices. Security in MCC must be provisioned on both mobile devices and in the cloud, similar to cloud computing.

MCC also has some limitations like mobile computing. The main advantages and some related issues are illustrated in figure 25. First, while a centralized MCC architecture is perfect for sharing a pool of computing resources, this might not be well suited for applications where the pervasiveness of devices is desired. Secondly, as both cloud computing and MCC need cloud-based services, and as these services are linked via the WAN connection, applications running on these platforms need to connect to the Internet at all times. Finally, for delay-sensitive applications, offloading computation to the cloud allows the latency to be reasonably high [5].

Among other problems facing MCC are bandwidth restriction in wireless access

networks and QoS support for mission-critical applications. On the security and privacy side, knowledge of the location and activities of the user by the service provider could cause problems with privacy. In general, cloud computing security problems, such as data leakage and data outsourcing issues, still exist [4].

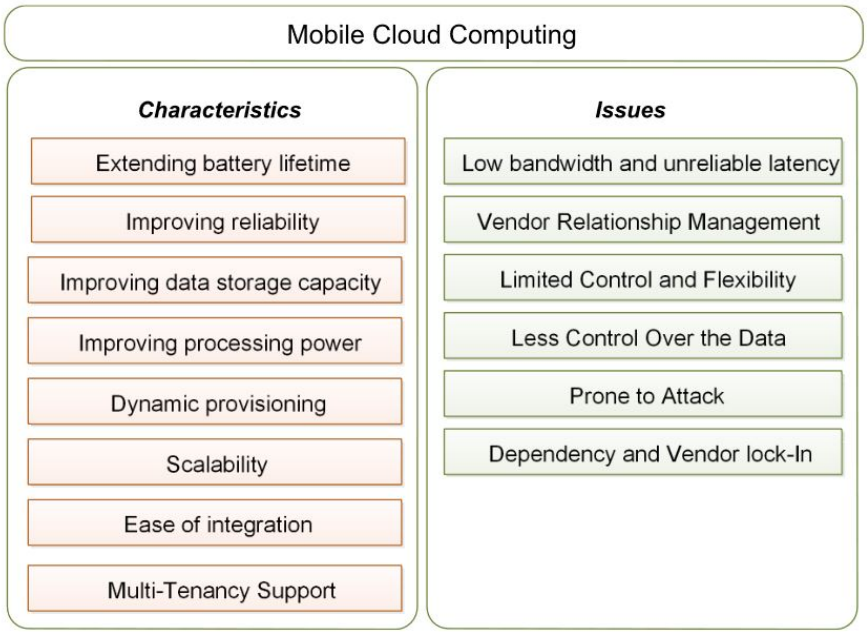

Figure 25: Mobile cloud computing characteristic and issues [4]

5.5 Mobile Edge computing

MEC suggests that computing and storage facilities be co-located at the base stations of cellular networks. In a remote area, MEC may either be connected or disconnected to cloud data centers. MEC allows two or three-tier hierarchical application deployments with end devices. It leverages the MEC server to be deployed near base station towers to process and store at the edge.

Some participants utilize this computing paradigm, such as, the mobile end users, network operators,Internet infrastructure provider (InPs), and application service provider. Mobile end users are the main consumer of the system and request their service via user equipment(UE). Then, the operation of the base stations, the mobile core network, and MEC servers are managed and maintained by network operators.

Internet access and routers are maintained by InPs. Providers of application services host the application services in the content delivery network (CDN) or within data centers. The nearest MEC will be checked for while handling applications from the UE. The MEC server will process user requests rather than forward them to remote Internet providers. In a situation where an application can not be processed or completed at the MEC, The request will be sent to remote CDNs or data centres[30]. MEC is the evolution of mobile base stations, according to [31].

It is collaborative telecommunication and IT networking implementation. This computing paradigm provides individual end-users and business customers with new vertical business segments and services. Through this computing model, various services could be offered which could not be done with traditional network architecture, figure 26 including IoT, location services, augmented reality, caching service, video analytics, and delivery of local content. It can provide low-latency access to local content in real-time or through caching content from the MEC server. The key drawback of this framework, however, is the MEC server installation, which is explicitly dedicated to MEC services. Scaling is another major problem with the growth in the demand for services over time.

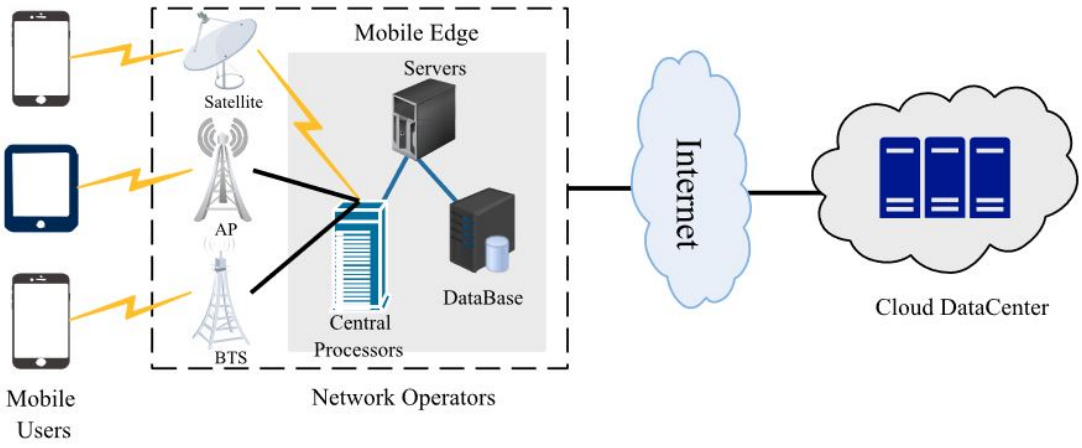

Figure 26: MCC/MEC computing architectures [4]

5.6 Mobile ad hoc cloud computing

This computing model is not always ideal for scenarios in which there is a lack of infrastructure or a centralized cloud, considering the ubiquitous existence of MCC. An ad hoc mobile network is the most decentralized type of network, consisting of nodes that form a temporary, dynamic network via routing and transport protocols. In an ad hoc mobile network, mobile devices form a highly dynamic topology of the network; the network created by mobile devices is highly dynamic and must handle devices that connect or leave the network continuously. Clouds that can be used for networking, storage, and computing can be created by ad hoc mobile devices. Instances such as disaster relief, group live video streaming, and unmanned vehicular systems could be used by MACC.

MACC vs cloud computing

Mobile ad hoc cloud computing (MACC) is different from cloud computing, basically because of the nature of ad hoc resources. MACC involves mobile devices that act as data providers, storage, and processing devices. Because of the lack of network infrastructure, Mobile devices in a mobile ad hoc cloud network are responsible for routing traffic among themselves.MACC provides relatively high computation by pooling local mobile resources to form an ad hoc cloud. In cloud computing, these attributes vary from target customers, architecture, and accessibility [5].

MACC vs MCC

In hardware, service access process, and distance from users, MACC is also different from MCC, as computation is performed on mobile devices in MACC, while in MCC it is far from mobile devices. MACC only requires mobile devices to run, while, in addition to mobile devices, MCC requires large-scale data centers used for cloud storage. In MCC, this results in high computing power, but also higher latency. MACC security must only be given on mobile devices, but maintaining trust in MACC

can be difficult without a stable framework for collaboration. Finally, in MACC, only mobile devices that are connected via Bluetooth, WiFi, and other cellular protocols have access to services [5].

MACC vs Fog

While fog computing can be done across a variety of devices rich in resources and deficient in resources, mobile ad hoc cloud computing is best suited to highly decentralized, dynamic topologies of networks in which Internet access is not guaranteed. Compared to fog computing, the connected devices in MACC are more decentralized, allowing devices to form a more dynamic network at locations of sparely connected devices or a rapidly changing network. An example of this is an ad hoc peer-to-peer file-sharing network [5].

5.7 Mist Computing

In recent years, *mist computing* has been introduced to capture a more extreme edge of end devices [32]. This computing paradigm describes dispersed computing at the IoT devices themselves with the horizon of future self-aware and autonomic systems in mind. Mist computing be can be viewed as the very first computing location in the IoT-fog-cloud continuum. It can be called informally "IoT computing" or "things computing." A wearable computer, a mobile device, a smartwatch, or a smart fridge could be an IoT device. Mist computing spreads computing, storage, and networking across the things through the fog. In a way, mist computing is a MACC superset; since the networking does not necessarily be ad hoc in mist, and the devices may not be mobile devices (more details are provided in Figure 29).

The authors in[33] present the concept of using nearby mobile devices for storage, caching, and computing purposes as a cloud computing environment. They research the use of mist computing to decrease the load for video dissemination applications in traditional WiFi infrastructures. In this research, spectators of a sporting event

organise themselves into WiFi-Direct groups and, wherever possible, share video replays, bypassing the central server and access points. "Another example of mist computing is this report, in which IoT devices not only operate as" thin clients "but also as" thin servers. Some other applications of mist computing are to protect the privacy of user data by local processing, and to deploy virtualized instances efficiently on single-board computers.

Mist computing's basic objective is to bring computing, i.e. sensors and actuators, to the very edge of the network. As in real life, physical systems would not be used if there is a connectivity failure between the cloud and the IoT device. IoT devices do not rely on the internet. IoT systems should not be able to use local intelligence using the guidelines given to act in the event of failure.

5.8 Guiding Principles of Mist Computing

- Network must provide information but not simply data.
- Only information that has been requested should be given by the network and only when requested.
- A dynamic system should be created base on information requires with interacting end devices using a subscriber provider model.
- Devices must be aware of the situation and adapt to the information needs and configuration of the network. We should not have rules for static bindings for devices and data providers. The devices must dynamically discover the data providers and execute application.

The cloud and fog are aware of the needs of the user and the global situation, while the mist is aware of the physical environment and local conditions. So the responsibility is to execute an IoT application together. The global situation must be communicated to the edge devices in order to do this, and the edge devices must

be able to grasp what or how they ought to respond in such circumstances. The IoT application then actually stretches from the very end of the edge network to the cloud. there are some differences between edge computing and mist computing. In edge computing, fixed data processing functionality exists at the edge of the network, application configuration is fixed, while in mist computing, there are functionalities, dynamic and flexible timings, high-level application-specific rules are possible, new applications can be assembled at runtime from existing devices [34].

5.9 Routing in the Mist:

For routing in mist computing, conventional wireless routing protocols are not suitable. The routing protocols here support communication from device to device. Suboptimal routing paths have been found to increase network bandwidth requirements. Also, any node should be able to connect to any gateway in the mist, reducing reliance on a particular gateway. In a huge network, often gateway failures cause a new gateway to be added, the nodes near that gateway should be able to link dynamically to such a node and efficiently create a new route between itself and the gateway so that the information unavailability crisis is resolved within the tolerance span [34].

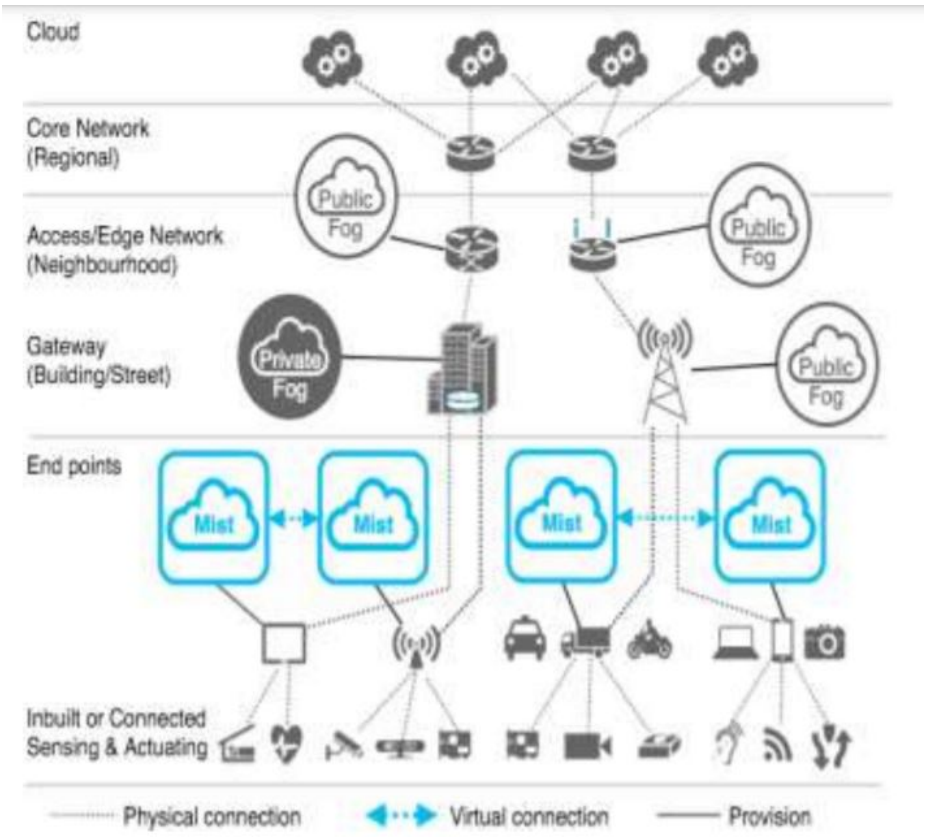

Figure 27: Architecture Of Mist Computing [34]

5.10 Mist Architecture

As shown in Figure 27, the mist nodes responsible to process the data to be handed over to the IoT devices that include sensors and actuators with a physical link between them. The mist nodes also monitor the quality of link parameters. The functionalities at the gateway level are loading updated application rules and tuning application parameters, monitoring local node health, computer-intensive service execution. Finally, new applications can be deployed at the cloud level and applications can be coordinated along with service quality management and health monitoring of running applications.

Figure 28 demonstrates the classification of fog-related computing paradigms and their overlap in terms of their scale. Comparison of fog computing and it's associated

computing paradigms are shown in the figure.

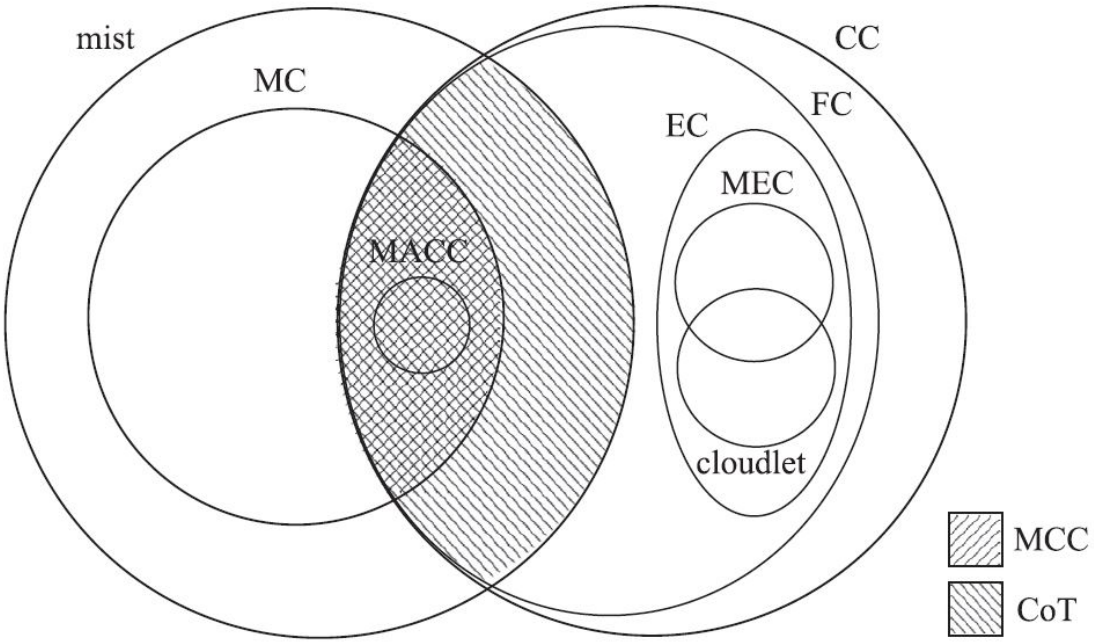

Figure 28: A classification of the scope of different computing paradigms [5]

5.11 Paradigms Comparison

All previous discussions about fog computing and related paradigms clearly show the importance of understanding the characteristics of these platforms in IT landscape. The strength and weaknesses of these computing paradigms vividly separate them from each other and make some paradigms more appropriate to particular use cases.

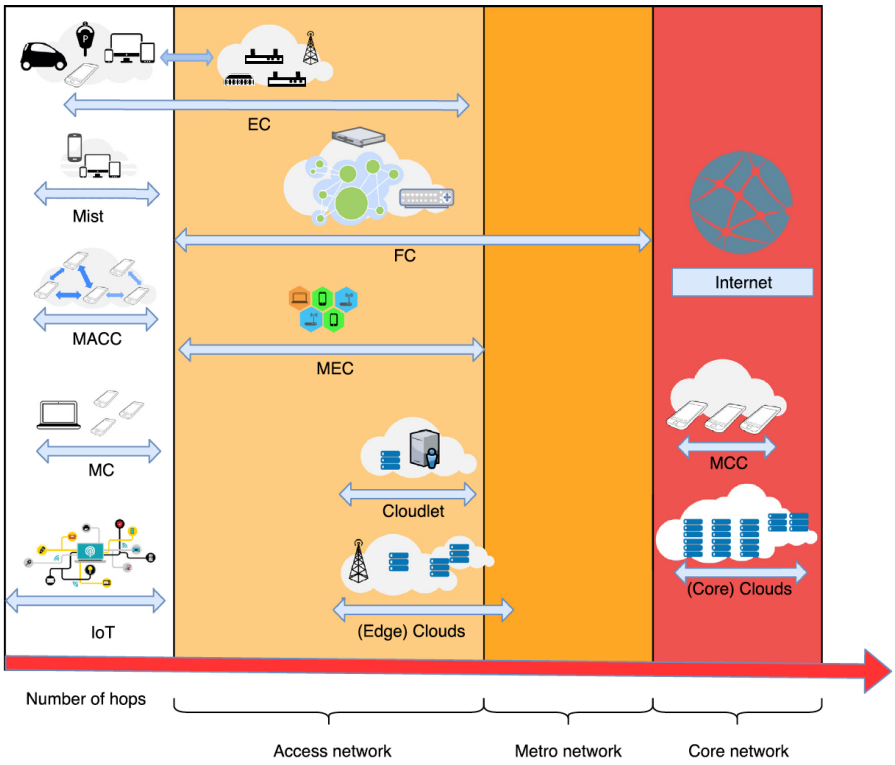

Figure 29: Comparison of fog computing and all other related computing paradigms [5]

The characteristics of computing paradigms are differentiated and summarized in Table 3.

6 Towards Fog Computing

It is estimated that by 2020, the number of connected "things" (devices, gadgets, home appliances, phones, computers, etc) will reach 50 billion. From one side, all these IoT devices are constantly generating data and, from the other side, most often they need to execute rapid analysis on the data and provide instant results. For example, consider a scenario in which a latency-sensitive application requires a quick act on data. In such a case, sending data to the cloud does not seem to be a good choice, because transferring a huge amount of data consumes a lot of bandwidth and,

Characteristics	CC	MC	FC	EC	MCC	MACC	Mist
users	General	Mobile	General	General	Mobile	Mobile	Mobile
Architecture	Centralized	Distributed	Decentralized/ Hierarchi-cal	Localized/ Distributed	Central cloud with distributed mobile devices	Distributed	Localized/ Distributed
Distance from user	Far	very close	Relatively close	Close	Far	Very close	Very close
Operator	Cloud service providers	Self-organized	Users and cloud service providers	Network infras-tructure provider-s/local businesses	Users and cloud service providers	Self-organized	Self-organized/local business
Service Type	Global	Local	Less global	Local	Local	Local	Local
Availability	High	Low	High	Average	High	Low	Low
Latency	High	Moderate	Low	Low	High	Moderate	Moderate
Power-Consumption	high	—	low	Low	Low on mo-bile devices	Low	Low
Server-Location	Installed in large dedi-cated buildings	—	Can be installed at the edge or in dedicated locations	Near edge devices	Installed in large dedicated buildings	—	—

Table 3: Comparison between Cloud Computing(CC), Mobile Computing (MC), Fog Computing(FC), Edge Computing(EC), Mobile Cloud Computing (MCC), Mobile Ad-hoc Computing and Mist Computing(MC) paradigms [35]

at the same time, adds additional latency to computations. Furthermore, Wide Area Network (WAN) latency adversely impacts energy efficiency and interactive response between the things and the cloud infrastructure. Taking all limitations into account, it is concluded that the cloud itself is not capable of connecting millions of things spread over large geographically distributed areas. Hence, Cisco [36] introduced the notion of Fog Computing as the extension of the cloud computing paradigm which moves the computing resources to the proximity of users from the core to the edge of the network.

Indeed, Fog computing bridges the gap between traditional cloud infrastructure

and the things, and provides storage, computation, and networking services. It is considered as complementary to the cloud, addressing emerging IoT applications that require low latency and/or fast mobility support.

Fog Computing enables organizations to perform an inordinate amount of real-time and low-latency IoT data locally at the source without consuming the organization's network bandwidth to send all the data back and forth to the cloud. Furthermore, by analyzing and processing data closer to the source, Fog Computing can significantly improve the performance of the system in terms of computations and service execution.

6.1 Definition of Fog Computing

Fog computing is a distributed paradigm of computing in which processing is done at the edge of the network with seamless integration of the cloud infrastructure. It provides a computing facility for IoT environments or other applications that are sensitive to latency. Sending data from all connected devices for analysis and processing on the cloud will require a huge amount of bandwidth and storage. All devices are not connected to the controller via IP but connected by some other industrial IoT protocols. Because of this, a process of translation is also required for the processing or storing of IoT device's information. The Fog computing paradigm offers an ideal location to immediately analyze most data near the devices that generate and function on that data. The Fog is placed close to things that are able to process the produced data and act on it. The devices inside the Fog setting are referred to as Fog devices. These nodes can be deployed anywhere with network connectivity: At the power pole, at the factory, next to the road, next to the railway line, in a vehicle, inside a shopping mall, on an oil rig, etc. A fog device is defined as a device that has processing,Memory, storage, and network capability. Although the Fog extends the cloud, technically it is placed between the cloud and IoT devices and provides processing and storage tasks in close proximity to the user. But the definition of fog

is debatable. Table 4 illustrates Fog definitions provided by different research works.

Defined by	Characteristics
Bonomi et al.	Highly virtualized, Reside between IoT devices and cloud, Not exclusively located at the edge
Cisco systems	Extend the cloud, Generally used for IoT, Can be deployed anywhere, Fog devices consists of processing, storage, and network connectivity
Vaquero and Rodero-Merino	Heterogeneous, ubiquitous and decentralized devices communication, storage and processing are done without third party invention, run in a sandboxed environment, leasing part of users devices and provide an incentive
IBM	Defined fog and Edge computing as the same notion, Not depends on centralized cloud, Resides at network ends, Place some resources and at the edge of the cloud

Table 4: Summary of Fog computing definitions [36]

6.2 Differences between cloud and fog computing

Fog computing architectures are based on Fog clusters where many Fog devices participate. On the other hand, data centers are the key physical components of clouds which increase operational costs and energy consumption. By comparison, in the Fog computing model, energy consumption and operating costs are low.

The Fog is located closer to the user, so one or a few hops may be the distance between users and Fog devices. This distance leads to having high latency for the cloud compared to the fog and also the cloud is a more centralized technique, while the Fog is a more dispersed geographical orchestration-based solution. Due to the high latency in the cloud, real-time interaction is not feasible for the cloud, but this issue can be easily overcome by Fog computing.

On the other hand, due to wireless connectivity, decentralized control, and power failure, the Fog failure rate is high. Since smart gadgets and handheld devices will participate in Fog systems, most devices in Fog environments will be wirelessly connected. These machines and other maintenance of the network devices are decentralized. When software is not correctly controlled, these devices may fail. Users may

not be aware of malicious software that may result in the failure of the system. Also, Fog processing may also fail in other instances, e.g. each Fog system is responsible for processing its application.

Therefore, the processing of the IoT application on a Fog system often takes a second priority. If the Fog device is fully utilized by the application of the device itself, then it will not be able to do any fog processing. The scheduling of apps and resources in the Fog is even more difficult. Furthermore, the Fog's failure management is competitive due to power failure, which is just a concern because the systems operate on battery power. In all, Table 5 indicates the technical differences between the cloud and the fog.

	Fog	Cloud
Participating Nodes	Constantly dynamic	Variable
Management	Distributed/Centralized	Centralized
Power Source	Battery/Direct Power/Green Energy	Direct Power
Power Consumption	Low	High
Computation Capacity	Low	High
Storage Capacity	Low	High
Network Latency	Low	High
Computation Cost	Low	High
Number of Intermediate hop	One/few	Multi
Nature of Failure	Highly diverse	Predictable

Table 5: Differences between Fog and Cloud [37]

6.3 Fog-Cloud Federation

Cloud and fog computing have obvious differences and trade-offs, it might be confusing which one to choose. Fog and cloud, however, complement each other; the

need of the other can not be replaced by one. Combining cloud and fog computing improves the data aggregation, processing, and storage capabilities. For example, the fog might filter, preprocess, and aggregate traffic streams from source devices in a stream processing application, while queries could be sent to the cloud with heavy analytical processing, or archival results. The collaboration between cloud and fog could be handled by an orchestrator. In particular, a fog orchestrator may provide an interoperable pool of resources, deploy and schedule application workflow resources, and monitor QoS. Via the use of SDN, fog service providers would have greater control over how a large number of fog nodes that transfer data between cloud and IoT devices are configured for the network [5].

6.4 Architecture of Fog Computing

For market adoption and deployment, Fog computing must have a standard architecture. There is no available standard architecture to date. However, many research works have presented Fog computing architectures. In this section, firstly the high-level architecture of Fog computing is discuss. Furthermore, some proposed architectures for Fog computing are summarized. Finally, a detailed architecture for Fog computing with a comprehensive description of each component of the architecture is presented [37].

6.5 High-level Architecture of Fog Computing

The Fog computing paradigm in high-level architecture has three separate layers, as shown in Figure 30.

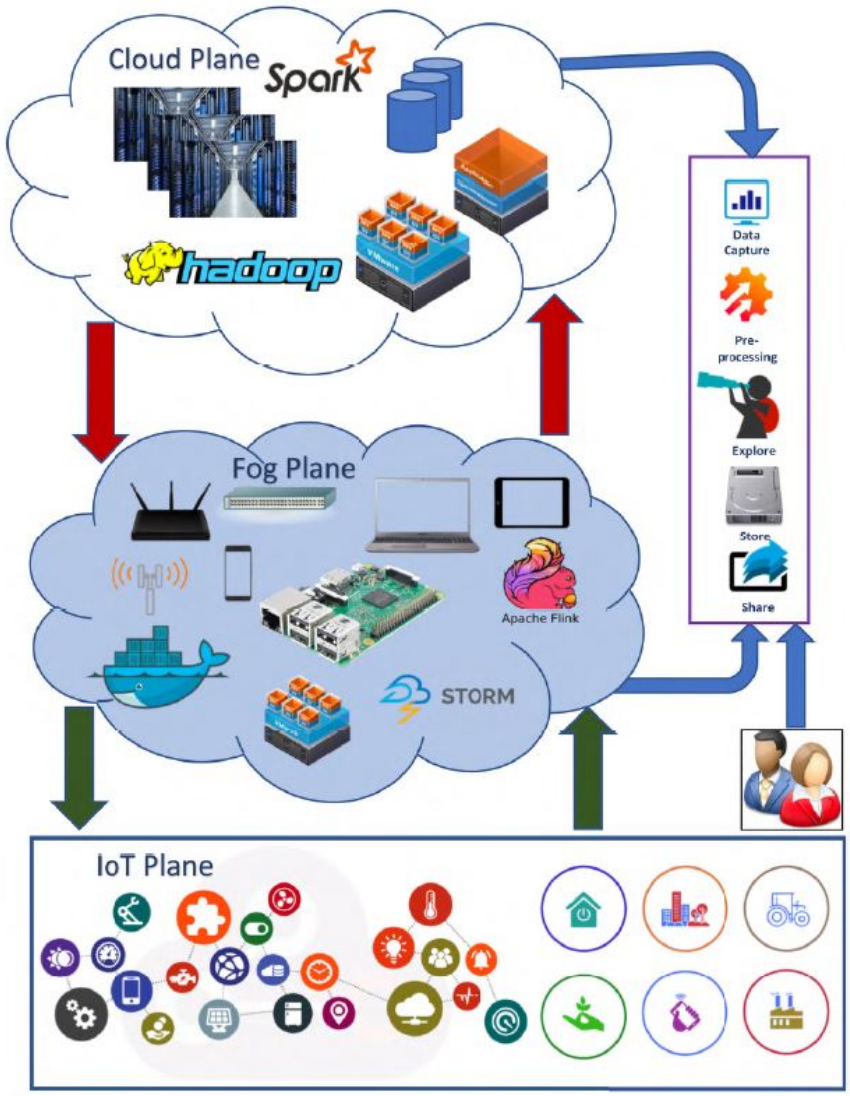

Figure 30: High-level Fog Computing Architecture [37]

There are three layers in this architecture: 1) IoT layer, where the IoT devices and end-users are located; 2) fog layer, where fog nodes are placed; and 3) cloud layer, where distributed cloud servers are located. A cloud server can consist of multiple processing units, such as a physical server rack or a server with multiple processing cores. Nodes are split into domains in each layer where a single application for IoT-fog-cloud Implementation is completed.

The fog layer is the most significant layer. This layer consists of all devices for intermediate computing. Traditional virtualization technology, similar to the cloud, can

be used on this plane. However, taking into account the accessibility of services, it is more fitting to use container-based virtualization. This layer collects sensor-generated IoT layer data and, after processing, sends an action-related request. The big data problem tends to be solved by processing generated data at the edge level, with billions of devices producing big data problems. In reality, the small and medium-scale processing of big data can be used at this stage.

The bottommost layer is the IoT plane, which consists of all connected devices. The devices on this plane perform the sensing and actuation process. Processing should be performed exclusively on the Fog plane for time-sensitive apps, while the cloud can conduct other processing that is not time-sensitive. The fog layer can, however, manage what needs to be sent to the cloud and what should not be sent. Based on their request, users can get services from both the Fog and the cloud. However, the cloud plane will manage complex processing and storage [37].

For example, Figure 31 shows that a domain of IoT nodes (for example, in a factory) is shown in dark green and that they interact with a domain of fog nodes associated with the application.

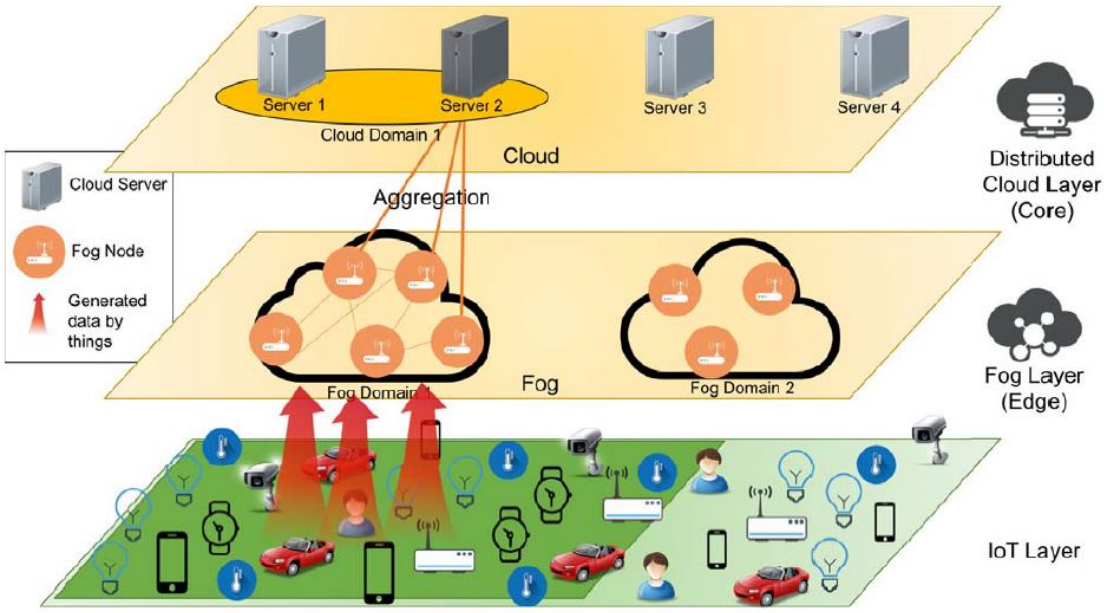

Figure 31: General framework for IoT-fog-cloud architecture [38]

IoT domain can include several IoT devices (or sensors) in a factory, in a farm, or in a smart home where all these sensors in a neighborhood vicinity make a single domain. All the nodes in one domain are typically located in close proximity to each other, such as in a neighborhood or at building levels. A collection of cloud servers for a single domain is associated with each domain of fog nodes.

The following is the basic way in which IoT nodes, fog nodes, and cloud servers work and communicate.

- IoT nodes can locally process requests, send them to a fog node, or send them to the cloud.
- Fog nodes can process requests, forward requests to other fog nodes within the same domain, or transmit requests to the cloud.
- Cloud servers manage requests and return a response to the IoT nodes.

6.6 Hierarchical Fog Computing Architecture

A hierarchical fog computing architecture is introduced in [39], in each fog node to provide flexible IoT services while maintaining user privacy. IoT devices for each user are connected with a proxy VM (located in a fog node), which collects, classifies, and analyses the raw data streams of the devices, transforms them into metadata, and transmits the metadata to the respective VMs of the application (which are owned by IoT service providers). The corresponding metadata from different proxy VMs is obtained by each application VM and its service is given to users.

Most of the data created by devices from users include personal information, such as photos/videos taken by cell phones and smart cars, GPS data, wearable device health information, and sensed smart home status by the sensors deployed in a smart home. Not only the individual, but even all of society will benefit from analyzing such humongous results. Analyzing the photos/videos captured by cameras, for example, may recognize and monitor a terrorist. In particular, the application provider sends each fog node a photo of the terrorist, and each fog node locally conducts face matching to compare the photo of the terrorist with the photos/videos taken by local devices. If matched, the respective photos/videos will be submitted to the cloud by the fog node for further processing. Thus, in order to offer such services, it seems that consumers have to share their personal details. Maintaining user privacy in the provision of such services is the main challenge.

A hierarchical architecture of fog computing is proposed to tackle this problem. As shown in Figure 32, each user is associated with a proxy VM, which is considered to be a user's private VM that provides versatile computing and storage resources (located in a nearby fog node).

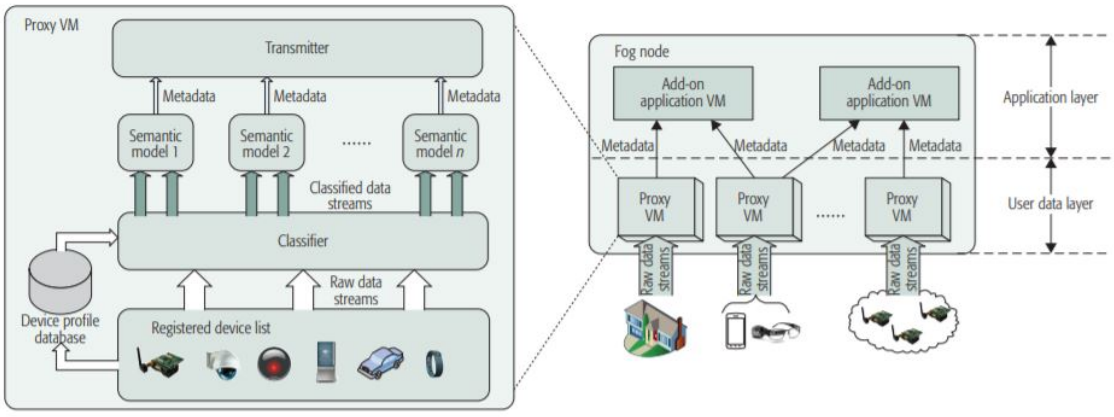

Figure 32: The hierarchical fog computing architecture [39]

This offers versatile tools for computation and storage. User-owned IoT devices are registered to the proxy VM of the user, which collects the raw Data streams created through a multi-interface BS from its registered devices are categorized into different groups based on the type of data. By analyzing the corresponding data sources, it produces the metadata and sends the metadata to the corresponding application VM. Notice that there is useful information in the metadata produced from the raw data streams Without compromising user privacy. For example, in the terrorist detection application, only the locations and timestamps of the matched photos/videos, rather than the original photos/videos, are uploaded to the application VM. The application VM, owned by the IoT service provider, provides a semantic model for each proxy VM to generate the metadata (e.g., the face matching algorithm in the terrorist detection application), it collects metadata from various proxy VMs and provides users with services.

For example, by analyzing the metadata from various proxy VMs, all terrorists will be detected, monitored, and arrested, thus safeguarding our society.

Proxy VM locations can be dynamic: if the registered devices are deployed statically (e.g. the sensors in the smart home), the proxy VM can be deployed statically in a nearby fog node, if any of the devices registered are mobile (for example, the mobile phone and wearable devices of a user travelling from home to work), as shown

in Figure 33.

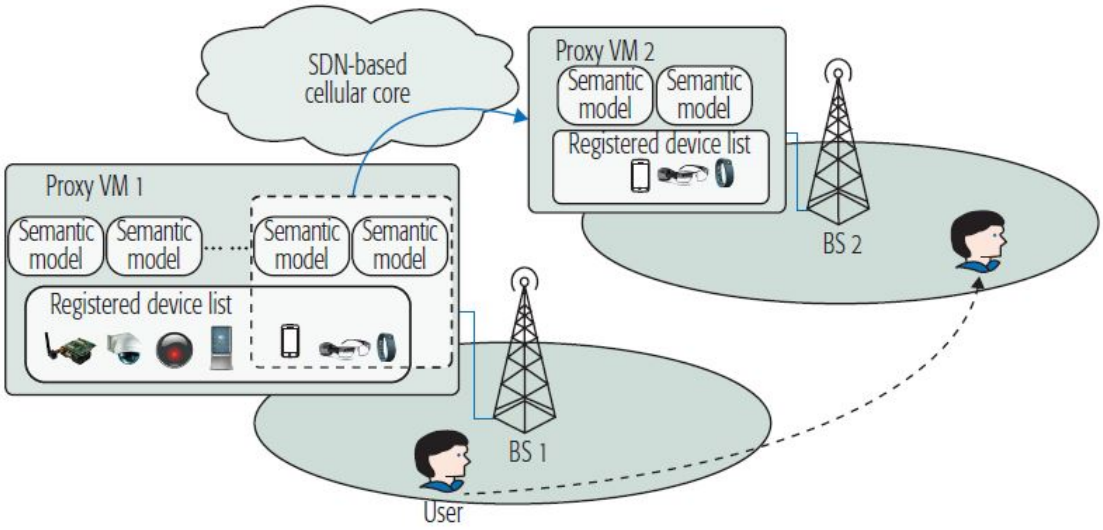

Figure 33: The illustration of the proxy VM decomposition and migration process [39]

The user's proxy VM can be broken down into two proxy VMs: the static IoT devices (in the home) continue to support one proxy VM, and as the mobile IoT devices roam away, the other proxy VM migrates to the other fog nodes.

The aim of migrating the proxy VM is to reduce traffic (i.e. uploading Cellular core network raw data streams from mobile devices to a proxy VM in the fog node) and the E2E latency between a user's mobile IoT devices and their proxy VM.

6.7 OpenFog Architecture

OpenFog is a hierarchical architecture for fog computing introduced by Open Fog Consortium. Figure 34 illustrates the locations of fog nodes in different deployment views.

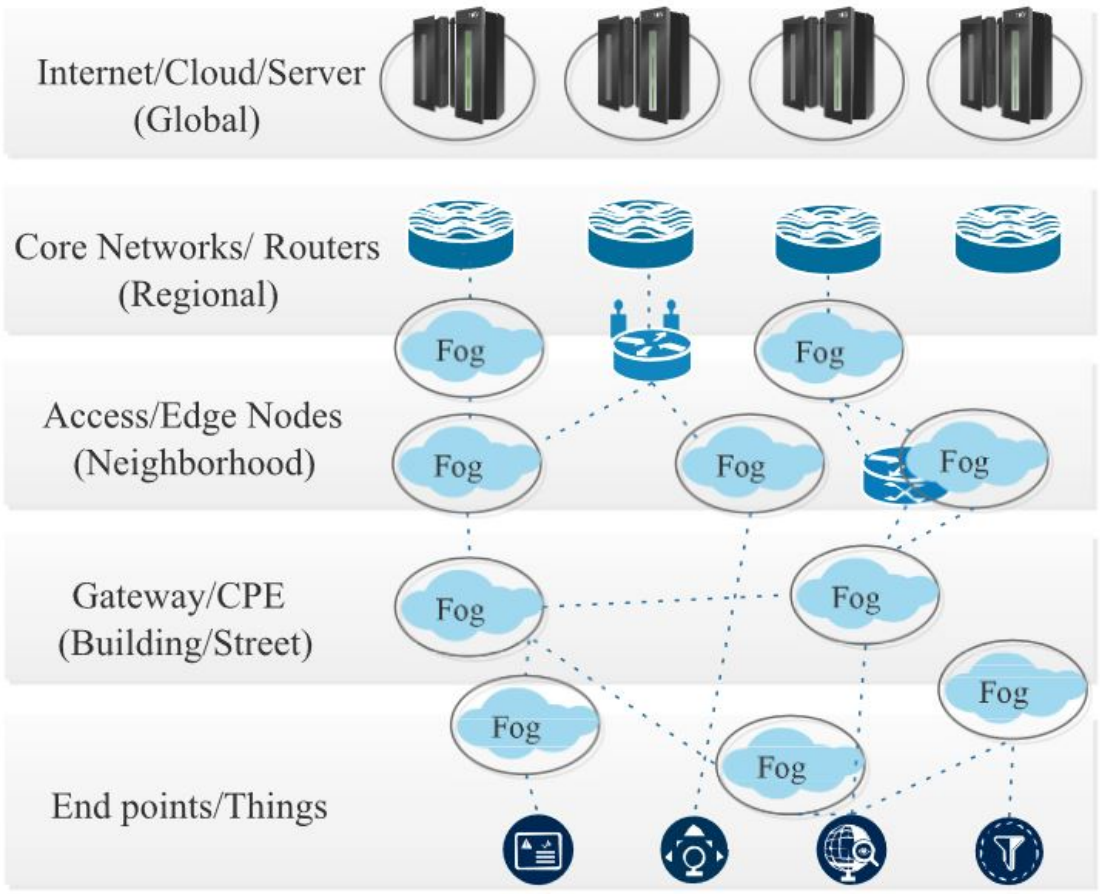

Figure 34: OpenFog Deployment Scenarios [4]

In multiple hierarchical orders, the fog layer covers all processing facilities between the endpoints and cloud servers, each of which can be deployed based on the data processing jobs' form, size, and latency specifications. Fog nodes can communicate through wired or wireless channels with each other. In several parameters, such as processing capacities, networking skills, and node reliability, each tier of fog nodes differs from other levels. To generate more intelligence, each tier sifts and extracts meaningful data. Data acquisition/collection and data normalization are done in the bottom tier of the architectural layer. Data filtering, compression, and transformation are done in the upper layer. The upper fog layer is close to the cloud which transforms the gathered data into a knowledge base for permanent storage. As the

layers are closer to the cloud, overall system intelligence and capability are improved. Fog nodes at the edge will architecturally need less processing, communication, and storage than nodes at high levels. However, Input and Output (I/O) accelerators needed to facilitate sensor data intake at the edge are far larger in aggregate than I/O accelerators built for higher-level nodes. As fog computing emerges, traditional centralized cloud computing continues to remain an essential component of computing systems.

A more detailed view of the OpenFog reference architecture is shown in Figure 35, in which four perspectives and three views are defined.

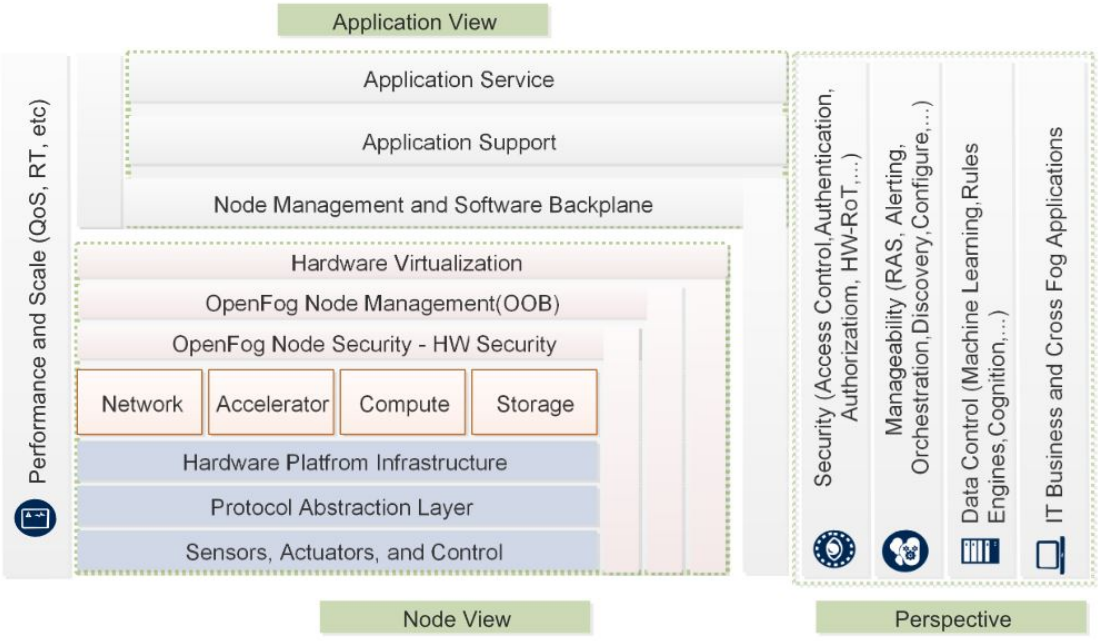

Figure 35: OpenFog layered architecture [4]

The perspectives cover the system's non-functional aspects, which include security, management, analytical and control, and fog business. There are Three views, software, node, and system views. The system view includes one or more node views coupled with other components to create a platform. The node view covers pieces and components from the system developer's point of view, while the system view deals with the interconnection of components and nodes and output from the point

of view of the system designer. Fog nodes need to serve many functions from the point of view of the node, including networking, computing, acceleration, storage, and managed sensors and actuators. The node requires to implement sufficient security mechanisms and management agents.

As with diverse heterogeneity of computing and networking, the abstraction layer must be implemented to provide standard APIs for connecting to other components of the framework. To provide additional computation throughput, the fog nodes involved in enhanced analytical need to configure accelerator modules such as graphical processing units, field-programmable gate arrays, and digital signal processors. Several forms of storage are used in fog nodes to meet the system's required reliability, stability, and data integrity. Furthermore, fog nodes can be linked to provide load balancing, stability, fault tolerance, data sharing, and cloud connectivity minimization in a mesh topology. The system software consists of three layers: application service, application support, node management, and software back-plane. Application support provides a wide variety of software that many apps (microservices) use and also share.

A multitude of computing clients or edge devices are used in the OpenFog architecture. This can work in conjunction with related cloud services to perform optimized storage, computing, networked communication, and related management tasks based on workload requirements. The following properties stand out in order to illustrate the difference between the OpenFog architecture and conventional cloud architectures. In particular, the architecture of OpenFog should be:

- Include lower latency storage at or near the deployment of the end-user and business.
- To avoid latency, network, and other migration costs (including bandwidth), perform the necessary computing near the end-user and data.
- Instead of having all communications to be routed and coordinated via the

backbone network, utilizing low latency communication at or near the end-user.

- Implement management elements at or near the endpoint, like network measurement , control, and configuration, rather than being solely managed by gateways like those in the LTE Core.

- Enable the results of telemetry and locally computed analytics to be copied to the backend cloud for further analytics and orchestration in a safe way.

An OpenFog Fabric consists of nodes or layers that can be distributed, centralized, or a mixture of them. It may rely and be implemented on dedicated hardware, software, or both. The common denominator is that this fabric distributes computing, communication, control, and storage resources and services through available devices, systems and clouds to achieve the desired purpose while fulfilling all application requirements.

It is not a binary decision to choose between a cloud and OpenFog. They form a continuum of mutually beneficial, inter-dependent. OpenFog architecture has a lot of unique advantages listed as follows:

- **Cognition:** OpenFog architecture brings awareness of client-centric objectives as an advantage which also provides autonomy.

- **Efficiency:** OpenFog architecture provides a pool of unused resources from devices which can be used to improve efficiency.

- **Agility:** OpenFog architecture provides on-demand scalability and rapid innovation of emerging technologies on the cloud infrastructure.

- **Latency:** OpenFog architecture provides real-time processing in the fog which results in lower latency.

Platform as a service (PaaS) is a cloud computing services category that offers a platform that allows consumers to create, run, and operate web applications without

the difficulty of constructing and maintaining the infrastructure usually associated with an application being built and launched.

The OpenFog architecture aims to define the necessary infrastructure to allow Fog as a Service (FaaS) to be designed to address certain categories of business challenges. The building blocks of the infrastructure and architecture in Figure 36 illustrate how FaaS can be activated and will be extended in the reference architecture.

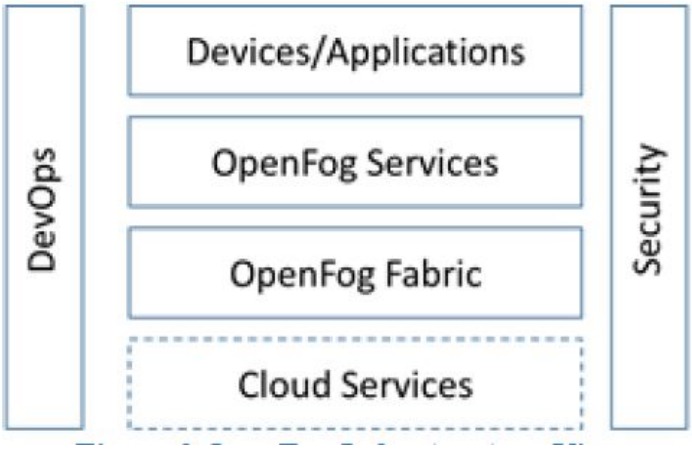

Figure 36: OpenFog Infrastructure View [40]

- **OpenFog Fabric:** It consists of building blocks that allow the creation of a homogeneous computer infrastructure that can provide useful services to the surrounding ecosystem (e.g. devices, protocol gateways, and other fog nodes). Generally, the homogeneous infrastructure is based on heterogeneous hardware and multi-vendor supplied platforms.

- **OpenFog Services:** These services are built on top of the fabric infrastructure which may result in network acceleration. There are a lot of these overlay services such as SDN, NFV, traffic offloading, complex event processing, and content delivery.

- **Devices/Applications:** Edge sensors, actuators, and applications that operate separately, within a fog deployment, or spanning fog deployments are devices

/ applications. The OpenFog service layer tackles this.

- **Cloud Services:** For analytical work that needs to run on a broader data scale or pre-processed edge data to create policies, Cloud Providers may take advantage of the cloud.

- **Security:** For OpenFog implementations, security is fundamental. Inside each architecture layer, discrete functionality units are wrapped with discretionary access control mechanisms so that the deployment of OpenFog and the surrounding ecosystem operate in a safe and secure environment. The OpenFog architecture would ensure that all data transfers between the endpoints involved are safeguarded by state-of-the-art information security practices.

- **Devops:** Automation allowed by an operationally efficient collection of standard DevOps processes and frameworks is powered by DevOps. OpenFog's DevOps drives the versatility of software updates and patching by continuously managed integration processes.

6.8 Cisco-Bonomi Reference Architecture

A reference architecture was introduced by Bonomiet et al. [41] which is depicted in Figure 37. There are five key components of the reference architecture, the heterogeneous physical resources layer, the fog abstraction layer, the orchestration layer of the fog service, IoT services, and the distributed message bus.

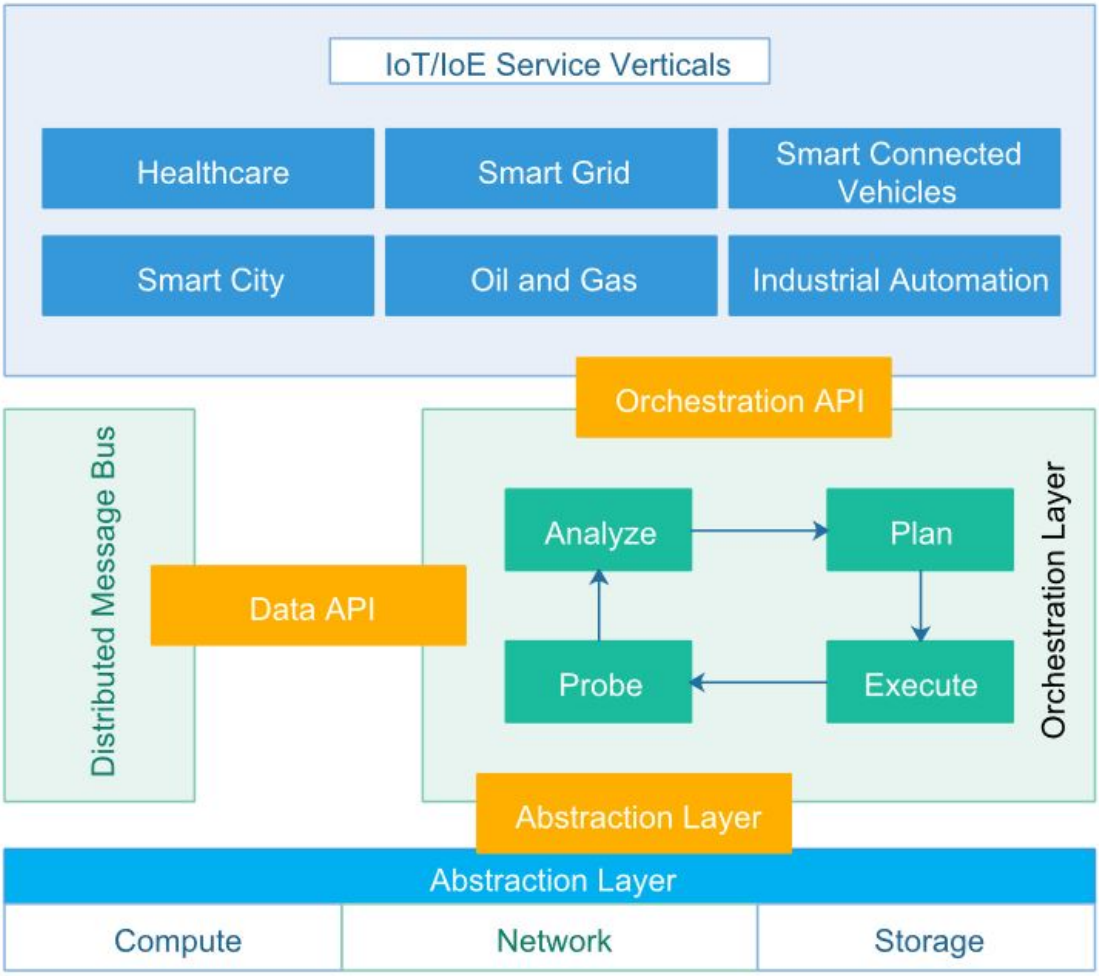

Figure 37: Cisco-Bonomi Reference Architecture [41]

- **The heterogeneous physical resources layer:** This layer consists of fog physical resources including servers, edge routers, access points, vehicles, sensors, and mobile phones.

- **The fog abstraction layer:** To allow smooth resource management and control through the higher layers, the fog abstraction layer is responsible for hiding the heterogeneous nature of the fog platform and providing a consistent and programmable interface. For monitoring, provisioning, and controlling physical and virtual resources, a collection of generic APIs are specified in this layer. Furthermore, for different components of the architecture, this layer defines

stability, privacy, and isolation policies.

- **The orchestration layer:** It is designed to control a fully distributed environment for fog computing and provides fog services with policy-based life-cycle management. This layer includes four key components, namely policy manager, life-cycle manager, capacity engine, and a distributed database containing business policies, state of fog nodes information, hardware and software capabilities of fog nodes. With a single global view and local implementation, it implements a distributed policy engine. Foglet is characterized as a software agent that extends the functionality of orchestration on edge devices. It uses abstraction layer APIs to track the health and state associated with the physical machine and its services that can be analyzed locally or globally.

- **The distributed message bus:** A scalable bus that is deployed to hold control messages for service orchestration and resource management is the distributed message bus.

Bonomi et al.[41] suggests that multi-tenancy is assisted by cloud and fog ecosystems. there are subtle differences in the nature of their client-organizations. Heterogeneous physical resources in a cloud are most commonly distributed in a centralized manner. Fog includes homogeneous resources in its distributed infrastructure-complements and extends the cloud to the edge and endpoints.

Fog infrastructure includes data centers, the core of the network, the edge of the network, and endpoints. Fog, as well as the cloud, facilitates the co-existence of applications from various tenants. Each tenant stipulates its topology and it is assigned a virtual topology. Three major resource groups are in the fog, namely computing, storage, and networking. In the above areas, the fog needs scalable virtualization that can be obtained by a Virtual File System, a Virtual Block, and a suitable Network Virtualization infrastructure. The Cloud profits from a policy-based provisioning process.

Similarly, to automatically control resources, Fog uses a policy-based orchestration and provisioning framework on top of the resource virtualization layer.

The issue of seamless resource management through a variety of platforms should be resolved by Fog. Fog platform hosts a various set of applications belonging to various verticals–smart connected vehicles to smart cities, industrial automation, to name but a few. The architecture offers data and control APIs that many applications can use. Data APIs were used to access the fog data store while control APIs were enabled to decide how to deploy an application.

The infrastructure of the Fog network is heterogeneous, ranging from high-speed connections between corporate data centers and the core to different wireless connectivity technologies. The fog abstraction layer defines the ability to run multiple service containers on a physical machine to maximize the usage of resources and provides policies for stability, privacy, and isolation. The service orchestration layer offers the ability to use multiple Foglet agents to handle services on a broad volume of Fog nodes with a wide variety of capabilities. The health and state of physical machines are tracked by these Foglets' abstraction APIs. The Bonomi platform enjoys the messaging bus to hold service orchestration, resource management, and distributed database control messages that are perfect for increasing the scalability and fault tolerance of fog.

6.9 Fog architectures for 5G and IoV

Control and data planes are closely connected in legacy networks, protocols running in switches and routers are immutable once they are installed. The network is now deeply ossified and can hardly be innovated by the operator. SDN, a modern network paradigm that allows the separation of control data Operation, provides the controller, via programmable control plane, with a global view of the network.

Therefore, it leads to the dynamic implementation of networks, agile network management, quicker innovation of software, and effective use of resources. In this section,

an SDN-enabled architecture for cloud-fog interoperation is discussed [42].

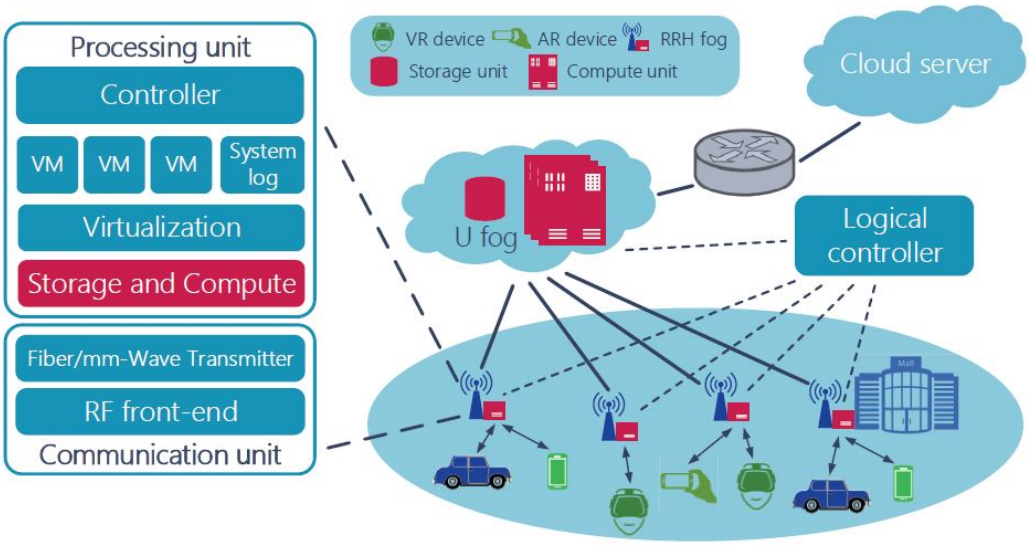

Figure 38: The SDN enabled cloud-fog interoperation architecture [42]

- **Components:** As highlighted in Figure 38, mobile users, fog servers, and the cloud server are included in the proposed architecture, where storage and processing services beyond the access network are referred to as cloud. On top of the cloud-RAN (C-RAN) fogs are deployed. C-RAN is a great carrier for Deploying fog, the role of traditional base stations is split into two parts:

  – remote radio head (RRH) for radio signal transceiving.

  – baseband unit (BBU) for high-speed baseband processing.

  C-RAN can well hold the facilities for fog computing by stacking storage and computation tools on BBU and RRHs. RRH fogs are widely distributed and linked via fiber link or millimeter-wave to the centralized BBU fog (20-40 km away [43]). In addition to the communication unit, a processing unit is fitted for each RRH fog. The tools for storage and computation are virtualized as isolated virtual machines (VMs), which is administered by a local fog controller. The

functionality of the RRH fog lies in two folds:

  - it performs baseband processing after radio signals are received.
  - it offers local storage and computing services that can be scheduled dynamically by the BBU fog controller.

  With more processing units, BBU fog is stacked, making it the most potent fog inside RAN. In addition to the features of RRH fog, the controller of the BBU also functions as a master controller (MC) in which all controllers in RRHs are synchronized. Notice that this configuration is compatible with the legacy C-RAN, and can therefore be implemented incrementally.

- **Interoperation:** Local network information is continuously exchanged among controllers with MC coordination, leading to a logic controller with group intelligence.Via either fog-fog interoperation or cloud-fog interoperation, the controllers analyze user requests and respond accordingly.

  Interoperation with fog-fog is initiated when user demands are beyond an individual fog capacity. The logical controller begins a crowdsourcing process upon receiving such requests. The MC decomposes the task based on the available resources on each fog server and distributes them accordingly. Once the decomposed activities have all been finished, the MC reassembles the findings and introduces them to mobile users.

  When user requirements can not be met only by fog computing, cloud-fog interoperation is enabled. In this regard, the MC abstracts the user requirement and determines whether fog preprocessing can speed up the provisioning of the service. Then the MC starts a process of crowdsourcing through fog-fog Interoperation or the direct sending of user requests to the cloud.

  With interoperation, more user requests can be managed locally, and the utilization of fog services is driven to the limit in the meantime. Taking fog caching

as an instance, it is possible to push frequently requested files to cloud fogs. Thus, via fog-fog interoperation, mobile users can fetch content from neighboring servers. Compared to the cloud, as fog storage is limited it is only possible to archive selected content and to continually erase dated data. Thus, to achieve a higher local request hit ratio, cloud and fog have to communicate frequently to maintain the optimality of fog content entry.[42]

6.10 Components of Fog Computing Architecture

The architecture of Fog computing consists of multiple layers. Figure 39 displays each layer and its components. Based on their functionality, which is described as the layer, the components are divided into several classes. These features allow IoT devices to connect with multiple Fog devices, servers, gateways, and the cloud.

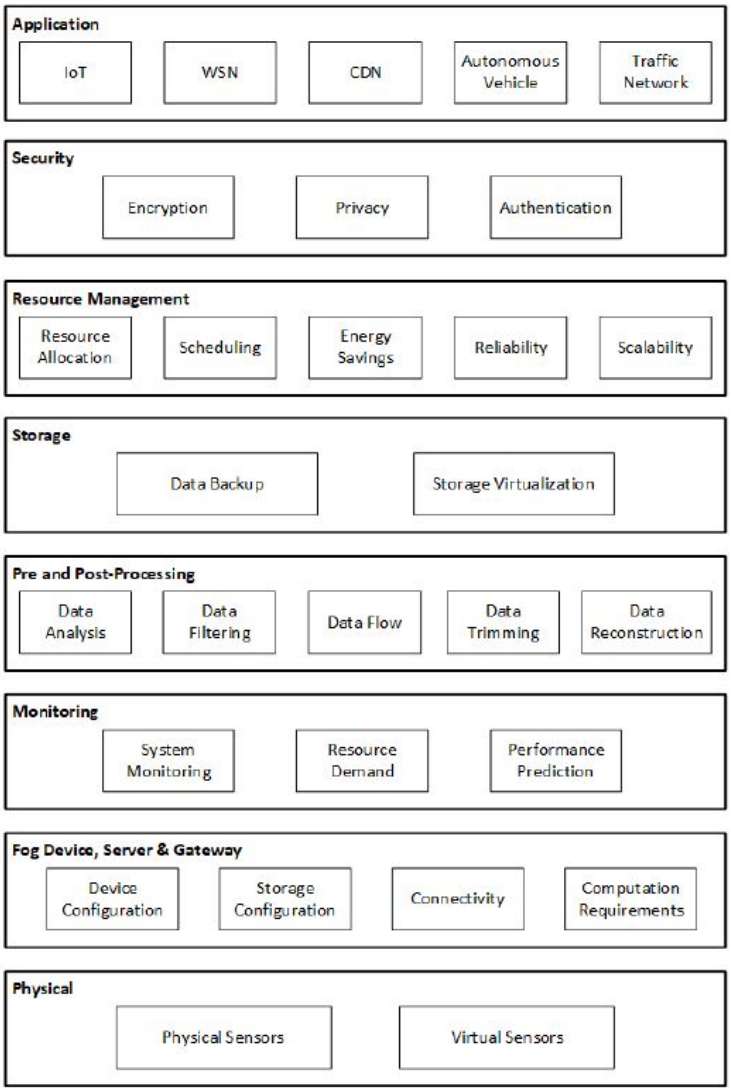

Figure 39: Components of Fog Computing Architecture [37]

6.11 Physical Layer

The different data types emitted by the sensors are the basic data source for Fog computing. Smart devices, temperature sensors, humidity sensors, smart homes, the CCTV surveillance system, the traffic monitoring system, self-driving cars, and so on could produce this data. For example, if we want to introduce a smart traffic management and monitoring system, from different sensors, roadside devices, and cameras, we need to get updated traffic conditions for all roads, which will help

control traffic signals. Future traffic demand must also be forecast by collecting data from several GPS sensors. The role of virtual sensors is also important in addition to physical sensors, if a road accident happened, it would not be possible to determine if the road should be blocked or traffic should continue to go using only a single sensor. This event can affect the road with one or more lanes, while another lane may allow the traffic to proceed, because of this incident, however, the traffic handling capacity will be limited.

In this case, a virtual sensor could help to obtain an immediate decision on road conditions, multiplexing of traffic, and redirection of traffic. The physical layer is therefore made up of physical and virtual sensors, where any system for generating data could fall into any of these classes.[37]

6.12 Fog Device, Server, and Gateway Layer

The Fog device, Fog server, or Fog gateway could be a standalone device or an IoT device [44]. It is clear, however, that the Fog server should have a higher configuration As it handles many Fog devices than the Fog system and gateway. Various factors are involved so that the Fog server can run. These include its function, the configuration of hardware, networking, the number of devices it can handle, etc. Whether the Fog server is separate from an IoT device or part of it is specified by its function. The Fog system will be connected to a group of physical and virtual sensors. Similarly, it will be connected to a Fog server by a group of Fog devices. Compared to the Fog system, the Fog server should have higher processing and storage capacity in this case.

A particular cluster of Fog devices that are linked to the same server may be able to Communicate as needed with each other. In the smart transportation use case, some application processing might depend on other Fog clusters. The Fog server and device layer are responsible for managing and maintaining information on hardware configuration, storage configuration, and connectivity of devices and servers. The computation specifications requested by different applications are also handled by this

layer. Requirements for computation depend on the flow of data and The number of IoT devices that are connected to the Fog system and the total number of Fog devices that are connected to the Fog server.[37]

6.13 Monitoring Layer

The monitoring layer still monitors the output and resources of the system, facilities, and responses. Components of device monitoring help select the necessary resources during service. In smart transportation system scenarios, various apps run. Therefore, it is evident that when resource availability is negative for computation or storage on a Fog computer, a situation could occur. It might happen to the Fog server in a similar situation. To counter those styles The Fog system and servers will request support from other peers in situations. The device monitoring aspect will therefore help effectively determine certain items. The portion of resource demand tracks current demand for resources and can forecast future demand for resources based on the current usage of resources and user activities. In this way, the device would be able to cope with any uncomfortable circumstances where there might be a resource outage [45].

Fog computing performance based on device load and resource availability can be predicted by performance prediction monitors. In service level agreements, this portion is necessary to maintain acceptable QoS specifications. If SLA violation happens often, because of the penalty, the cost of the device for the provider will be increased. Although performance prediction will not completely remove this problem, by predicting the performance and use of the system, it will be able to minimize overall SLA violations.[37]

6.14 Pre and Post-Processing Layer

There are various elements in this layer, which primarily operate on basic and advanced data analysis. At this level, acquired data and data trimming are analyzed

and filtered, reconstruction and restoration are often undertaken where appropriate. After processing the data, the portion of the data flow decides whether the information needs to be saved locally or sent to the cloud for long-term storage [46], this phenomenon is referred to as stream processing.

In the case of the smart transportation system, data from several sensors will be produced. In order to gain insight into the produced data, the generated data will be analyzed and filtered in real-time.

Processing data at the edge and minimizing the amount of data that needs to be stored is the key challenge in Fog computing. There could be no use for all the generated data. In certain instances, storing all the data generated would not even be a reasonable idea. For example, if data is produced from a sensor every second, the mean value of the data can be stored within a minute or an hour, depending on the sensor on criteria for submission.[37]

6.15 Storage Layer

It is the responsibility of the storage module to store data by storage virtualization. The component of the data backup ensures data availability and mitigates data loss. A pool of storage devices linked by a network functions as a single storage device in the storage virtualization concept, which is easier to manage and maintain.

One of the major advantages of storage virtualization is to use less-expensive storage or commodity hardware to provide enterprise-class features. In order to minimize the complexity of the storage structure, the storage layer thus enables storage virtualization. During system service, storage in a system can fail at any point. Therefore, to avoid any unnecessary circumstances, it is necessary to backup essential data. Periodic or personalized data backup systems take care of the data backup module in this layer [37].

6.16 Resource Management Layer

The elements in this layer maintain resource distribution and scheduling and deal with problems of energy saving. The reliability factor ensures the reliability of the scheduling of applications and the allocation of resources. Scalability guarantees the scalability of Fog assets during peak hours when demand for resources is high. The cloud tackles horizontal scalability, while Fog Computing attempts to have horizontal and vertical scalability [46].

Many distributed network, processing, and storage device resources are available. This is a crucial problem for distributed resources using the application processing. Therefore, in which the resource allocation aspect handles and retains resource allocation related problems, resource allocation, deallocation, and reallocation can occur. Another critical problem is that several programs can run concurrently in the Fog computing environment. Therefore, proper scheduling of these apps is needed. The application planning aspect takes care of these applications based on different goals.

This layer also has components that save energy, which manage resources in an energy-efficient way. Energy efficiency also has a positive environmental effect and helps to minimize running costs. Based on different reliability indicators and metrics, reliability components manage the requirement for a system's reliability. Fog computing is a dynamic system that requires both IoT devices, Fog devices, and the cloud to be taken care of. A system or link may therefore fail at any stage, so reliability management is an essential issue [37].

6.17 Security Layer

The components in this layer, which also protect the privacy of Fog users, will maintain all security-related issues such as contact encryption and secure data storage. Fog computing, like cloud computing, is intended to be implemented as a type of utility computing. However, the user connects to the cloud in the cloud computing concept, for services, however, the user will link to the Fog infrastructure for the services in

the Fog computing concept, while the Fog middleware will handle and maintain cloud communications.

A consumer who wishes to connect to a service must be approved by the provider. The authentication component in the security layer processes user authentication requests so that they can connect to the Fog computing service environment[47]. Maintaining encryption between communications is necessary to maintain security, so that security breaches by outsiders do not occur. The encryption component encrypts all IoT device connections from and to Towards the Cloud. Components of Fog computing are often linked via a wireless network, so security issues are important.

6.18 Application Layer

While Fog was designed to serve IoT applications, Fog computing is also supported by many other Wireless Sensor Network (WSN) and CDN-based applications. Fog computing would be able to take advantage of any application that has latency-aware characteristics. This implies some kind of utility-based service that could fit into Fog computing by offering a better quality of service and cost-efficiency. For example, Augmented Reality-based applications should adopt Fog computing because of its nature. It is obvious that Augmented Reality will change the modern world in the near future. Fog computing can meet the needs of real-time processing for Augmented Reality applications, which can maintain continuous enhancement of Augmented Reality-related services [37].

7 Fog computing Applications

fog computing is applied in different fields and industries. In this section, a comprehensive review of fog computing applications is introduced.

7.1 Healthcare

Leveraging the fast advancements in information and communication technology, electronic healthcare (e-healthcare) emerged itself as a revolutionary new paradigm. E-healthcare quickly swaps the means of traditional healthcare after technological improvements. IoT plays a key role in redefining e-healthcare as the Internet of Healthcare Things (IoHT) in this ever-changing healthcare scenario, where both individuals and computers connect, communicate, gather, and share data through the incorporation of physical objects, hardware, software, and computer devices. Connecting the digital world to the physical world, IoHT helps healthcare devices (e.g. Fitbits, sensors, Bluetooth, mobile devices, etc.) to capture health-related data with the aid of widespread and ubiquitous computing and e-healthcare systems (e.g., blood oxygen saturation, blood pressure, weight, glucose level, respiratory, heart rate, etc).

There are other reported works on IoT-based healthcare include: emergency medical service, smart rehabilitation system, do-it-yourself solution focusing on patient-oriented infrastructure development, smart hospital system, anomaly detection, body sensor network-based healthcare system, cardiac arrhythmia management system, and self-aware early warning system.

However, a large number of heterogeneous, multidimensional, and multimodal data databases are created by the various players in the e-healthcare ecosystem, which is a major challenge. In order to process this enormous amount of healthcare data, systems with enormous storage and processing capacity are required that can analyze big data, therefore cloud computing was used. IoHT, big data, and cloud computing needed to converge to build the IoHT ecosystem to form the next generation of e-healthcare systems.

Nonetheless, due to network overloading, the current number of IoHT devices causes rising latency, thereby reducing the system suitability for real-time applications. In order to address this issue, fog computing is used. However, the architecture

of fog computing may be susceptible to the single point of failure, since it relies more on the gateway device. Mist computing can be used to build an optimized network that bridges IoHT devices to the virtual computing environment to further increase response time by reducing data traffic on fog nodes in local networks, thus reducing response latency and improving IoHT system efficiency and lifetime.

Different IoHT ecosystem data types and applications need different processing and response times. Addressing these problems, a heterogeneous five-layer mist, fog, and cloud-based IoHT architecture are proposed in [48] which is capable of managing and routing real-time and offline/batch mode data efficiently.

Therefore, the proposed IoHT architecture is capable of choosing suitable data transfer policies to reduce latency based on disparate data sources.

- Ensuring optimum utilization of resources by delegation and distribution Layer processes with relatively lower loads.

- Guaranteeing minimum delay in transmission by proper load balancing; Assuring the most favorable distribution of data-sensitive resources for prioritized data transmission.

The following sections describe the various components of the framework.

Ecosystem

An ecosystem is made up of different devices and applications that ensures uninterrupted data flow. In fact, ecosystem is vital for successful design of an IoHT framework. In the following, an IoHT ecosystem is illustrated in Figure 40.

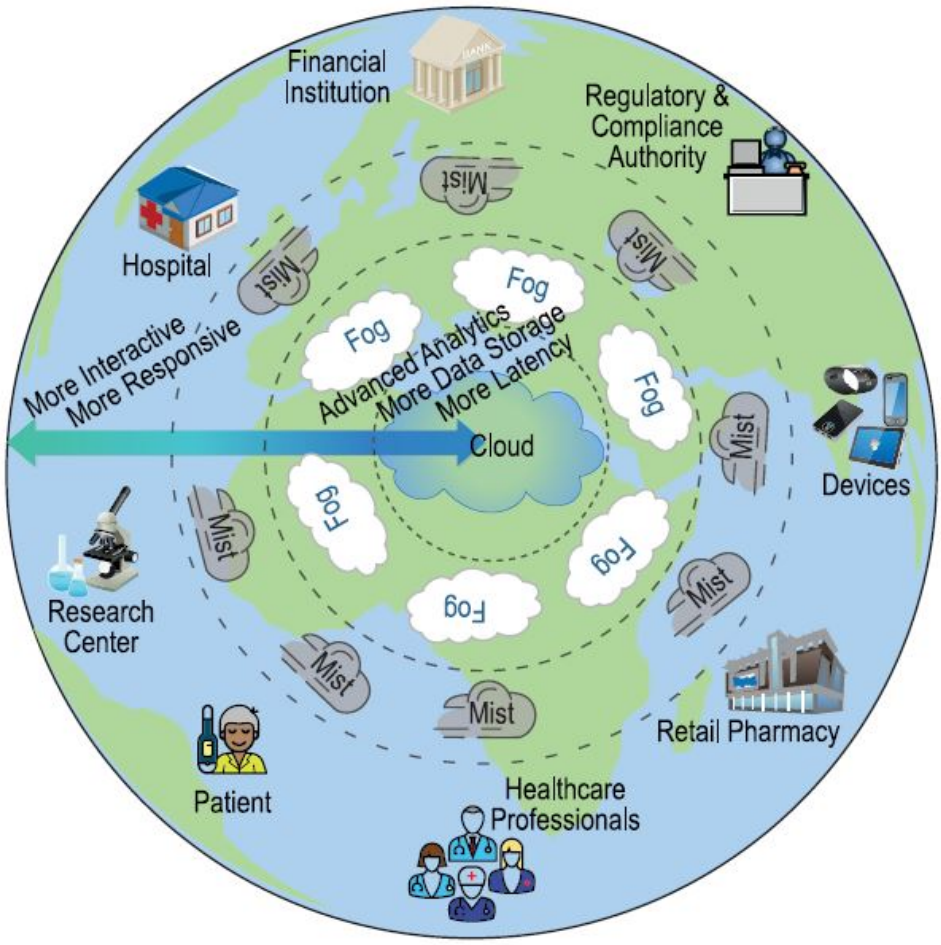

Figure 40: Overview of an IoHT Ecosystem [48]

As the ecosystem diagram shows, different stakeholders in IoHT who reside at The outer circle (e.g., healthcare organizations and professionals, patients, applications, and information systems) link to the inner circles aimed at smooth sharing of information with their respective counterparts. The outer circle with very few analytical capabilities is the most interactive and sensitive one. The analytical capabilities, along with latency and data storage, are growing steadily towards the inner circles. So, to ensure delay tolerant data transmission of real-time data as well as big data, the proposed architecture adopts appropriate layer-specific data transmission polices.

Network Architecture

Figure 41 shows the five layers of the IoHT framework's architecture. The five layers are the perception layer, mist layer, fog layer, cloud layer, and application layer.

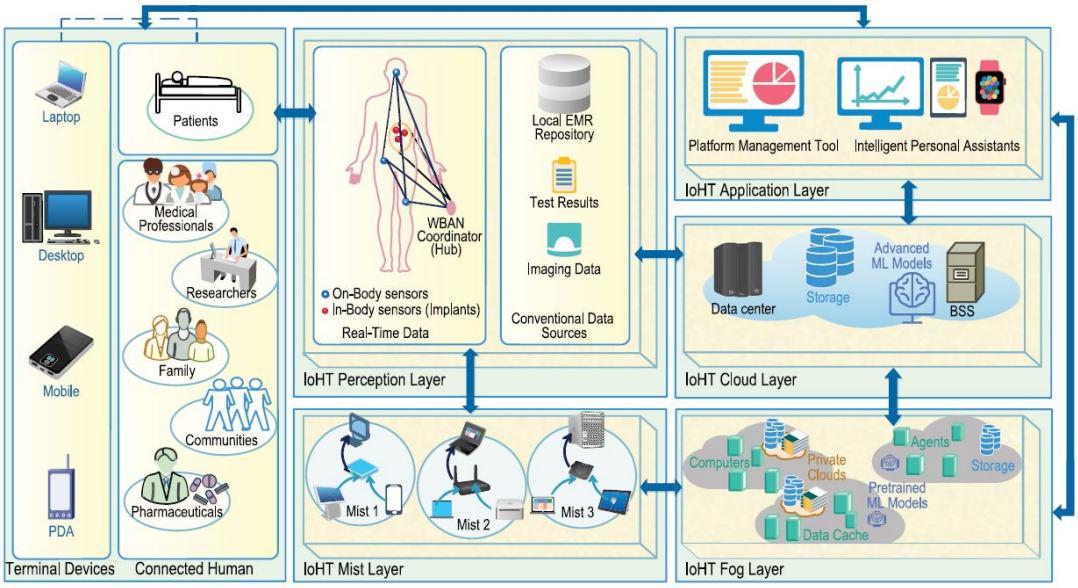

Figure 41: Architecture of the proposed IoHT framework [48]

Each of these layers was developed with predefined functionalities related to the data transmission and processing pipeline of the IoHT framework. Figure 42 illustrates a block diagram with the functionality of individual layers.

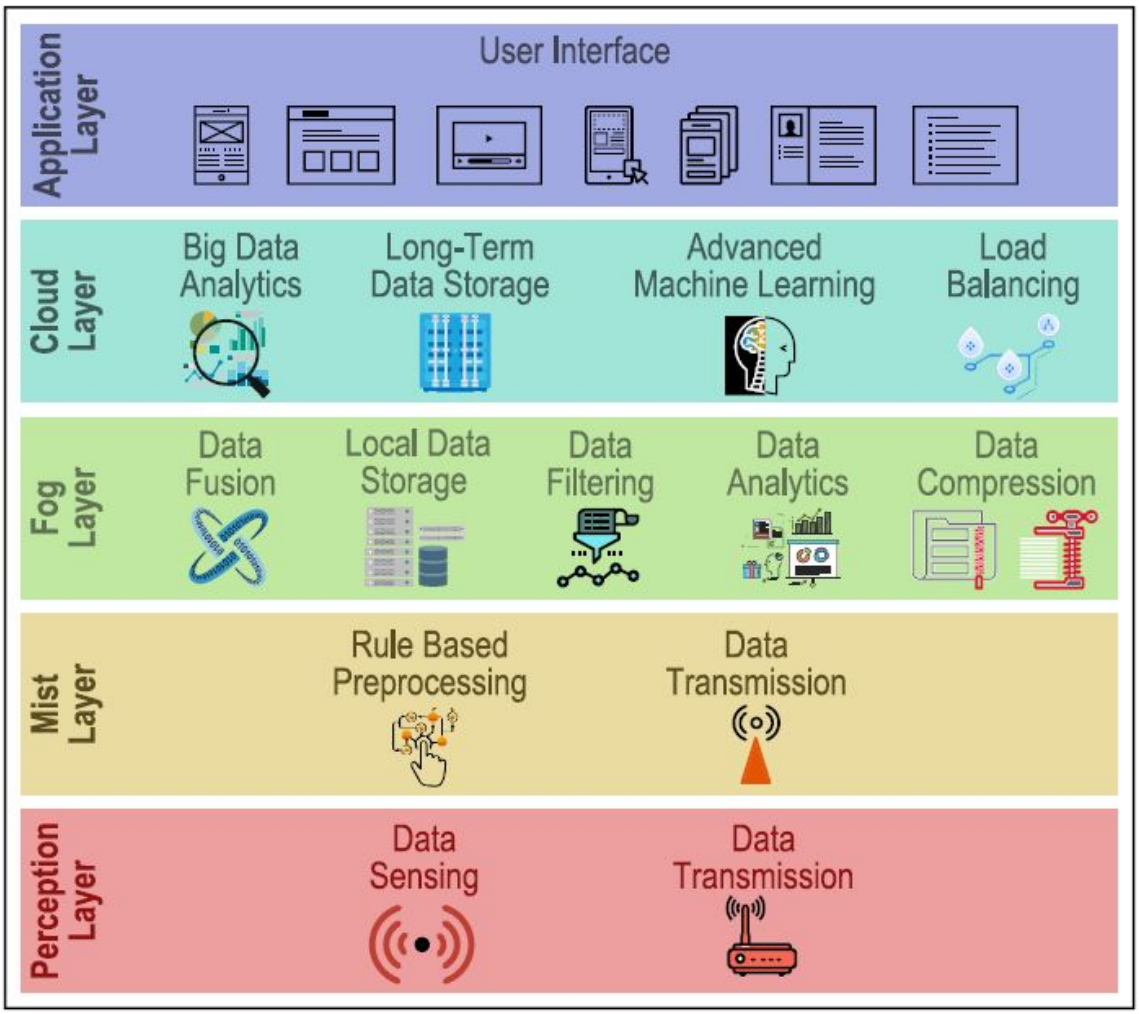

Figure 42: IoHT Framework [48]

- **Perception Layer:** The lowest layer of the proposed IoHT framework is the perception layer. This layer is responsible for distinguishing physical objects and collecting contextual Data from devices that produce real-time as well as non-real-time data and healthcare data. The data is primarily calculated by small sensors, embedded systems, RFID tags and readers from individuals and their environments, small to large diagnostic and healthcare devices, medical and clinical imaging devices, and any devices allowed for data acquisition and transmission.

  Big health data [e.g., standardised eHR, electronic medical record (eMR), clin-

ical/medical imaging data, unstructured clinical reports, etc.] are available in addition to real-time healthcare data, Requiring separate handling due to their need for advanced data analytics. In the IOHT framework, both types of healthcare data are transmitted to a specific overlaying layers either mist or fog based on the data type and their processing requirements.

- **Mist Layer:** This layer has been introduced To facilitate time-sensitive data processing. Mist computing locates directly within the network fabric which functions on the extreme edge of it. The main responsibility of this layer is performing basic processing of the sensor data (e.g., data aggregation, fusion, and filtering). Mist computing provides optimal resource utilization of the Things.

- **Fog Layer:** One of the key driving forces behind IoT technology growth is the need to process "on the fly" data to detect anomalies, provide real-time warnings, and automatically trigger appropriate actions which need a system with high responsiveness and minimal latency. Due to their high latency, using centralized cloud-based models is inadequate for this objective. In such cases, it is necessary to decentralize and delegate processing loads to different layers based on the demand of the application.

  A decentralized architectural pattern for getting computing resources and application services closer to the edge is formed by the fog layer, thus reducing the latency of response. The functional components in this layer provides, local data storage , data filtering, data compression, data fusion, and intermediate data analytics to minimize cloud disposable load, boost device efficiency and QoS, and save backbone bandwidth

- **Cloud Layer:** The cloud layer can connect to the perception layer, fog layer, and application layer. Aggregated healthcare data from the fog layer are sent to the cloud layer for advanced analytics and long-term storage of big data. Also,

data from non-sensor sources, such as eMR, eHR, e-prescription platforms, etc. are integrated seamlessly. this layer performs various advanced data analytics including, machine learning, data mining, rule-based processing, and automated reasoning based algorithms to accomplish extracting meaningful insights from the heterogeneous healthcare data, However, it can boost device efficiency at this layer by delegating suitable processing loads to the fog layer and using the cloud layer for computationally costly operations.

- **Application Layer:** The top-most layer of IoHT architecture is the application layer. It offers user interfaces between stakeholders/consumers from the IoHT and the Framework itself to represent directly the economic and social benefits generated. Various healthcare applications are distributed to the appropriate stakeholders through these user interfaces. This layer also offers access rights and privileges directly to the healthcare application developers and users for related services from the cloud or fog layer.

7.2 game analytics

With the growing success of Massively Multiplayer Online Gaming (MMOG) and the rapid growth of mobile gaming, cloud gaming shows great promises over the traditional MMOG gaming model as it frees players from hardware and game installation requirements on their local computers. However, as graphics rendering is transferred to the cloud, the transmission of data between end-users and the cloud significantly increases the latency of response and restricts device coverage, thereby preventing cloud gaming from achieving high user quality of service (QoS).

Previous research has proposed deploying more datacenters to address this issue, but it comes at a prohibitive cost. A lightweight framework called CloudFog was introduced which includes "fog" consisting of supernodes responsible for rendering and streaming game videos to their nearby players. Fog allows the cloud to only be responsible for intensive game state computation and transmitting supernode update

information, which greatly reduces traffic, thus reducing latency and bandwidth usage. To further enhance QoS, it is proposed in [49] the reputation-based supernode selection strategy to allocate a suitable supernode to each player that can provide a satisfactory game video streaming service, the receiver-driven encoding rate adaptation strategy to increase the continuity of playback, the social network-based server assignment strategy to prevent contact between servers in data interaction.

The response latency is not seriously affected by uploading from the players to the cloud and downstream latency is a significant factor for QoS, which is affected by the video streaming rate of the game. Therefore to reduce the downstream latency by reducing the traffic transmitted from the cloud it is proposed a new design that streamed game videos from nearby supernodes to players, instead of from remote game servers.

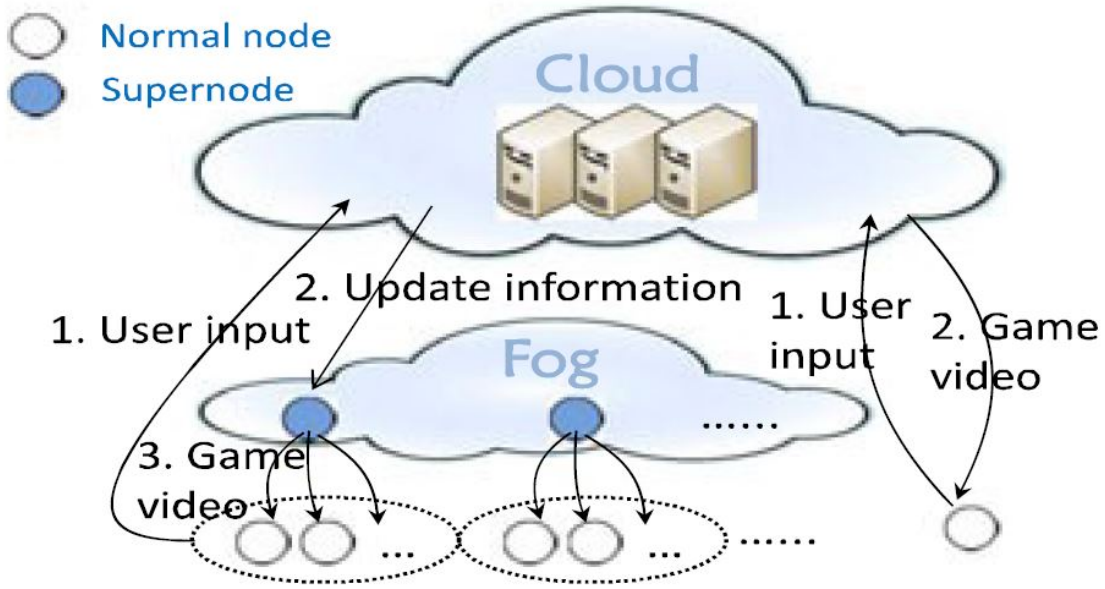

Figure 43: Fog-assisted cloud gaming infrastructure [49]

Since MMOG's computing of a virtual environment has a very high demand for server capabilities, this role is the responsibility of the cloud. Figure 43 demonstrates the fog-assisted infrastructure for cloud gaming. The fog consists of supernodes and

normal nodes that are connecting to their nearby supernodes. The normal nodes that cannot find nearby supernodes are directly connected to the cloud.

When each supernode is initially deployed, it is pre-installed with the game client. When a regular acts (e.g. initiating a strike or moving to a new location) during gameplay, this information is sent to the cloud server. The server gathers action data from all system players involved and performs the calculation of the virtual world's new game state (including the new shape and location of objects and avatar states). The cloud then sends the updated data to the super-node, which accordingly updates its virtual world.

Then makes a video of the game based on viewing location and angle. Finally, encrypt the video of the game and stream it. As a player is close to its network distance supernode, and cloud traffic is greatly reduced, the delay in game video transmission is much shorter than that of directly uploading game video from the cloud than in existing cloud computing systems.

7.3 Video Analytics

Edge computing has shown its ability to reduce response timing, lower bandwidth usage, and improve energy efficiency along with the trend of moving computation from the network core to the edge where the most data is produced. At the same time, video analytics with low latency is becoming increasingly relevant for public safety, counter-terrorism, self-driving vehicle applications. Since these activities are either computational or bandwidth-intensive, edge computing suits well with its ability to use computation and bandwidth from and between each layer flexibly. A framework (LAVEA) [12] built on top of an edge computing platform is proposed in [50] that discharges computing between customers and edge nodes, collaborates with nearby edge nodes to provide video analytics of low latency at locations closer to users.

[12]Latency-aware Video Analytics on Edge Computing Platform

LAVEA System Design Goals and Overview

- **Latency:** One of the basic criteria of edge computing device design is the ability to provide low latency services.
- **Flexibility:** Edge computing system should be able to flexibly utilize the hierarchical resources from client nodes, nearby edge nodes, and remote cloud nodes.
- **Edge-First:** By edge-first, it means that the edge computing platform is the first choice of the computation offloading target.

LAVEA is inherently an edge computing platform that supports video processing with slow latency. Edge computing nodes and edge clients are the key components. If a client is running tasks and the nearby edge computing node is available, it is possible to determine whether to run a task locally or remotely. Figure 44 illustrates the architecture of LAVEA.

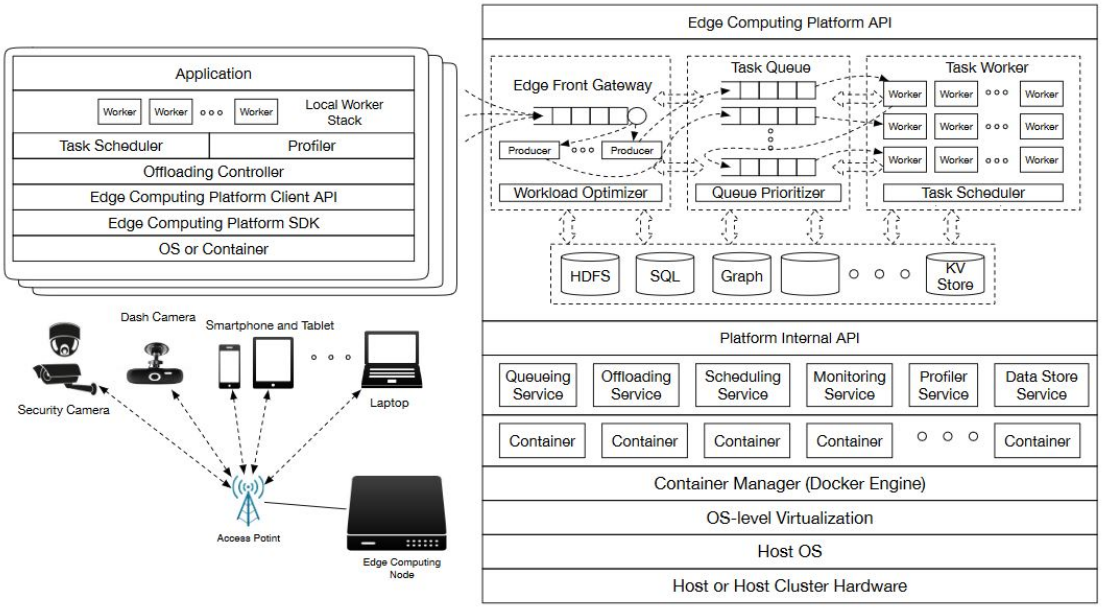

Figure 44: The Architecture of Edge Computing platform for LAVEA [50]

- **Edge Computing Node:** The edge computing node in LAVEA offers edge

computing services to nearby mobile devices. The edge computing node is called the edge-front, connected to the same access point or base station as the clients. it ensures that edge computing services can be as pervasive as Internet connectivity by installing edge computing nodes with an access point or base station. Multiple edge computing nodes will function together, and the edge front still functions as the master and communicates with other edge nodes and cloud nodes.

As shown in Figure 44, the lightweight virtualization strategy is used to provide various clients with resource allocation and isolation. Via client APIs, any client may send tasks to the platform. The framework is responsible for shaping the workload, controlling queue preferences, and planning tasks. These functions are implemented via multiple micro-services provided by internal APIs, such as queueing service, scheduling service, data store service, etc.

- **Edge Client:** Since most edge clients are either resource-constrained machines or require requests from a large number of clients to be fulfilled, an edge client normally performs lightweight data processing tasks locally and discharges heavy tasks to the nearby edge computing node. The edge client in LAVEA has a thin client architecture to ensure that all customers can run it without adding too much overhead.

  There is only one worker for low-end devices to make progress on the job assigned. The profiler and the offloading controller are the most critical part of the client node architecture, serving as participants in the corresponding profiler service and service offloading. A client may provide offloading information to the edge-front node with the profiler and offloading controller and fulfill the received offloading decision

7.4 Image and face recognition

Recognition and perception-based mobile applications that are time-sensitive, such as image recognition, are increasing in recent years. These applications recognize the surroundings of users and augment them with information and media. They are latency-sensitive. On the one side, the execution is usually offloaded to the cloud given the compute-intensive nature of the tasks performed by such applications.

On the other hand, network latency is incurred by offloading such applications to the cloud, which can increase the user-perceived latency. Consequently, to minimize latency, edge computing has been proposed to allow devices to discharge intensive tasks to edge servers instead of the cloud. A different model for using edge servers was proposed For recognition applications such as web caching, Edge Computing, they used the edge as a specialized cache and formulates the predicted latency for such a cache. It demonstrates that using an edge server, such as a traditional web cache, will lead to higher latencies for recognition applications.

It proposes [51] Cachier, a framework that uses the caching model together with new latency optimizations to reduce latency by adaptively balancing load between the edge and the cloud, using spatiotemporal locality of requests, using offline application analysis, and estimating network conditions online. For image-recognition applications, it evaluates Cachier and demonstrates that techniques yield 3x speed in responsiveness and work accurately over a variety of operating conditions.

Mobile Image Recognition

Recognition programs are essentially a kind of information retrieval system, similar to systems such as the WWW for document retrieval. To retrieve the relevant content, the code needs to identify the object in the request for which the user requires content and then use the object's identifier. As shown in Figure 45, the algorithm operates in a pipeline manner inside the cloud.

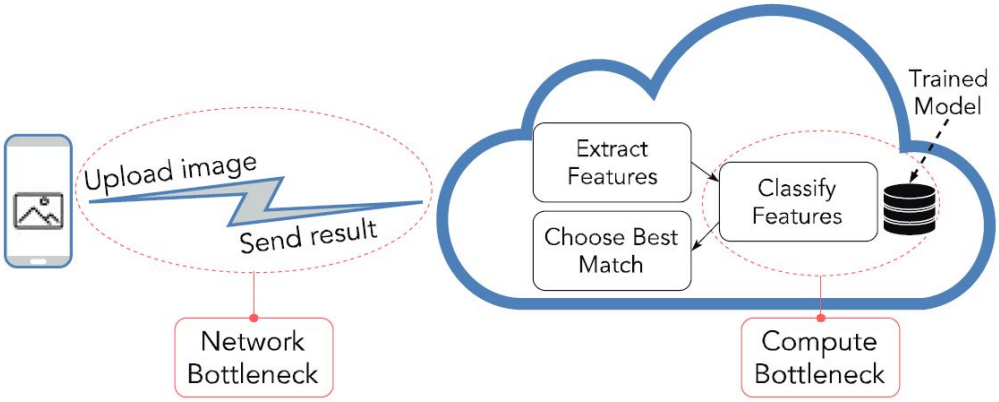

Figure 45: Typical architecture for mobile image-recognition applications [51]

- **Extract features:** From the input image, a series of features are extracted. The characteristics are numerical vectors describing the image. The category of features that are usually used is called local features, such as corners and edges, which are extracted from "interesting" points in the picture.

- **Classify and match features:** First, using a trained model, these extracted features are classified. Using previously extracted characteristics from training images of all possible objects that can be queried have made this model offline. The more objects one wants to recognize, the bigger this model will be.

- **Choose the best match:** Once the closest characteristics are identified, the object with the most characteristics matches the request image and is selected as the recognized object. A threshold of a minimum number of matches is set to ensure accuracy. This can be followed by geometric inspection to confirm that the features are correctly arranged in space in the request image. This also helps to identify the object in the request image [52].

These stages are usually carried out in the cloud for mobile image-recognition applications. The image is captured by the mobile device and uploaded over cellular networks and sent over the Internet to the cloud, as seen in Figure 45. In the cloud,

the image-recognition methods are then carried out and the response is returned to the computer. Usually, these responses are in some type of knowledge or material, such as annotation strings explaining what the consumer sees (e.g. identification of painting), or media to be overlaid on top of recognized items (e.g. augmented reality).

Total latency can be broken down into two key factors in such an architecture, (1) network latency, and (2) compute time.

- **Network latency:** Applications experience this latency because, for each recognition request, they need to upload large quantities of data to the cloud, over the Internet. If the "distance" is reduced between the computing entity, currently the cloud, and the mobile device, this delay can be reduced.

- **Compute time:** Algorithms for image recognition are compute-intensive. the key contributor to latency is the classification of request-image attributes Using the trained model. Besides, the scale of this model has a direct effect on latency, which is the number of trained items. The processing time for a large number of objects can lead to high latency and thus a poor user experience.

To tackle these latency issues as mentioned before in [51], the idea of caching at the edge is introduced which leverages the compute resources available at the edge, and creates an image-recognition (IR) cache.

7.5 Fog Computing in Connected Vehicles

One of the possible scenarios for fog, such as the integration of fog computing with traditional vehicle ad hoc networks (VANET) to form the Internet of Vehicles (IoV) or vehicle fog computing, is vehicle-related applications. An architecture that considers vehicles as smart devices that are mobile and fitted with multiple sensors and have the capacity to collect user traffic information from computing and communication is proposed in this article. Both intra-vehicle sensors and the environment will

collect the data. Fog nodes can be installed at the edge of vehicular networks in this architecture to make data collection more efficient, along with real-time processing, organization, and storage of traffic data.

Figure 46 illustrates three main layers namely, the smart vehicles which collect data, the roadside units/fog nodes as the fog layer, and the cloud servers as the cloud layer

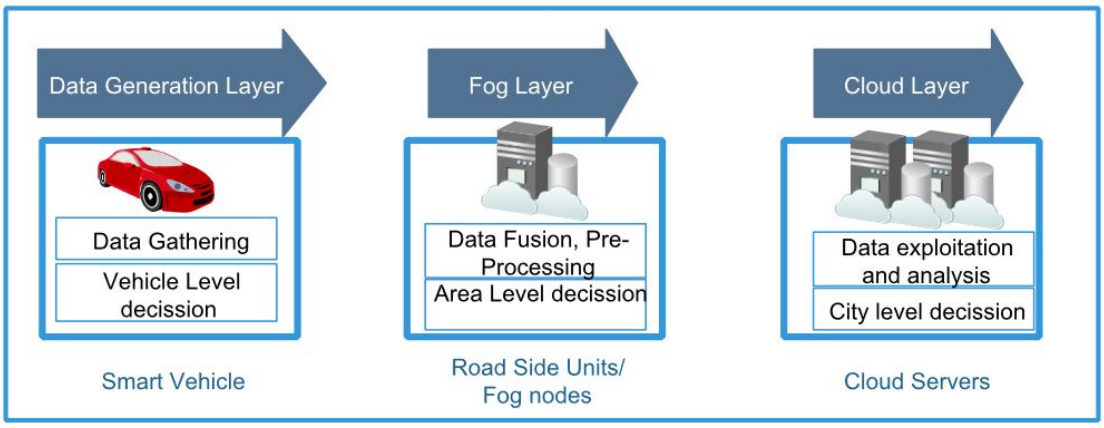

Figure 46: Fog computing platform for connected vehicle applications [53]

smart vehicles have a critical role as data sources in a vehicular fog computing system because of their real-time processing, sensing (e.g. cameras, radars, and GPS), connectivity, and storage capabilities. It has been calculated that the amount of data generated by the different sensors in a smart vehicle is about 25 GB / h in a single day. To make real-time decisions, some of these data will be processed by the smart vehicle, while the rest of the data will be exchanged and uploaded to the fog nodes for further analysis, such as traffic control planning. In different places in the city can be deployed by the roadside device as a fog node that allows the platform to process collected data and send it to cloud servers.

In a vehicular fog computing system, this feature can be extended as a middleware system that links cloud servers to smart vehicles. These units/nodes will have more functionality and provide more diverse services for smart vehicles, such as navigation,

video streaming, and smart traffic lights, as compared to existing vehicle networks. Cloud servers allow monitoring at the city level, permanent storage of data, and play a role as a centralized control system. To make globally optimal decisions, these servers will receive the data from all fog nodes. For example, to achieve optimum city-level traffic control, they would track, maintain, and control the city's road traffic infrastructure [53].

7.6 Security and Privacy Architecture

Networked applications cover a wide variety of privacy-sensitive and mission-critical use cases, including private information transfer (such as images, medical reports), routine everyday activities (such as shopping, transportation), and management of business resources (such as supply chains). Deployment of these applications requires Security. Therefore, Security in a multi-tier computing environment involves some challenges that need to be identified and addressed. Several enabling technologies are used, such as wireless networks, distributed and peer-to-peer systems, and platforms for virtualization. This diversity of technology involves both the safety of all these elements and the orchestration of various security measures to preserve the integrity of the ecosystem.

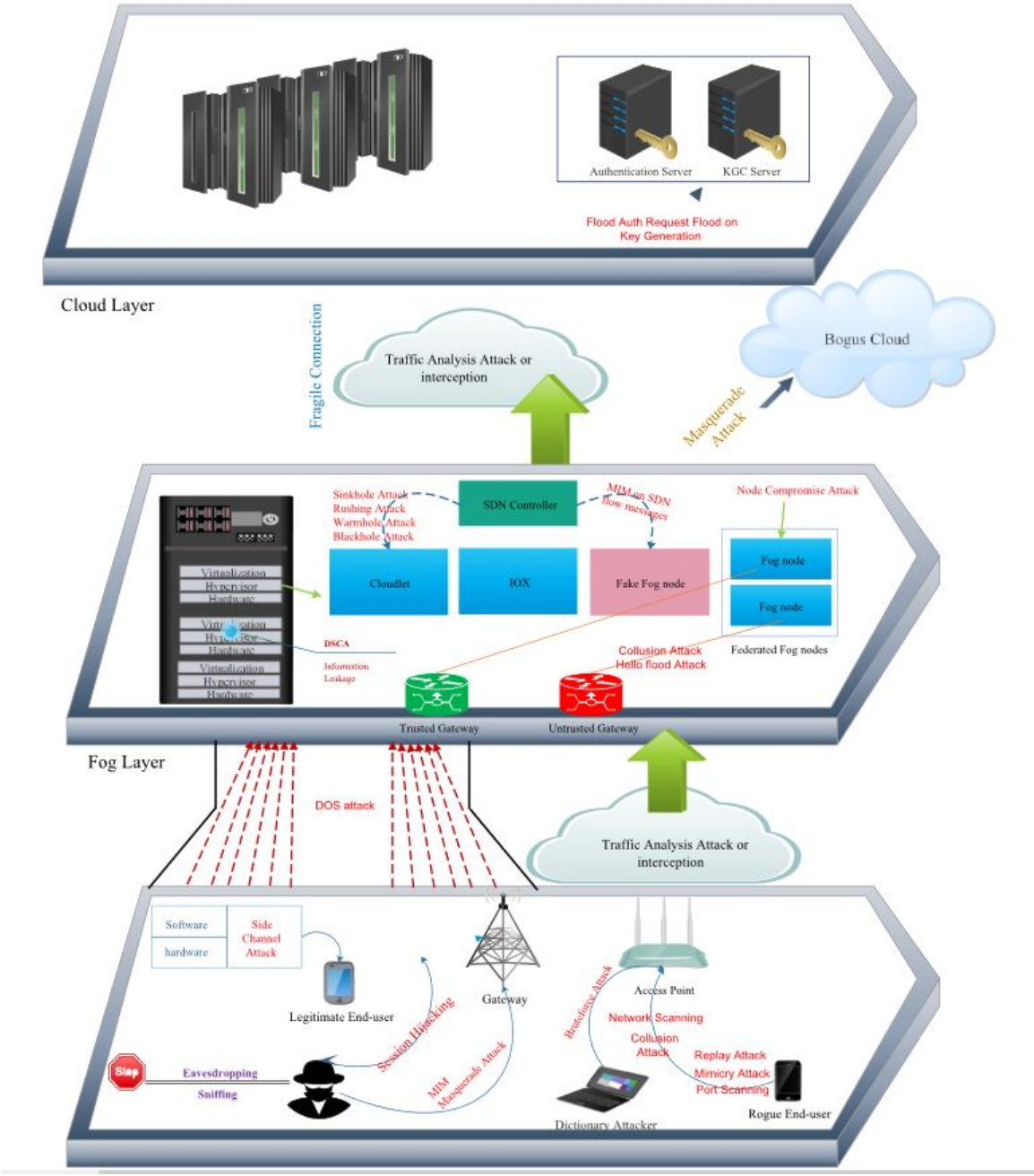

Figure 47: A layered security architecture for fog computing [53]

The three-layer model for fog security is illustrated in Figure 47. The flow of requests and data in the system is illustrated and possible attacks are visualized in each layer. Fog-access is the point of connection to the end-user devices responsible for end-device management and access control in the fog ecosystem. Fog-computing requires all the computing nodes in the fog layer, including access node computing.

In this layer, vulnerabilities associated with device integrity and availability and data confidentiality are handled. The fog-cloud interconnection layer addresses possible threats in the interconnection of both subsystems.

7.7 Access Layer Security

The access layer deals with the authentication, authorization, access control, and data security of network-connecting edge devices in a widely distributed set-up. This layer addresses specific issues in a fog ecosystem such as the decentralized and distributed nature of edge paradigms, interoperability, and mobility support, and location awareness are considered.

7.8 Fog Layer Security

The computing layer includes all the nodes involved in sub-cloud computing. There may also be access layer functionality for a computing node. Fog nodes, for example, can be located at the edge of networks and may communicate in a distributed manner with each other.

8 Challenges and future direction in Fog Computing

Fog computing is considered to be the promising extension of the Cloud computing paradigm to deal with IoT-related network edge problems. However, computational nodes are heterogeneous and distributed in Fog computing. Security assurance is also predominant in Fog computing. Analyzing the features of Fog computing from structural, service-oriented and security perspectives, the challenges in this field can be listed as follows [54]:

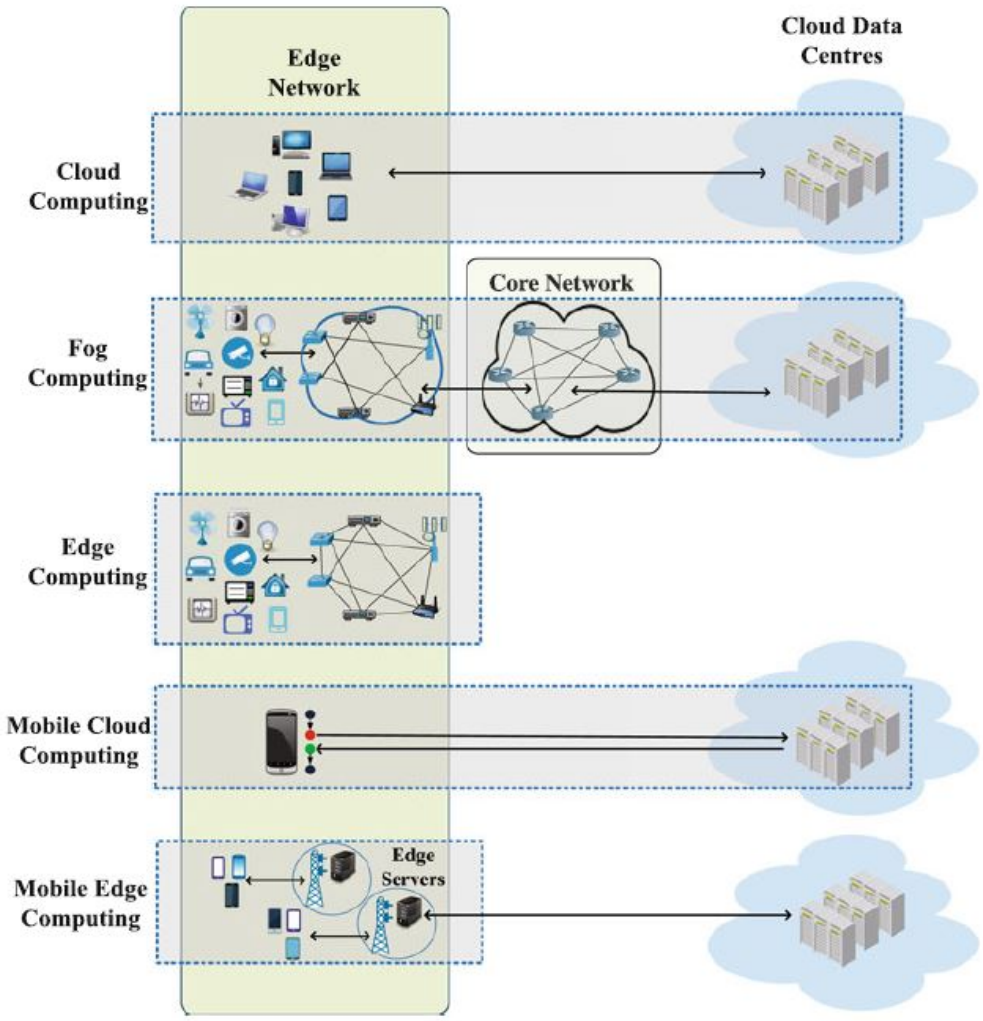

Figure 48: Computation domain of Cloud, Fog, Edge, Mobile Cloud and Mobile Edge computing [54]

- **Structural issues**

  – Various components can be used as possible Fog computing infrastructure from both the edge and core network. These modules are usually fitted with different types of processors, but they are not used for general computing purposes. In addition to their traditional activities, it will be very difficult to provide the components with general-purpose measurement. The selection of appropriate nodes, corresponding resource configuration, and deployment places are also essential in Fog, based on operational re-

quirements and execution environment.

  - In Fog computing, computational nodes are spread around the edge network and can be virtualized or shared. In this situation, it is important to identify appropriate techniques, metrics, etc. for inter-nodal collaboration and effective resource provisioning.

- **Service-oriented**

  - The development of large-scale applications in resource-constrained nodes is not very simple compared to conventional datacentres since all nodes are not resource enriched.
  - It is required to define policies for the distribution of computational tasks and resources between IoT devices/sensors, Fog, and Cloud infrastructures. It is also difficult to design data visualization via web-interfaces in Fog computing.
  - The Service Level Agreement (SLA) in Fog computing is often impacted by many variables, such as service costs, energy consumption, application functionality, data flow, network status etc. Therefore, it is quite difficult to specify the service provisioning metrics and corresponding Service Level Objectives (SLOs) in a particular scenario. Besides, retaining the basic QoS of the Fog nodes for which they are designed is highly required.

- **Security aspects**

  - Since Fog Computing is built on traditional components of the network, it is extremely vulnerable to security attacks.
  - In a largely distributed paradigm like Fog computing, authenticated access to services and privacy maintenance are difficult to ensure.
  - Implementation of data-centric integrity security mechanisms can have a significant effect on the QoS of Fog computing.

8.1 Fog Security and Privacy

Since fog computing is proposed in the context of the Internet of Things (IoT) and originates from cloud computing, fog computing inherits cloud security and privacy concerns. Although some problems can be solved using existing technologies, due to the distinct characteristics of fog computing, other problems are posing new challenges, such as heterogeneity in fog nodes and fog networks, mobility support requirements, large geo-distributed nodes, position awareness, and low latency. Several security and privacy issues in fog computing are discussed in this section [55].

8.2 Trust and Authentication

Data centers are normally operated by cloud service providers for the deployment of cloud computing. However, due to various implementation options, fog services providers may be different parties:

- Internet service providers or wireless carriers, who have control of home gateways or cellular base stations, may build fog with their existing infrastructures.
- Fog infrastructure is designed for the situations where cloud providers would like to extend their services to the edge of the network.
- End-users who own a private local cloud and wish to minimize ownership costs would like to convert the private local cloud into a fog and lease spare resources on the private local cloud.

  This flexibility complicates the trust situation of fog.

Rogue Fog Node

A rogue fog node would be a fog device or fog instance that pretends to be legitimate and coaxes end users to connect to it. For instance, a fog administrator may be allowed to handle fog instances in an insider attack but may instantiate a rogue fog instance instead of a legitimate one. The feasibility of a man-in-the-middle attack in fog

computing is demonstrated in this article [56]. The authors show when the gateway is replaced by a rouge one, the adversary can eavesdrop, collect, and manipulate all the incoming and outgoing traffic between the cloud and end-users. The presence of fake fog nodes would jeopardize the security and privacy of user data. Due to many factors, this issue is difficult to solve in fog computing.

- complex trust situation calls for different trust management schemes.
- Dynamic development, in-stance virtual machine deletion, makes it difficult to maintain a blacklist of rogue nodes. A measurement-based approach that allows a client to avoid rogue access point (AP) link has been suggested by Han et al.[57] Their technique uses the round-trip time between end-users and the DNS server to detect client-side rogue APs.

Authentication

Authentication is an essential problem for fog computing security since services are provided by front fog nodes to massive-scale end users. The key security problem of fog computing has been considered by Stojmenovic et al.[56] as authentication at various fog node levels. Traditional authentication based on PKI is not successful and has low scalability. An inexpensive, reliable, and user-friendly solution to the authentication problem in local ad-hoc wireless networks was proposed by Balfanz et al.[58], based on physical contact for pre-authentication in a location-limited channel. In the case of Cloudlet, NFC can also be used to simplify the authentication process. It will be useful to apply biometric-based authentication in fog computing as biometric authentication evolves in mobile computing and cloud computing, such as fingerprint authentication, face authentication, touch-based or keystroke-based authentication, etc.

8.3 Network Security

Wireless network security is a major concern for fog networking, due to the predominance of wireless in fog networking such as Jamming attacks, sniffer attacks. Normally, we have to trust a network administrator's manually generated configurations in the network and isolate network management traffic from regular data traffic. Fog nodes, however, are deployed at the edge of the Internet, definitely putting a heavy burden on network management, imagining the cost of maintaining massive cloud servers that are distributed across the edge of the network without easy maintenance access. In many aspects of fog computing, employing SDN can facilitate implementation and management, increase network scalability and reduce costs. SDN technique in fog computing allows fog networking security new challenges and opportunities which are listed here:

- **Network Monitoring and Intrusion Detection System (IDS)**: CloudWatch [32] can leverage OpenFlow [21] to route traffic for security monitoring applications or IDS.

- **Traffic Isolation and Prioritization**: Traffic isolation and prioritization may be used to avoid network congestion or shared resources such as CPU or disk I/O from being dominated by an attack.

- SDN leverages VLAN ID(tag) to isolate traffic in the VLAN group and separate malicious traffic.

- **Network Resource Access Control:** Klaedtke et al [59] has proposed an access control scheme on a SDN controller based on OpenFlow,

- **Network Sharing:** If the network sharing with guests is carefully configured with security considerations, fog-enhanced routers in the home network may be opened to guests. This article [60]has suggested OpenWiFi, in which guest

WiFi authentication is transferred to the cloud to determine guest identity; guest access is given independently, and accounting is enforced to delegate guest accountability.

8.4 Secure Data Storage

User data is outsourced in fog computing, and user control over data is transferred to the fog node, which introduces the same security threats as in cloud computing. First, because the outsourced data might be lost or wrongly changed, it is difficult to maintain data integrity. Second, the data uploaded may be exploited for other interests by unauthorized parties.

An auditable data storage service has been proposed to counter these risks. To protect data in the context of cloud computing. To provide integrity, confidentiality, and authentication capability for the cloud storage system, techniques such as homomorphic encryption and searchable encryption are combined to allow a client to verify the data stored on untrusted servers. Want et al. [61] have suggested public privacy auditing for cloud-based data relying on a third-party auditor (TPA) using homomorphic authenticator and random mask technique to safeguard TPA privacy. Prior storage systems use erasure codes or network coding to deal with data corruption detection and data repair to ensure data storage reliability, While Cao et al.[62] has suggested an LT code scheme that offers lower storage costs, much faster retrieval of data, and comparable communication costs.

There are new problems in fog computing in the design of secure storage systems to achieve low latency, enable the dynamic operation, and deal with interplay Between Cloud and Fog.

8.5 Secure and Private Data Computation

Another critical problem in fog computing is to provide security and privacy-preserving computation outsourced to fog nodes.

Verifiable Computing

Verifiable [13] computing provides a computing device to offload the functional computation to other perhaps untrusted servers while maintaining verifiable results. The other servers evaluate the function and return the result with proof that the function has been correctly computed. In fog computing, the fog user should be able to verify the correctness of the computation to install confidence in the computation offloaded to the fog node. Below are some methods to fulfilled verifiable computing.

A verifiable computing protocol has been proposed by Gennaro et al.[13] that enables the server to return a computer-sound, non-interactive proof that can be verified by the client. The protocol can provide input and output privacy (at no additional cost) For the client in such a way that no input and output information is learned by the server. A framework named *Pinocchio* has been developed by Parno and Gentry so that the client can check general server computations while relying only on cryptographic assumptions. The client produces a public assessment key with Pinocchio to explain its computation, and the server then tests the computation and uses the assessment key to produce proof of correctness [64].

Data Search

Sensitive data from end-users must be encrypted before outsourced to the fog node to protect data privacy, making efficient data utilization services difficult. One of the key services is a keyword search, i.e., keyword search among encrypted data files. Several searchable encryption schemes have been developed by researchers that allow a user to securely search encrypted data via keywords without decryption. In this article [65], the authors proposed the first encrypted data search scheme ever to include proven secrecy for encryption, query isolation, managed search, and support of hidden query.

[13]The term verifiable computing was formalized in [63]

8.6 Privacy

The deliberate spreading of private information, such as data, location or usage, are gaining attention when end users are using services like cloud computing, wireless network, IoT. There are some challenges to address such privacy in fog computing, since fog nodes are in the vicinity of end-users, and can capture more sensitive data than the remote cloud lying in the core network.

8.7 Data Privacy

Privacy-preserving algorithms can run between the fog and the cloud in the fog network, although those algorithms are typically resource-prohibited on the end devices. In general, the fog node at the edge collects sensitive data produced by sensors and end devices. Techniques such as homomorphic encryption can be used to enable privacy-preserving aggregation without decryption at local gateways [66].

8.8 Usage Privacy

The usage pattern in which a fog client uses the fog services is another privacy problem. For instance, In a smart grid, reading the smart meter will reveal a lot of household information, such as when there is no person at home and when the TV is turned on which violates the privacy of the user. While privacy-preserving mechanisms have been suggested in smart metering, due to the absence of a trusted third party (i.e. a smart meter in a smart grid) or no counterpart system such as a battery, they cannot be directly implemented in fog computing. The fog node can quickly obtain end-user consumption statistics.

The fog client generates dummy tasks and offloads them to multiple fog nodes, hiding its real tasks among the dummy ones which is a possible naive solution. This approach would, however, increase the payment of the fog customer, as well as waste resources and energy. Another option would be to build a smart way to partition the framework to make sure that private information is not exposed by the offloaded

resource usages.

8.9 Location Privacy

In fog computing, the privacy of the location refers to the privacy of the fog clients' location. The fog node, to which the tasks are offloaded, will assume that the fog client is closer and farther from other nodes, as a fog client normally offloads its tasks to the nearest fog node. Besides, if a fog client uses multiple fog services at multiple locations, assuming the fog nodes collude, it may report its route trajectory to the fog nodes. The location privacy of the person or the object is at risk as long as such a fog client is attached to a person or a significant object. If a fog client always selects its nearest fog server strictly, the fog node will certainly understand that the fog client that uses its computing resources is nearby. The only way to protect the privacy of the location is by obfuscating identity so that the fog client does not recognize the fog client even if the fog node knows that a fog client is nearby.

There are several identity obfuscation techniques; in [67], for example, the authors use a trusted third party to create fake IDs for each end-user. A fog customer does not necessarily select the nearest fog node but selects one of the fog nodes that it can achieve according to certain parameters, such as latency, reputation, load balance, etc. The fog node can only understand the rough position of the fog client, but can not do so precisely. Once the fog client uses computing resources from multiple fog nodes in an area, however, its position can boil down to a small region, as its location must be at the intersection of the covers of multiple fog nodes.

9 Conclusion

For many sectors, including manufacturing, electricity, transport, smart cities, education, retail, healthcare, and government, the Internet of Things (IoT) accelerates digital transformation and provides a lot of advantages. The number of linked devices and IoT networks is rapidly growing due to the fundamental benefits of IoT for differ-

ent businesses and industries. Edge/Fog Computing was introduced as a promising solution to handle billions of connected devices and big data that are generated by IoT, which are often security-critical and time-sensitive.

In this project, we discussed cloud computing, Edge and Fog computing, and Mist computing and how they relate to other computing paradigms, such as cloudlets, Mobile Edge Computing (MEC), and Mobile Cloud Computing (MCC). Next, we presented a taxonomy of fog computing research topics and summarized the state-of-the-art in all these computing paradigms. We posed challenges and future directions for all paradigms.

Bibliography

[1] G. Davis, "2020: Life with 50 billion connected devices," in *2018 IEEE International Conference on Consumer Electronics (ICCE)*, 2018.

[2] T. G. Rodrigues, K. Suto, H. Nishiyama, and N. Kato, "Hybrid method for minimizing service delay in edge cloud computing through vm migration and transmission power control," *IEEE Transactions on Computers*, vol. 66, no. 5, pp. 810–819, 2017.

[3] P. Mell, T. Grance, *et al.*, "The nist definition of cloud computing," 2011.

[4] P. Habibi, M. Farhoudi, S. Kazemian, S. Khorsandi, and A. Leon-Garcia, "Fog computing: A comprehensive architectural survey," *IEEE Access*, vol. 8, pp. 69 105–69 133, 2020. DOI: 10.1109/ACCESS.2020.2983253.

[5] A. Yousefpour, C. Fung, T. Nguyen, K. Kadiyala, F. Jalali, A. Niakanlahiji, J. Kong, and J. P. Jue, "All one needs to know about fog computing and related edge computing paradigms: A complete survey," *Journal of Systems Architecture*, 2019.

[6] R. Buyya, J. Broberg, and A. M. Goscinski, *Cloud computing: Principles and paradigms*. John Wiley & Sons, 2010, vol. 87.

[7] M. M. Mosbah, H. Soliman, and M. A. El-Nasr, "Current services in cloud computing: A survey," *arXiv preprint arXiv:1311.3319*, 2013.

[8] N. Chohan, C. Bunch, S. Pang, C. Krintz, N. Mostafa, S. Soman, and R. Wolski, "Appscale: Scalable and open appengine application development and deployment," in *Cloud Computing*, D. R. Avresky, M. Diaz, A. Bode, B. Ciciani, and E. Dekel, Eds., Berlin, Heidelberg: Springer Berlin Heidelberg, 2010, pp. 57–70, ISBN: 978-3-642-12636-9.

[9] S. A. Ahson and M. Ilyas, *Cloud computing and software services: theory and techniques*. CRC Press, 2010.

[10] F. Liu, J. Tong, J. Mao, R. Bohn, J. Messina, L. Badger, and D. Leaf, "Nist cloud computing reference architecture," *NIST special publication*, vol. 500, no. 2011, pp. 1–28, 2011.

[11] S. Nist and T Grance, "The nist definition of cloud computing," *Commun. ACM*, vol. 53, no. 6, pp. 50–50, 2011.

[12] M. Hogan, F. Liu, A. Sokol, and J. Tong, "Nist cloud computing standards roadmap," *NIST Special Publication*, vol. 35, pp. 6–11, 2011.

[13] D. Talia, "Clouds for scalable big data analytics," *Computer*, vol. 46, no. 5, pp. 98–101, 2013.

[14] S. Subashini and V. Kavitha, "A survey on security issues in service delivery models of cloud computing," *Journal of Network and Computer Applications*, vol. 34, no. 1, pp. 1 –11, 2011, ISSN: 1084-8045. DOI: https://doi.org/10.1016/j.jnca.2010.07.006. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1084804510001281.

[15] X. Tan and B. Ai, "The issues of cloud computing security in high-speed railway," in *Proceedings of 2011 International Conference on Electronic Mechanical Engineering and Information Technology*, vol. 8, 2011, pp. 4358–4363.

[16] J. Feng, Y. Chen, W. Ku, and P. Liu, "Analysis of integrity vulnerabilities and a non-repudiation protocol for cloud data storage platforms," in *2010 39th International Conference on Parallel Processing Workshops*, 2010, pp. 251–258.

[17] "Cyberguarder: A virtualization security assurance architecture for green cloud computing," *Future Generation Computer Systems*, vol. 28, no. 2, pp. 379 –390, 2012, ISSN: 0167-739X.

[18] H. Tianfield, "Security issues in cloud computing," in *2012 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*, 2012, pp. 1082–1089.

[19] M. Zhou, R. Zhang, W. Xie, W. Qian, and A. Zhou, "Security and privacy in cloud computing: A survey," in *2010 Sixth International Conference on Semantics, Knowledge and Grids*, IEEE, 2010, pp. 105–112.

[20] "Integration of cloud computing and internet of things: A survey," *Future Generation Computer Systems*, vol. 56, pp. 684 –700, 2016.

[21] W. Shi, J. Cao, Q. Zhang, Y. Li, and L. Xu, "Edge computing: Vision and challenges," *IEEE internet of things journal*, vol. 3, no. 5, pp. 637–646, 2016.

[22] W. Shi, J. Cao, Q. Zhang, Y. Li, and L. Xu, "Edge computing: Vision and challenges," *IEEE Internet of Things Journal*, vol. 3, no. 5, pp. 637–646, 2016. DOI: 10.1109/JIOT.2016.2579198.

[23] M. De Donno, K. Tange, and N. Dragoni, "Foundations and evolution of modern computing paradigms: Cloud, iot, edge, and fog," *Ieee Access*, vol. 7, pp. 150 936–150 948, 2019.

[24] P. Garcia Lopez, A. Montresor, D. Epema, A. Datta, T. Higashino, A. Iamnitchi, M. Barcellos, P. Felber, and E. Riviere, *Edge-centric computing: Vision and challenges*, 2015.

[25] B. Varghese, N. Wang, S. Barbhuiya, P. Kilpatrick, and D. S. Nikolopoulos, "Challenges and opportunities in edge computing," in *2016 IEEE International Conference on Smart Cloud (SmartCloud)*, IEEE, 2016, pp. 20–26.

[26] M. Satyanarayanan, "Pervasive computing: Vision and challenges," *IEEE Personal communications*, vol. 8, no. 4, pp. 10–17, 2001.

[27] R. Balan, J. Flinn, M. Satyanarayanan, S. Sinnamohideen, and H.-I. Yang, "The case for cyber foraging," in *Proceedings of the 10th workshop on ACM SIGOPS European workshop*, 2002, pp. 87–92.

[28] C. Mouradian, D. Naboulsi, S. Yangui, R. H. Glitho, M. J. Morrow, and P. A. Polakos, "A comprehensive survey on fog computing: State-of-the-art and research challenges," *IEEE Communications Surveys & Tutorials*, vol. 20, no. 1, pp. 416–464, 2017.

[29] G. A. Lewis, S. Echeverría, S. Simanta, B. Bradshaw, and J. Root, "Cloudlet-based cyber-foraging for mobile systems in resource-constrained edge environments," in *Companion Proceedings of the 36th International Conference on Software Engineering*, 2014, pp. 412–415.

[30] M. T. Beck, M. Werner, S. Feld, and S Schimper, "Mobile edge computing: A taxonomy," in *Proc. of the Sixth International Conference on Advances in Future Internet*, Citeseer, 2014, pp. 48–55.

[31] G. I. Klas, "Fog computing and mobile edge cloud gain momentum open fog consortium, etsi mec and cloudlets," 2015.

[32] P. M. Pinto Silva, J. Rodrigues, J. Silva, R. Martins, L. Lopes, and F. Silva, "Using edge-clouds to reduce load on traditional wifi infrastructures and improve quality of experience," in *2017 IEEE 1st International Conference on Fog and Edge Computing (ICFEC)*, 2017, pp. 61–67.

[33] P. M. Pinto Silva, J. Rodrigues, J. Silva, R. Martins, L. Lopes, and F. Silva, "Using edge-clouds to reduce load on traditional wifi infrastructures and improve quality of experience," in *2017 IEEE 1st International Conference on Fog and Edge Computing (ICFEC)*, 2017, pp. 61–67.

[34] M. K. Yogi, K Chandrasekhar, and G. V. Kumar, "Mist computing: Principles, trends and future direction," *arXiv preprint arXiv:1709.06927*, 2017.

[35] M. M. Mosbah, H. Soliman, and M. A. El-Nasr, "Current services in cloud computing: A survey," *ArXiv*, vol. abs/1311.3319, 2013.

[36] F. Bonomi, R. Milito, J. Zhu, and S. Addepalli, "Fog computing and its role in the internet of things," in *Proceedings of the First Edition of the MCC Workshop on Mobile Cloud Computing*, ser. MCC '12, Helsinki, Finland: Association for Computing Machinery, 2012, 13–16, ISBN: 9781450315197. DOI: 10.1145/2342509.2342513. [Online]. Available: https://doi.org/10.1145/2342509.2342513.

[37] R. K. Naha, S. Garg, D. Georgakopoulos, P. P. Jayaraman, L. Gao, Y. Xiang, and R. Ranjan, "Fog computing: Survey of trends, architectures, requirements, and research directions," *IEEE access*, vol. 6, pp. 47 980–48 009, 2018.

[38] A. Yousefpour, G. Ishigaki, R. Gour, and J. P. Jue, "On reducing iot service delay via fog offloading," *IEEE Internet of Things Journal*, vol. 5, no. 2, pp. 998–1010, 2018. DOI: 10.1109/JIOT.2017.2788802.

[39] X. Sun and N. Ansari, "Edgeiot: Mobile edge computing for the internet of things," *IEEE Communications Magazine*, vol. 54, no. 12, pp. 22–29, 2016. DOI: 10.1109/MCOM.2016.1600492CM.

[40] O. C. A. W. Group *et al.*, "Openfog architecture overview," *White Paper OPFWP001*, vol. 216, p. 35, 2016.

[41] F. Bonomi, R. Milito, P. Natarajan, and J. Zhu, "Fog computing: A platform for internet of things and analytics," in *Big data and internet of things: A roadmap for smart environments*, Springer, 2014, pp. 169–186.

[42] P. Yang, N. Zhang, Y. Bi, L. Yu, and X. S. Shen, "Catalyzing cloud-fog interoperation in 5g wireless networks: An sdn approach," *IEEE Network*, vol. 31, no. 5, pp. 14–20, 2017.

[43] M. A. Habibi, M. Nasimi, B. Han, and H. D. Schotten, "A comprehensive survey of ran architectures toward 5g mobile communication system," *IEEE Access*, vol. 7, pp. 70 371–70 421, 2019. DOI: 10.1109/ACCESS.2019.2919657.

[44] M. Taneja and A. Davy, "Resource aware placement of data analytics platform in fog computing," *Procedia Computer Science*, vol. 97, pp. 153–156, 2016.

[45] M. Aazam and E.-N. Huh, "Fog computing micro datacenter based dynamic resource estimation and pricing model for iot," in *2015 IEEE 29th International Conference on Advanced Information Networking and Applications*, IEEE, 2015, pp. 687–694.

[46] N. K. Giang, M. Blackstock, R. Lea, and V. C. Leung, "Developing iot applications in the fog: A distributed dataflow approach," in *2015 5th International Conference on the Internet of Things (IOT)*, IEEE, 2015, pp. 155–162.

[47] H Tom and L. Gao, "Fog computing: Focusing on mobile users at the edge," *Networking and Internet Architecture*, 2016.

[48] M. Asif-Ur-Rahman, F. Afsana, M. Mahmud, M. S. Kaiser, M. R. Ahmed, O. Kaiwartya, and A. James-Taylor, "Toward a heterogeneous mist, fog, and cloud-based framework for the internet of healthcare things," *IEEE Internet of Things Journal*, vol. 6, no. 3, pp. 4049–4062, 2019. DOI: 10.1109/JIOT.2018.2876088.

[49] Y. Lin and H. Shen, "Cloudfog: Leveraging fog to extend cloud gaming for thin-client mmog with high quality of service," *IEEE Transactions on Parallel and Distributed Systems*, vol. 28, no. 2, pp. 431–445, 2017. DOI: 10.1109/TPDS.2016.2563428.

[50] S. Yi, Z. Hao, Q. Zhang, Q. Zhang, W. Shi, and Q. Li, "Lavea: Latency-aware video analytics on edge computing platform," in *2017 IEEE 37th International Conference on Distributed Computing Systems (ICDCS)*, 2017, pp. 2573–2574. DOI: 10.1109/ICDCS.2017.182.

[51] U. Drolia, K. Guo, J. Tan, R. Gandhi, and P. Narasimhan, "Cachier: Edge-caching for recognition applications," in *2017 IEEE 37th International Conference on Distributed Computing Systems (ICDCS)*, 2017, pp. 276–286. DOI: 10.1109/ICDCS.2017.94.

[52] R. Szeliski, *Computer vision: algorithms and applications*. Springer Science & Business Media, 2010.

[53] C. Huang, R. Lu, and K. R. Choo, "Vehicular fog computing: Architecture, use case, and security and forensic challenges," *IEEE Communications Magazine*, vol. 55, no. 11, pp. 105–111, 2017. DOI: 10.1109/MCOM.2017.1700322.

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

[55] S. Yi, Z. Qin, and Q. Li, "Security and privacy issues of fog computing: A survey," in *International conference on wireless algorithms, systems, and applications*, Springer, 2015, pp. 685–695.

[56] I. Stojmenovic and S. Wen, "The fog computing paradigm: Scenarios and security issues," in *2014 federated conference on computer science and information systems*, IEEE, 2014, pp. 1–8.

[57] H. Han, B. Sheng, C. C. Tan, Q. Li, and S. Lu, "A measurement based rogue ap detection scheme," in *IEEE INFOCOM 2009*, IEEE, 2009, pp. 1593–1601.

[58] D. Balfanz, D. K. Smetters, P. Stewart, and H. C. Wong, "Talking to strangers: Authentication in ad-hoc wireless networks.," in *NDSS*, Citeseer, 2002.

[59] F. Klaedtke, G. O. Karame, R. Bifulco, and H. Cui, "Access control for sdn controllers," in *Proceedings of the third workshop on Hot topics in software defined networking*, 2014, pp. 219–220.

[60] K.-K. Yap, Y. Yiakoumis, M. Kobayashi, S. Katti, G. Parulkar, and N. McKeown, "Separating authentication, access and accounting: A case study with openwifi," *Open Networking Foundation, Tech. Rep*, 2011.

[61] C. Wang, Q. Wang, K. Ren, and W. Lou, "Privacy-preserving public auditing for data storage security in cloud computing," in *2010 proceedings ieee infocom*, Ieee, 2010, pp. 1–9.

[62] N. Cao, S. Yu, Z. Yang, W. Lou, and Y. T. Hou, "Lt codes-based secure and reliable cloud storage service," in *2012 Proceedings IEEE INFOCOM*, IEEE, 2012, pp. 693–701.

[63] R. Gennaro, C. Gentry, and B. Parno, "Non-interactive verifiable computing: Outsourcing computation to untrusted workers," in *Annual Cryptology Conference*, Springer, 2010, pp. 465–482.

[64] B. Parno, J. Howell, C. Gentry, and M. Raykova, "Pinocchio: Nearly practical verifiable computation," in *2013 IEEE Symposium on Security and Privacy*, IEEE, 2013, pp. 238–252.

[65] D. X. Song, D. Wagner, and A. Perrig, "Practical techniques for searches on encrypted data," in *Proceeding 2000 IEEE Symposium on Security and Privacy. S&P 2000*, IEEE, 2000, pp. 44–55.

[66] R. Lu, X. Liang, X. Li, X. Lin, and X. Shen, "Eppa: An efficient and privacy-preserving aggregation scheme for secure smart grid communications," *IEEE Transactions on Parallel and Distributed Systems*, vol. 23, no. 9, pp. 1621–1631, 2012.

[67] W. Wei, F. Xu, and Q. Li, "Mobishare: Flexible privacy-preserving location sharing in mobile online social networks," in *2012 Proceedings IEEE INFOCOM*, IEEE, 2012, pp. 2616–2620.

[68] A. Alzahrani, N. Alalwan, and M. Sarrab, "Mobile cloud computing: Advantage, disadvantage and open challenge," in *Proceedings of the 7th Euro American Conference on Telematics and Information Systems*, New York, NY, USA: ACM, 2014.

[69] E. Damiani, D. C. di Vimercati, S. Paraboschi, P. Samarati, and F. Violante, "A reputation-based approach for choosing reliable resources in peer-to-peer networks," in *Proceedings of the 9th ACM conference on Computer and communications security*, 2002, pp. 207–216.

[70] A. Isidori, *Nonlinear Control Systems: An Introduction*, 2nd. Springer-Verlag: Berlin, 1989.

[71] A. J. Krener, "Approximat linearization by state feedback and coordinate change," *Systems Control Letters*, vol. 5, pp. 181–185, 1984.