Mathematical Statistician and Engineering Applications
ISSN: 2094-0343
2326-9865

Potential Challenges of Developing Large-Scale Computing Infrastructures over Distributed Clouds

Mirza Mohammed Akram Baig

Senior Member of Technical Staff, Illumio Inc

Opinions expressed in the paper are solely of the author(s) and do not represent the company's views.

Article Info

Page Number: 122 - 145

Publication Issue:

Vol 71 No. 4 (2022)

Abstract

Distributed information management systems facilitate numerous network services. However, the growing size of its infrastructures makes it difficult for administrators to manage large-scale dynamic distributed systems in an efficient manner. Developing large-scale computing infrastructures over distributed clouds presents various challenges. The key challenges addressed in this paper include scalability, load balancing, cloud interoperability, network latency, and fault tolerance. Scalability challenges mostly develop due to the increased monitoring of data of different sizes. Load balancing challenges develop because of the difficulties involved in harmonizing the workload in a distributed cloud. Concerning cloud interoperability, the sustenance of operations becomes more and more challenging as technology advances. Cloud interoperability challenges identified include portability and mobility, cloud-service integration, security, privacy, and trust, along with management, monitoring, and audit. Network latency challenges arise from delays that happen because of the time taken to process data on a server and return results back to the client on the network. The effectiveness of network latency measures depends on two factors – latency and traffic load, along with latency standards. Fault tolerance also counted among the potential challenges of developing large scale computing infrastructures over distributed computing. Some of the main challenges to consider include heterogeneity and the deficiency in standards, need for automation, downtime in the clouds, Recovery Point Objective (RPO) and Recovery Time Objective (RTO) considerations, as well as workloads in the cloud. Preparing for and minimizing the likelihood of these challenges can boost the development of large-scale computing infrastructures over distributed clouds.

Article History

Article Received*: 25 March 2022*

Revised*: 30 April 2022*

Accepted*: 15 June 2022*

***Publication**: 19 August 2022* **Keywords**: cloud computing, virtual machines, scalability, load balancing, cloud interoperability, network latency, fault tolerance.

1. Introduction

In today's society, large-scale computing infrastructures have become crucial in creating and implementing emergent distributed applications [1]. Distributed applications operate on various computers on a network simultaneously and can be stored on servers or within the cloud computing environment. Compared to traditional applications that operate on one system, distributed applications function on various systems at the same time to fulfil one task or job.

Distributed information management systems facilitate numerous network services, including monitoring and management, resource management, service placement, task scheduling, and distribution of content [1]. Some examples of large-scale computing infrastructures include PlanetLaband Enterprise Desktop Grids. These computing infrastructures facilitate the functioning of distributed applications, examples being peer-to-peer systems, content distribution networks, and distributed games [1].

Most of these computing infrastructures feature numerous personal workstations and dedicated servers dispersed globally. For instance, PlanetLab, a research network available worldwide that encouraged the establishment of novel network services had 1353 nodes located at 717 sites in 48 countries at its peak[2]. One more example is the Planet-Scale grid costing about 5 billion euros ($6.3 billion U.S.) with over 100,000 CPUs[3]. These CPUs consist mostly of PCs and workstations that can be accessed across universities and research labs spread across the U.S., Europe, Taiwan, Japan, and other locations [3]. With new developments in the computing world and the size of infrastructures growing, administrators find it very tough to administer large-scale dynamic distributed systems in an efficient manner.

From a broad perspective, large-scale computing infrastructures can be classified into grid and cloud computing [4], [5]. Grids consist of resource cluster pools in various geological locations controlled by enormous organizations in the government and education spheres. Cloud computing, on the other hand, expands grids into the business sector through the provision of resources based on a subscription.

Cloud computing presents resources considered abstract and its resource capacity is classified as elastic. Besides, cloud computing also features a programmable self-service interface that utilizes a pay-per-use pricing system[4].The art of providing large-scale computing resources through the cloud was initially limited to a few providers. However, the arrival and quick expansion of private and hybrid clouds has considerably strengthened providers and consumers of large-scale resources[4]. The increase in large-scale resource providers has also been supported by the convenience of cloud computing toolkits, including Eucalyptus[6], Nimbus[7], and OpenNebula[8].

Developing large-scale computing infrastructures and providing their resources in the cloud environment is a challenge[4], [9]. Conventional scheduling methods have been used for grid scheduling, but these are not suitable, neither are they effective for the cloud environment. Precisely, these scheduling methods are not feasible where hybrid clouds allow the use of resources from various providers[4]. Even though the cloud presents a picture of access to unlimited resources, this is an illusion since there are many other aspects to consider before utilizing the service such as cost, flexibility, and programmability[4], [9].

Developing large-scale computing infrastructures over distributed clouds presents various challenges. To begin with, the distributed factors contribute to the occurrence of various issues, including cloud interoperability, fault tolerance, and fault tolerance among many others [10]. Second, such an infrastructure features numerous nodes and this introduces new encounters affiliated with the scalability of cloud infrastructures and distributed applications [10].

2. Potential Challenges

2.1 Scalability

Scale is one of the key challenges that develops when managing millions of cores. The actions used to scale in distributed clouds may be classified into vertical and horizontal scaling[11]. Vertical scaling entails incorporating more horsepower to the equipment utilized by the systems. It is possible to add more horsepower by including more processors, bandwidth, and memory among other aspects. Vertical scaling is the means through which applications are utilized on large shared-memory servers[11]. Quite the reverse, horizontal scaling involves including more of related software or hardware resources. For

instance, in a normal service consisting of two layers, additional front-end nodes are incorporated or released whenever there is a rise in users or in the quantity of the workload. Horizontal scaling suits more applications that are installed on distributed servers[11].

In many ways, scalability challenges develop due to the increased monitoring of data sizes of different magnitudes [12], [13]. Besides, provisioning, deployment, and scheduling approaches must work across scale when developing large-scale computing infrastructures over distributed clouds. Common monitoring and management systems are consolidated and have no ability to level-up to millions of management objects within cloud systems [12], [13]. As such, there is need for more distributed approaches that have scalability properties when creating large scale computing infrastructures over distributed platforms. This would allow for the easy scaling up or down of the systems used for monitoring and management and help meet the necessary cloud requirements [12], [13].

Management of failure and optimization of performance also require application of scalable and intelligent methods[12], [13]. Such methods would help examine the huge amounts of monitoring data and incorporate smart optimizations to deal with challenges. Scalable approaches would guarantee dynamic management of resources in the presence of regular failure of components. Besides, the many components of the system work at various time scales, additionally heightening challenges with the management process[12], [13].

Continuous monitoring and analysis are a primary element for closed loop management within cloud data centers[12]. Prevailing approaches use centralized methods to collect data, aggregate and analyze it across datacenter subsystems and machines. These current solutions encounter various scale shortcomings within cloud data centers. Data centers that are large would generally result in enormous amount of monitoring data generated across various management domains. These monitoring and analysis solutions would face scalability challenges associated with high load on the central management servers (CMS), prolonged response time for analyzing key events, and decreased accuracy resulting from bigger sampling frequencies [12].

One more challenge with scalability of large-scale computing infrastructures in distributed clouds entails change in the use, load, as well as structure of data center machines [12]. It is necessary that the monitoring and analysis structure demonstrate minimal lag and inevitable latencies under changes happening due to an increase in load or a change in the size of the system or rather the structure of the data centers. Nonetheless, computing infrastructures should also be robust to failures associated with examining parts within a data center[12].

Attempts to deal with the scalability challenges in large scale computing infrastructures have previously been tested on OpenCirus, a scientific research cloud that features high rates of isolation[12], [14]. The tested infrastructure relies on the general principles of data local analysis, adaptable distributed architectures, and adaptation based on the control layer [12].As a cloud computing testbed, OpenCirrus features 14 sites that are scattered geographically, each consisting of a minimum of 1,000 cores along with supplementary memory and storage [12], [15]. Since every site is controlled separately, the overall testbed exists as a union of varied sites. In many research projects, OpenCirrus provides the benefits of operating experiments at scale, taking advantage of frequently shared stack, services, as well as best practices. OpenCirrus also helps create interactions across the layers. Applications integrating OpenCirrus tend to improve in performance and scalability making use of outcomes from the basic systems research. For instance, Jayasinghe and his colleagues achieved relatively higher throughput values after carrying out experiments on OpenCirrus[14]. More crucially, OpenCirrus showed much better scalability. The researchers did not notice any drop in performance when moving n-tier application from a traditional datacenter to an Infrastructure as a Service (IaaS) cloud [14].

In a different experiment, Jayasinghe and his colleagues examine IaaS clouds using multi-tier workloads [16]. Most importantly, the researchers use the RUBBoS benchmark application and evaluate its performance and scalability when presented in Amazon EC2, OpenCirrus, and Emulab[16]. The findings of the study showed that increasing the number of nodes provided better performance on Emulab and OpenCirrus. Besides, RUBBoS showed good scalability on Emulab and OpenCirrus, but showed poor scalability on EC2[16]. These findings imply that creating large-scale computing

infrastructuresin the cloud computing environment requires a lot of experimental analysis to be fully understood and recognized as a strong technological alternative.

Over the years, there have been scholarly attempts to address scalability challenges in the cloud computing environment. Most solutions targeting cloud data centers seem to address scalability issues by minimizing the number of VM resources taken into account, in most cases considering information related to the CPU or the memory [17], [18]. While useful, these methods are likely to encounter major drawbacks since restricting the monitoring to CPU or memory resources may be inefficient in supporting the consolidation strategies of virtual machines. With these challenges in mind, Canali and Lancellotti suggested in their study that it is possible to deal with scalability issues by taking advantage of the similarity in the behavior of virtual machines, especially in terms of resource use patterns [18].Based on the proposed method, the best approach to managing scalability challenges should entail clustering virtual machines operating similar customer applications and showing related behaviors in the use of resources[18].

While solutions to scalability challenges have been proposed [14], [16]–[18], Cáceres et al. suggest that the proposed solutions are not easy to implement [11]. These researchers indicate that the factors that could enhance or weaken scalability could be difficult to identify. At times, the actions implemented to enhance one of the scalability capabilities could eventually spoil the others. For instance, Cáceres et al. submit that the incorporation of compression algorithms to enhance Space scalability could affect load scalability [11].

In sum, scalability must be kept in mind from the very start when developing large-scale computing infrastructures over distributed clouds. This guarantees that a system can grow and increase in complexity based on the number of users from hundreds to thousands or even millions. By taking scalability into account, the chances of failure and reimplementation of the systems within distributed clouds would be minimized[11].

2.2 Load balancing

Load balancing presents another potential challenge in the cloud computing environment. It is an approach that entails disbursing the workload regularly to each node in the workspace to ensure that no

node in the system is either dazed or idle every time [19]. In an efficient load balancing set of rules, the system makes sure that every node in the system has an identical measure of work. Balancing the workload in a distributed cloud is one of the remarkable stresses since it is difficult to understand the number of requests distributed occasionally in the cloud system [19].

In essence, load balancing remains a significantissuein cloud computing and requires allocating the work load impartially on all the nodes to encourage better use of the present resources[19].
Load balancing can be achieved using a number of techniques in the cloud environment, including LBVS, honeybee foraging behavior, CLBVM, SBLB for internet distributed services, join-idle queue, decentralized content aware LB, index name server, stochastic hill climbing, HBB-LB, cloud server optimization, response time-based LB, ant colony optimization, PLBS, and A2LB[19]–[22].The first four techniques are further explained below.

A few studies[19]–[21] have suggested a load balancing method based on Virtual Storage (LBVS), which provides a lot of data storing abilities on cloud. Under LBVS, the system achieves load balancing using two modules that are a duplicate copy, thereby reducing the reaction time and enhancing the potential of rescuing the system from tragedy. The LBVS technique helps strengthen the use ratio of stock resource, and helps build the flexibility and strength of the large-scale computing infrastructure[19], [21].

A second load balancing technique for the cloud draws inspiration from the Honeybee[19], [22]. The technique is encouraged by the supposed behavior of a cluster of honeybees searching and collecting food. Forager bees find relevant sources of food and return to the hive to communicate the information to the hive using the “waggle dance,” [22]. Next, the honeybeesgo along with the forager to the found source of food and begin collecting it. In the same manner, the honeybee foraging behavior load balancing technique employs a group of servers that are organized into virtual servers. Each server works on a simulated service list of requests[22]. The technique performs load balancing with the help of close-by server activities. An increase in the range of the infrastructure improves the efficiency of the system. This load balancing technique is more suitable in cases where typical forms of service are needed [19].

A different case involves the use of a Central Load Balancing Strategy for Virtual Machines (CLBVM)[19], [23]. The CLBVM technique oversees the job of the cloud delicately. The policy considers various considerations when engaged in load balancing. First, it considers the network load as being constant and not changing regularly [23]. The technique also assumes that each virtual machine has various identifications. While the approach improves the whole system efficiency, the load balancing algorithm takes into account the frequent change in state, to the extent that unwarranted migrations are shunned [19], [23].

Load balancing can also be achieved using the Server-based load balancing (SBLB) for internet distributed services technique [19], [24]. This method encourages reducing the facility reaction times using a set of rules that limit the redistribution of demands to the close-by distant servers. The technique helps web servers to endure overloads. Experiments on SBLB for internet distributed services have revealed slow migration and response time [19], [24]. This shows that the time required to move the tasks from one node to the other is minimal and the moment between sending a request and the response time is also low [19]. The slow periods reported enhance the entire efficiency of the technique.

A description of the other techniques follows here in a nutshell. Join-idle queue is suitable for internet resources that can be extended autonomously and provides a lot of load balancing utilizing discrete dispatchers [19], [25]. The decentralized content aware LB strategy helps the scheduler to pinpoint the most suitable node for processing the demands[19], [26].The Index Name Server (INS) load balancing technique excludes the duplication and repetition of information in the cloud framework. The technique works based on a mix of repetition takes full advantage of the access point [19]. The Stochastic Hill Climbing method supports easy loop shifting and includes two parameters. The first parameter is a contender creator while the other is an approximation that positions every suitable result [19]. The Honey Bee Behavior inspired Load Balancing (HBB-LB) technique supports to gain efficient load balancing all around virtual machines and increase throughput [19], [27]. The cloud server optimization technique employs two standards to facilitate cloud improvement. The first standard advances the cloud framework at the host machine level, while the second standard improves the cloud framework using active limit values[19]. The response time based LB technique is a defensive load balancing technique

that focuses on the reaction time of each demand [19]. The ant colony optimization technique spots the overwhelmed and servers loaded below, thereby operating actions affiliated with load balancing among servers in a data center[19]. The PLBS is a centralized load balancing policy whose goal is to reduce the unevenness of the load on the nodes [19], while the Autonomous Agent Based Load Balancing (A2LB) technique presents positive load computation of a virtual machine within a data center[19], [28].

A review of scholarly work reveals that load balancing in the distributed cloud faces many potential challenges [29]. Some of the challenges mentioned include virtual machine migration, spatially distributed nodes in a cloud, single point of failure, algorithm complexity, emergence of small data centers, and energy management[29]. These challenges are further elaborated in the next passages.

Cloud computing has a service-on-demand nature that requires that resources be provided whenever there is a service request[29]. In some cases, resources have to be moved from a single physical server to another, at times on a far location. As such, the development of load-balancing algorithms in large-scale computing infrastructures must consider the time of migration and the security of the apparatus involved. These two factors affect the performance and influence the probability of attacks on distributed clouds[29].

The cloud computing nodes are also dispersed geologically. The challenge with such a distribution is that the set of rules of load balancing must be developed to take care of certain parameters, such as the bandwidth of the network, the spaces among the nodes, speeds of communication, and the space between the client and the available resources [29]. As such, the spatial distribution of nodes in distributed clouds presents a major challenge in the development of large-scale computing infrastructures.

As noted earlier in this paper, some of the load-balancing procedures take on a centralized approach[29]. With such a design, the whole system stands a chance of crashing of the node executing the algorithm or the controller succumbs to failure. Such a possibility presents a challenge when

developing large scale computing infrastructures since one must develop distributed or rather devolved algorithms[29].

The algorithms used in load balancing should also be simple. Specifically, the development must put in mind the implementation and operation aspects. The use of complex algorithms can potentially cause adverse effects on the full performance of a large-scale computing infrastructure running on a distributed cloud[29].

The emergence of miniature data centers in cloud computing can also be a challenge in load balancing[29]. In essence, small data centers are much inexpensive and use less energy compared to large data centers. With small data centers, it is possible to distribute computing resources across the world. However, the benefits of small data centers can also present a major challenge in load balancing. The key issue here is to develop load-balancing algorithms that can facilitate a decent response time[29].

The other challenge to consider when designing load balancing algorithms in distributed clouds relates to energy management[29]. The design of the algorithms should allow them to reduce the amount of energy consumed. Load balancing should be designed in a manner that takes advantage of the physical servers. Efficiently working algorithms would easily monitor the workload of servers and migrate the virtual machines from under-loaded servers to those that have minimal load. In essence, efficient load balancing mechanisms will aid power management aspects as well[29].

2.3Cloud interoperability

Cloud interoperability has been defined in previous scholarly work as the effortlessness of relocation and incorporation of applications and data between various cloud providers[30].With advancements in cloud computing, the capacity to sustain interoperability turns out to be more and more significant. System-of-systems engineering is essential in mapping out interoperability services in the distributed cloud setup. One of the most popular models for system-of-systems interoperability includes the Levels of Information System Interoperability (LISI) Maturity Model issued by the Department of Defense (DoD) C4ISR Architecture Working Group [30], [31]. The LISI model groups the level of

complexityby considering the exchange and sharing of information and services among systems. These processes are fulfilled based on PAID, which represents the intently connectedelements – Procedures, Applications, Infrastructure, and Data[30]. These attributes are discussed extensively in the next passages.

The procedures attribute stands for the level of interoperability emanating from functioningrules and processes, effective program development guidance, together with conformity of technical and system design specifications. In many cases, the specifications may cover hardware, communications, system software, application standards, and data [30].

The second attribute, application, echoes the capacity of the software applications to operate on distinct systems and platforms as they develop through the reliable stages of interoperability. Applications can often range from the unconnected positioned at the low end to those that are meant for cross-discipline or rather cross-organizational borders located towards the high end [30].

The infrastructure feature indicates the level and structure of connectivity existent between the systems and applications[30]. For instance, this may involve a comparison of a point-to-point phone connection against a wide-area network existing across various systems and communication procedures. Infrastructure also denotes the interaction of systems with each other. In this regard, an analysis of infrastructure may involve comparing the application specific interface against web services that are independent of the platform[30].

The last attribute, data, represents the suppleness of the format of data and the vibrancy of the information conveyed across systems and domains[30]. In many cases, information may range from files featuring one type of data to those with combined information. Mostly, the availability of data encourages all sorts of representation, presentation, along with exploitation[30].

The LISI model referred to earlier in this section consists of five levels of maturity: Level 0, Level 1, Level 2, Level 3, and Level 4[30], [32], [33]. Level 0 entails remote interoperability within a physical environment. This level features manual extraction and incorporation of data from various separate

systems[30], [32], [33]. Level 2 features linked interoperability within an environment with peers. The level features electronic connections with discrete data, individual applications, standardized product exchange, and simple collaboration. Level 3 features domain-based interoperability within a cohesive environment[30], [32], [33]. The level is typified by wide-area networks, distinct applications, collective data, cultured collaboration, and common databases. The final level facilitates enterprise-based interoperability within a collective environment. Level 4 features wide-area networks, shared data, common applications, cutting-edge collaborations, and cross-domain sharing of information. While the LISI model concentrates on the exchange of information between systems, it falls short in providing a foundation for examining the reliability of interoperability between clouds, otherwise referred to as cloud-to-cloud interoperability (C2CI)[30], [32], [33].

Cloud interoperability challenges likely to occur when developing large-scale computing infrastructures include portability and mobility, cloud-service integration, security, privacy, and trust, together with management, monitoring, and audit[30], [34]. The next few passages discuss these interoperability challenges in much detail.

When discussing portability and mobility challenges, most cloud computing adopters question whether they can deploy current cloud artifacts on the services of another service provider without adjusting these artefacts[30], [34].Portability refers to the capacity to shiftan image in a “down” state from one host to the other and load it at its last stop. Quite the reverse, mobility refers to the movement of anon-screen computer workload from a single host to the other without dropping the connections between clients[34]. Both portability and mobility are a prime indicator of the level of interoperability within distributed clouds. Precisely, mobility across the boundaries of cloud providers is one of the targets of a strong and interoperable cloud. An interoperable cloud also requires advancement in aspects, such as open standards for virtual machine (VM) images, cloud-to-cloud application interfaces (APIs), and development in virtualization technologies[30], [32], [34]. These aspects are meant to encourage the migration of live virtual machine sessions, worldwide IP addresses, as well as data services to fixed resources across cloud boundaries[30], [34].

The second challenge to interoperability when developing large scale computing infrastructures over distributed clouds relates to cloud-service integration. In most cases, enterprises must incorporate both on-premise together with software-as-a-service (SaaS) applications to manage the needs of a business and maintain control over the operations and data critical to the mission[30], [32], [34]. The guidelines used to incorporate software applications through an API require a lot of coding and ongoing maintenance because of the regular modifications and updates that come up. Allowing the interaction of both SaaS and on-premise applications through Web services and integrating service-oriented architecture (SOA) principles has been suggested as a suitable means of solving the cloud-service integration challenge [30], [32], [34].

The third interoperability challenge affecting distributed clouds is a combination of security, privacy and trust issues. Enterprises using large-scale computing infrastructures over distributed clouds need the assurance from the service providers that the provided services can be depended to deliver certain levels of security and privacy[30], [32]–[35]. For instance, enterprises need the assurance that the provided services can control access to personally identifiable information (PII) through cloud services. Dealing with security, privacy, and trust issues in a distributed cloud would therefore require solutions to the classical challenges that develop in multi-level security (MLS) and cross -domain systems. Some of these problems include federated identity management, adequate monitoring, logging and auditing, and active role-based access control (RBAC)[30]. Mature distributed cloud environments will also have to provide their customers with a suitable degree of security precision to assuage reservations concerning the security and privacy provided by the cloud[30], [36].

Interoperability issues tied to security are linked with the administration of the cloud. These issues may entail handling the users, resources and data through security policies. The developed policies seek to facilitate authentication, management of sessions, access control, and communications through the network [30]. Moving from a legacy client-server model to one based on the cloud lessens some security issues but also introduces new ones. Failure to comprehend the novel safety issues or instinctively trying to implement legacy security rules and guidelines during cloud migration can contribute to certain challenges. According to the Distributed Management Task Force (DMTF) Open Cloud Standards Incubator Process and Deliverables model, management and control of cloud security

is one of the components that is still work-in-progress[30], [35]. As the industry develops a variety of cloud solutions to facilitate large-scale computing infrastructures, the gap between management, control, and the security of the cloud cannot be overlooked. Distributed clouds with a distinct cloud security policy is a crucial indicator of the level of interoperability between clouds. Therefore, the development of large-scale computing infrastructures over distributed clouds must incorporate some acceptable means for accurately associating the quality of the safety guaranteed by the cloud service. Nelson et al. [37] add that it is crucial to develop formal trust relationships across cloud boundaries. Studies [30], [37] also indicate that solid methods for authenticating and authorizing users must be developed for clouds to be interoperable. Effective solutions to security, privacy, and trust are a viable indicator of efficient cloud interoperability.

Apart from security, privacy and trust issues in distributed clouds, enterprises must also consider management, monitoring, and audit aspects[30], [38]. Cloud users need to regularly be assured that the security and privacy policies developed to facilitate interoperability are regularly applied and that the service level agreements (SLA) are met as the cloud services transfer across various boundaries. For this to be achieved, even procedures and tool sets must be used to observe and account for the degree of services and conformity or infringement of safety and privacy procedures in distributed clouds[30], [38].

In sum, the development of large-scale computing infrastructures over distributed clouds must consider interoperability challenges[30]. Attaining semantic and syntactical interoperability remains a major challenge in cloud computing. Integral to achieving interoperability is the ability to utilize the most suited cloud service and data to resolve a certain business need at a time. Interoperability must also consider factors, such as cost, quality, and the level of security needed. There is a lot to learn from the open-source software community which is facilitating the development of cloud interoperability through the uptake of the OpenStack and Cloud Foundry projects[30].

2.4 Network Latency

Network latency refers to any form of delay that occurs due to time taken to process data on a server and return results back to the client on the network[39]–[41]. A low latency means that there are small-

scale delays in obtaining the response from the server, while high network latency means that the delay is much longer. Even though the maximum possible bandwidth of a network connection is usually fixed from a theoretical perspective, the actual bandwidth changes from time to time and is influenced by high latencies [40]. Disproportionate network latency can create bottlenecks and avert data from filling the network pipe. These blocks eventually decrease the effectiveness of bandwidth. The influence of latency on the network bandwidth can either be provisional and lasting for a few seconds or can be unrelenting resting on the source of the delays.

In essence, factors that influence network latency include thin client hardware, the internet service provider, and the maximum transmission unit. A thin client refers to any diskless workstation featuring minimum hardware resources. Such a client has less system memory or RAM compared to the typical computer. The thin client solely depends on the server for all the processing power, memory, storage of data, and many other application software. The response time of the thin client once a request has been made on the server depends on the hardware requirements, specifically the speed and the memory of the processor. Inefficient hardware requirements can, therefore, affect the functioning of the thin client and eventually influence network latency within the cloud[40].

The second factor that can affect network latency, and therefore affect services within a distributed cloud, is the internet service provider. Even though the bandwidth does not have a direct impact on the latency, the number of routers positioned between the client and the server can have an impact on the ping times. It is also possible for each router to create a routing delay, which can contribute to network latency and influence the development of large-scale computing infrastructures[40].

The maximum transmission unit (MTU) can also contribute to network latency. The MTU refers to the principal single packet that can move from a network, through the internet service provider, to the destination. The MTU is usually set at 1500, but may need to be lowered where a digital subscriber line (DSL) or where some other technology with a packet overhead is being used. Without lowering the value, a network can easily experience packet fragmentation, which can have a serious impact on the speed of the network and contribute to latency [40].

Various techniques can be used to compute network latency, including finding the Round Trip Time between two systems. The Round Trip Time refers to the quantity of time needed to send a communication from a single point to another and obtaining acknowledgement back to the sender. While the RTT is not the only means to calculate network latency, it is the most common [39], [40].A few tools that could be useful in measuring RTT include ping, open source utility SmokePing, and traceroute[39]. On a digital subscriber line or cable Internet connection, the network latencies range below 100 milliseconds (ms); the best latencies should be below 25 ms on these platforms. Quite the reverse, satellite internet connections report average latencies of 500 ms or higher[40]. Network latencies that are in excess can contribute to a decrease in the bandwidth effectiveness, especially when building large scale computing infrastructures over distributed clouds. To this end, it is always important to run a network latency test during the development process.

The effectiveness of network latency measures depends on two factors – latency and traffic load, along with latency standards. A change in the traffic load affects network latency. In most occasions, an increase in the load occasions an increase in network latency. At the same time, measuring latency while considering network load can be complicated. Measurements should be taken at different network loads to fully depict network latency against the traffic load. Nonetheless, knowing the bandwidth usage patterns makes it possible to test network latency with clear levels of network utilization. The second factor, latency standards, involves defining a level of latency that is considered acceptable. To minimize network latency issues in a distributed cloud, it is important to define normal end-to-end latency values to aid the setting of a reasonable latency goal. Standards are central to the monitoring process and help to identify typical latencies between every remote and the servers. A rise in the end-to-end latency could easily signify a network problem.

Some “rules of thumb” that could govern end-to-end network latency when developing large scale computing infrastructures fall under three categories. First, an RTT end-to-end latency of 30ms or less is healthy[40]. Such a measurement only needs to be monitored to track potential changes. Second, round trip latencies measuring between 30ms and 50ms must be monitored[40]. All potential approaches that can help lower the latencies should be considered in such a circumstance.The last category involves RTT latencies over 50ms, which often need immediate attention to ascertain the

cause of the latency and suggest possible solutions to minimize the end-to-end latency. Such latencies also need to be monitored to track any improvements[40].

2.5 Fault Tolerance

Fault tolerance also counts among the potential challenges of developing large scale computing infrastructures over distributed computing. The fault tolerance ability of a system allows it to offer the required services even with component failures, or in the event of one or more faults [42], [43].Large scale computing infrastructures are bound to experience failures because of errors and faults. An abnormal state or the presence of a bug may make the computing infrastructure incapable of performing its required tasks. Different types of faults that a computing infrastructure must develop tolerance towards include omission faults, aging related, response faults, software faults, timing faults, and miscellaneous faults as demonstrated in figure 1.

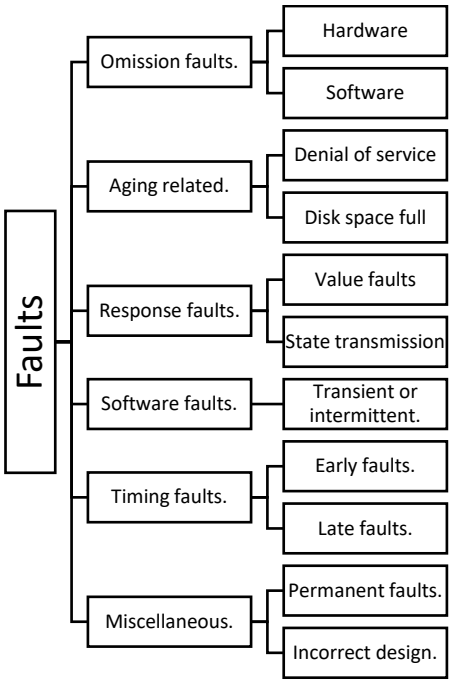

Figure 1 – Classification of faults [42].

The methods used to create fault tolerance capability in distributed computing can be grouped into three general kinds – redundancy techniques, tolerance policies, and load balancing fault tolerance. Examples of redundancy techniques include hardware, software, and time redundancy. The second

method involves the application of a series of policies that are either reactive or proactive. Methods for creating and increasing the fault tolerance ability of computing infrastructure based on load balancing can also rely on the hardware, software, and the network. [44]–[46] provide a detailed description of these methods.

Potential fault tolerance challenges in distributed clouds need to be considered when developing large scale computing infrastructures. Some of the main challenges to consider include heterogeneity and the deficiency in standards, need for automation, downtime in the clouds, Recovery Point Objective (RPO) and Recovery Time Objective (RTO) considerations, as well as workloads in the cloud. These challenges are described further in the next passages.

Within distributed computing, it is possible to commoditize computing resources. This capability allows various hardware and OS vendors to use varying architectures. It is also possible to use one large cloud system with components supported on heterogenous platforms. Such heterogeneity puts a lot of pressure on the development of fault tolerance solutions since they must consider factors affiliated with every OS vendor, creating potential challenges in the development of large scale computing infrastructures [47].

Distributed clouds have a smart future that needs extensive automation. With the number of virtual machines hosting cloud systems increasing regularly, it would be difficult for humans to manage such systems. Such a possibility means that the development of large scale computing infrastructures must consider automation procedures to manage the available fault tolerance solutions. However, automation still faces various challenges including the lack of generic frameworks (APIs) that can be applied to distributed clouds[47].

The other fault tolerance challenge is downtime in the clouds. The architecture of distributed clouds consists of various data centers that are organized geographically and controlled by different vendors [47]. Downtime on one data center can interfere with operations in many organizations. Since organizations have varying Service Level Agreements (SLAs) for clouds, enforcement of fault tolerance must take into account the SLAs for all the organizations within the same architecture [47].

Incorporation of fault tolerance in distributed clouds must also consider RPO and RTO. RPO refers to the amount of data that could be lost if a server experiences a failure, while RTO is the quantity of time needed for the system to set up and run after succumbing to a failure [47]. When developing large scale computing infrastructures, the goal is to minimize the RPO and RTO to the minimum. However, resilient methods must be modelled to facilitate continuous minimization of these factors.

The other consideration would be the workload in the cloud, which could be either cloud-enabled or cloud-native workloads[47]. It is possible to map all these workloads on a single fault tolerance framework that controls all the components as one unit. That means that the fault tolerance mechanism used within distributed clouds must include capabilities to monitor all the components, whether based locally or in the cloud in a single view[47]. This can prove challenging to achieve since both proactive and resilient methods must be incorporated.

3. Conclusion

Several challenges arise when building large-scale computing infrastructures over distributed channels. The distributed elements contribute to challenges such as cloud interoperability, network latency, along with fault tolerance. In the same manner, the process of developing the computing infrastructures brings to light other challenges with scalability and load balancing. These challenges could serve as building blocks that allow users to develop a myriad of computing infrastructures over distributed clouds.

References

[1] K. Huang and D. Zhang, “DHT-based lightweight broadcast algorithms in large-scale computing infrastructures,” *Future Gener. Comput. Syst.*, vol. 26, no. 3, pp. 291–303, Mar. 2010, doi: 10.1016/j.future.2009.08.013.

[2] “PlanetLab | An open platform for developing, deploying, and accessing planetary-scale services.” https://planetlab.cs.princeton.edu/ (accessed Mar. 11, 2022).

[3] P. Thibodeau, "Planet-Scale grid," *Computerworld*, Oct. 10, 2005. https://www.computerworld.com/article/2558519/planet-scale-grid.html (accessed Mar. 14, 2022).

[4] K. Lee, G. Buss, and D. Veit, "A heuristic approach for the allocation of resources in large-scale computing infrastructures," *Concurr. Comput. Pract. Exp.*, vol. 28, no. 5, pp. 1527–1547, Apr. 2016, doi: 10.1002/cpe.3709.

[5] I. Foster, C. Kesselman, and S. Tuecke, "The Anatomy of the Grid: Enabling Scalable Virtual Organizations," *Int. J. High Perform. Comput. Appl.*, vol. 15, no. 3, pp. 200–222, Aug. 2001, doi: 10.1177/109434200101500302.

[6] D. Nurmi *et al.*, "The Eucalyptus Open-Source Cloud-Computing System," in *2009 9th IEEE/ACM International Symposium on Cluster Computing and the Grid*, May 2009, pp. 124–131. doi: 10.1109/CCGRID.2009.93.

[7] P. Marshall, K. Keahey, and T. Freeman, "Elastic Site: Using Clouds to Elastically Extend Site Resources," in *2010 10th IEEE/ACM International Conference on Cluster, Cloud and Grid Computing*, May 2010, pp. 43–52. doi: 10.1109/CCGRID.2010.80.

[8] D. Milojičić, I. M. Llorente, and R. S. Montero, "OpenNebula: A Cloud Management Tool," *IEEE Internet Comput.*, vol. 15, no. 2, pp. 11–14, Mar. 2011, doi: 10.1109/MIC.2011.44.

[9] C. Reiss, A. Tumanov, G. R. Ganger, R. H. Katz, and M. A. Kozuch, "Heterogeneity and dynamicity of clouds at scale: Google trace analysis," in *Proceedings of the Third ACM Symposium on Cloud Computing*, New York, NY, USA, Oct. 2012, pp. 1–13. doi: 10.1145/2391229.2391236.

[10] P. Riteau, "Building Dynamic Computing Infrastructures over Distributed Clouds," in *2011 First International Symposium on Network Cloud Computing and Applications*, Nov. 2011, pp. 127–130. doi: 10.1109/NCCA.2011.27.

[11] J. Cáceres, L. M. Vaquero, L. Rodero-Merino, Á. Polo, and J. J. Hierro, "Service Scalability Over the Cloud," in *Handbook of Cloud Computing*, B. Furht and A. Escalante, Eds. Boston, MA: Springer US, 2010, pp. 357–377. doi: 10.1007/978-1-4419-6524-0_15.

[12] T. Forell, D. Milojicic, and V. Talwar, "Cloud Management: Challenges and Opportunities," in *2011 IEEE International Symposium on Parallel and Distributed Processing Workshops and Phd Forum*, May 2011, pp. 881–889. doi: 10.1109/IPDPS.2011.233.

[13] J. Gao, P. Pattabhiraman, X. Bai, and W. T. Tsai, "SaaS performance and scalability evaluation in clouds," Irvine, CA, USA, Dec. 2011. doi: 10.1109/SOSE.2011.6139093.

[14] D. Jayasinghe, S. Malkowski, Q. Wang, J. Li, P. Xiong, and C. Pu, "Variations in Performance and Scalability When Migrating n-Tier Applications to Different Clouds," in *2011 IEEE 4th International Conference on Cloud Computing*, Jul. 2011, pp. 73–80. doi: 10.1109/CLOUD.2011.43.

[15] A. I. Avetisyan *et al.*, "Open Cirrus: A Global Cloud Computing Testbed," *Computer*, vol. 43, no. 4, pp. 35–43, Apr. 2010, doi: 10.1109/MC.2010.111.

[16] D. Jayasinghe, S. Malkowski, J. Li, Q. Wang, Z. Wang, and C. Pu, "Variations in Performance and Scalability: An Experimental Study in IaaS Clouds Using Multi-Tier Workloads," *IEEE Trans. Serv. Comput.*, vol. 7, no. 2, pp. 293–306, Apr. 2014, doi: 10.1109/TSC.2013.46.

[17] C. Tang, M. Steinder, M. Spreitzer, and G. Pacifici, "A scalable application placement controller for enterprise data centers," in *Proceedings of the 16th international conference on World Wide Web*, New York, NY, USA, May 2007, pp. 331–340. doi: 10.1145/1242572.1242618.

[18] C. Canali and R. Lancellotti, "Improving Scalability of Cloud Monitoring Through PCA-Based Clustering of Virtual Machines," *J. Comput. Sci. Technol.*, vol. 29, no. 1, pp. 38–52, Jan. 2014, doi: 10.1007/s11390-013-1410-9.

[19] M. O. Ahmad and R. Z. Khan, "Load Balancing Tools and Techniques in Cloud Computing: A Systematic Review," in *Advances in Computer and Computational Sciences*, Singapore, 2018, pp. 181–195. doi: 10.1007/978-981-10-3773-3_18.

[20] N. J. Kansal and A. I. Chana, "Existing load balancing techniques in cloud computing: A systematic review," *J. Inf. Syst. Commun.*, vol. 3, no. 1, pp. 87–91, 2012.

[21] H. Liu, S. Liu, X. Meng, C. Yang, and Y. Zhang, "LBVS: A Load Balancing Strategy for Virtual Storage," in *2010 International Conference on Service Sciences*, May 2010, pp. 257–262. doi: 10.1109/ICSS.2010.27.

[22] M. Randles, D. Lamb, and A. Taleb-Bendiab, "A Comparative Study into Distributed Load Balancing Algorithms for Cloud Computing," in *2010 IEEE 24th International Conference on Advanced Information Networking and Applications Workshops*, Apr. 2010, pp. 551–556. doi: 10.1109/WAINA.2010.85.

[23] A. Bhadani and S. Chaudhary, "Performance evaluation of web servers using central load balancing policy over virtual machines on cloud," in *Proceedings of the Third Annual ACM Bangalore Conference*, New York, NY, USA, Jan. 2010, pp. 1–4. doi: 10.1145/1754288.1754304.

[24] S. Begum and C. S. R. Prashanth, "Review of Load Balancing in Cloud Computing," *Int. J. Comput. Sci. Issues IJCSI*, vol. 10, no. 1, pp. 343–352, Jan. 2013.

[25] Y. Lu, Q. Xie, G. Kliot, A. Geller, J. R. Larus, and A. Greenberg, "Join-Idle-Queue: A novel load balancing algorithm for dynamically scalable web services," *Perform. Eval.*, vol. 68, no. 11, pp. 1056–1071, Nov. 2011, doi: 10.1016/j.peva.2011.07.015.

[26] G. Soni and M. Kalra, "A novel approach for load balancing in cloud data center," in *2014 IEEE International Advance Computing Conference (IACC)*, Feb. 2014, pp. 807–812. doi: 10.1109/IAdCC.2014.6779427.

[27] D. B. L.D. and P. Venkata Krishna, "Honey bee behavior inspired load balancing of tasks in cloud computing environments," *Appl. Soft Comput.*, vol. 13, no. 5, pp. 2292–2303, May 2013, doi: 10.1016/j.asoc.2013.01.025.

[28] A. Singh, D. Juneja, and M. Malhotra, "Autonomous Agent Based Load Balancing Algorithm in Cloud Computing," *Procedia Comput. Sci.*, vol. 45, pp. 832–841, Jan. 2015, doi: 10.1016/j.procs.2015.03.168.

[29] E. J. Ghomi, A. M. Rahmani, and N. N. Qader, "Load-balancing algorithms in cloud computing: A survey," *J. Netw. Comput. Appl.*, vol. 88, pp. 50–71, Jun. 2017, doi: 10.1016/j.jnca.2017.04.007.

[30] S. Dowell, A. Barreto, J. B. Michael, and M.-T. Shing, "Cloud to cloud interoperability," in *2011 6th International Conference on System of Systems Engineering*, Jun. 2011, pp. 258–263. doi: 10.1109/SYSOSE.2011.5966607.

[31] Department of Defense, "C4ISR Architecture Framework Version 2.0." C4ISR Architecture Working Group (AWG), Dec. 18, 1997. Accessed: Mar. 16, 2022. [Online]. Available: https://www.afcea.org/education/courses/archfwk2.pdf

[32] R. Rezaei, T. K. Chiew, S. P. Lee, and Z. Shams Aliee, "Interoperability evaluation models: A systematic review," *Comput. Ind.*, vol. 65, no. 1, pp. 1–23, Jan. 2014, doi: 10.1016/j.compind.2013.09.001.

[33] R. Rezaei, T. K. Chiew, and S. P. Lee, "An interoperability model for ultra large scale systems," *Adv. Eng. Softw.*, vol. 67, pp. 22–46, Jan. 2014, doi: 10.1016/j.advengsoft.2013.07.003.

[34] J. Urquhart, "Exploring cloud interoperability, part 2," *CNET*, May 07, 2009. https://www.cnet.com/tech/tech-industry/exploring-cloud-interoperability-part-2/ (accessed Mar. 16, 2022).

[35] DMTF, "Interoperable clouds - A white paper from the Open Cloud Standards Incubator." Nov. 11, 2009. Accessed: Mar. 16, 2022. [Online]. Available: https://www.dmtf.org/sites/default/files/standards/documents/DSP-IS0101_1.0.0.pdf

[36] K. M. Khan and Q. Malluhi, "Establishing Trust in Cloud Computing," *IT Prof.*, vol. 12, no. 5, pp. 20–27, Sep. 2010, doi: 10.1109/MITP.2010.128.

[37] A. J. Nelson, G. W. Dinolt, J. B. Michael, and M.-T. Shing, "A security and usability perspective of cloud file systems," in *2011 6th International Conference on System of Systems Engineering*, Jun. 2011, pp. 161–166. doi: 10.1109/SYSOSE.2011.5966591.

[38] R. B. Uriarte, F. Tiezzi, and R. D. Nicola, "SLAC: A Formal Service-Level-Agreement Language for Cloud Computing," in *Proceedings of the 2014 IEEE/ACM 7th International Conference on Utility and Cloud Computing*, USA, Dec. 2014, pp. 419–426. doi: 10.1109/UCC.2014.53.

[39] S. U. Sharma and Y. B. Gandole, "Virtualization approach to reduce network latency for thin client performance optimization in cloud computing environment," Coimbatore, India., Jan. 2014. doi: 10.1109/ICCCI.2014.6921753.

[40] S. U. Sharma and Y. B. Gandole, "Understanding network latency in thin client environment," *Int. J. Eng. Sci. Innov. Technol. IJESIT*, vol. 2, no. 1, Jan. 2013, [Online]. Available: http://www.ijesit.com/Volume%202/Issue%201/IJESIT201301_04.pdf

[41] K. Obraczka and F. Silva, "Network latency metrics for server proximity," in *Globecom '00 - IEEE. Global Telecommunications Conference. Conference Record (Cat. No.00CH37137)*, Nov. 2000, vol. 1, pp. 421–427 vol.1. doi: 10.1109/GLOCOM.2000.892040.

[42] P. Kumari and P. Kaur, "A survey of fault tolerance in cloud computing," *J. King Saud Univ. - Comput. Inf. Sci.*, vol. 33, no. 10, pp. 1159–1176, Dec. 2021, doi: 10.1016/j.jksuci.2018.09.021.

[43] T. J. Charity and G. C. Hua, "Resource reliability using fault tolerance in cloud computing," in *2016 2nd International Conference on Next Generation Computing Technologies (NGCT)*, Oct. 2016, pp. 65–71. doi: 10.1109/NGCT.2016.7877391.

[44] M. Nazari Cheraghlou, A. Khadem-Zadeh, and M. Haghparast, "A survey of fault tolerance architecture in cloud computing," *J. Netw. Comput. Appl.*, vol. 61, pp. 81–92, Feb. 2016, doi: 10.1016/j.jnca.2015.10.004.

[45] A. Ganesh, M. Sandhya, and S. Shankar, "A study on fault tolerance methods in Cloud Computing," in *2014 IEEE International Advance Computing Conference (IACC)*, Feb. 2014, pp. 844–849. doi: 10.1109/IAdCC.2014.6779432.

[46] A. Fedoruk and R. Deters, "Improving fault-tolerance by replicating agents," in *Proceedings of the first international joint conference on Autonomous agents and multiagent systems: part 2*, New York, NY, USA, Jul. 2002, pp. 737–744. doi: 10.1145/544862.544917.

[47] M. A. Mukwevho and T. Celik, "Toward a Smart Cloud: A Review of Fault-Tolerance Methods in Cloud Systems," *IEEE Trans. Serv. Comput.*, vol. 14, no. 2, pp. 589–605, Mar. 2021, doi: 10.1109/TSC.2018.2816644.