ISSN 2278-3091
Volume 8, No.3, May - June 2019
International Journal of Advanced Trends in Computer Science and Engineering
Available Online at http://www.warse.org/IJATCSE/static/pdf/file/ijatcse17832019.pdf
https://doi.org/10.30534/ijatcse/2019/17832019

Enhancing Performance of IoT Networks through High Performance Computing

Dr. J, Sasi Bhanu[1], Dr. JKR Sastry[2], P. Venkata Sunil Kumar[3], B. Venkata Sai[4], K.V. Sowmya[5]
[1]CMR Institute of Technology, Kandlakoya Village, Medchal District, Hyderabad, India, bhanukamesh1@gmail.com
[2,] Koneru Lakshmaiah Education Foundation, Vaddeswaram, Andhra Pradesh, India, drsastry@kluniversity.in
[3,4]Koneru Lakshmaiah Education Foundation, Vaddeswaram, Andhra Pradesh, India, drsastry@kluniversity.in
[5]Koneru Lakshmaiah Education Foundation, Vaddeswaram, India, sowmyakambhampati@kluniversity.in

ABSTRACT

Performance of IOT based networks generally degrades as the system starts functions due to various reasons especially due to presence of heterogeneity; need to move the data from layer to layer in either of the directions to move the data from sensors to the storage of the same on remote server and to answer queries regarding the functioning of sensing and actuating mechanisms; the queries initiated from a remote location by the users through a Cloud computing interface. The performance degradation is generally due to the presence of heterogeneity and latency at each of the layers contained within an IOT network for forwarding the data packets either way. Maximum performance degradation is expected at gateway or restful services layer. The performance of an IOT network largely dependent of RESTful services layer especially the number of services to be supported are many in number.

This paper presents a method of optimizing the performance of the IOT networks through implementation of High performance computing at RESTful services layer.

Key words: IOT networks, Performance optimization, Gateway, latency

1. INTRODUCTION

Internet of Things (IoT) is commonly viewed as a network of items embedded with sensors that are connected to the Internet. The items may have embedded intelligence; the intelligence can also be distributed or hosted like in a cloud. A broader view of IoT is a networked connection of everyday objects including computers, sensors, humans, etc. loss of communication due to existence of interference, jitter, fading, noise, power dissipation, existence of heterogeneity, need for downsizing the speeds to match the speeds of smaller devices, frequent break down of the devices requiring re-configuration and many such factors.

Some of the IOT based system must function in real time failing which; the meeting of the deadlines may lead to disasters. The applications that are built through use of IOT based technologies are very time sensitive and the performance of such networks as designed is one the most important bottlenecks.

The number of devices connected via the Internet rapidly grows. One commonly estimates that around six billions humans and almost the same number of devices will be connected to the Internet by 2015. Nevertheless, IoT and the associated concept of smart world give rise to many complex optimization problems. The topic related to the efficient use of embedded, distributed or hosted intelligence in IoT is fundamental in order to address the smart world challenges.

Internet of things and related technologies are being used extensively these days to solve many of the problems that directly face or captures the environmental or spatial data which keeps changing quite rapidly. The devices contained within IOT network are quite fragile or gets degraded quite rapidly for various reasons leading to heavy performance degradation. The performance of the IOT networks also gets degraded due to existence of heterogeneity and due the need to undertake protocol conversion quite rapidly.

Many layers exits in each of the IOT based network and completely different types of computing is undertaken. The kind of devices used within the network greatly varies commencing form a simple sensor to a controller, RESTful services server, and a gateway to a remote cloud computing system. The protocols used in each of the layer are quite different. Need for use of gateways is a frequent requirement.

The performance of IOT networks greatly suffers due to many issues which must be properly mitigated to ensure that performance of an IOT networks is maintained as designed. One of the Major concerns is to mitigate the latency that exists in each of the layers of the IOT based network.

Performance of an IOT network can be measured in terms of response time, throughput, data transmission accuracy and latency.

Internet of Things (IoTs) is gaining increasing significance due to real-time communication and decision making capabilities of sensors integrated into everyday objects. IOT systems are complex that very small things like sensors are integrated with high end things like clouds. The IOT systems are designed for some specified level of performance. The bottle neck of IOT based systems must be detected and corrective actions are to be taken.

The performance of IOT based systems must be predicted while IOT system is up and running. Performance bottlenecks are to be found and corrective actions are taken in real time. Different mechanisms are to be incorporated into the IOT based systems to not only predict but also take corrective actions to maintain desired level of performance. However, performance computation and prediction becomes a significant challenge in IoT networks due to varying needs of applications coupled with the resource constrained nature of sensors.

RESTfull server is a kind of a Middle layers in a IOT based network. The server is implemented as WEB services server providing different kinds of services to the remote clients who are situated on remote cloud. Maximum latency is noticed in this layer especially when numbers of services are to be supported by the server.

Nowadays, high performance computing is more and more important for the economic and technological reasons. The high performance computing also becomes an indicator to measure the power of a country. Therefore, it is important and meaningful to improve the performance and universality of high performance computing. HPC is helpful to improve the performance of the system or to get the faster outputs. High-performance computing (HPC) is the use of parallel processing for running advanced application programs efficiently, reliably and quickly. The term applies especially to systems that function above a teraflop or 10^{12} floating-point operations per second. The term HPC is occasionally used as a synonym for supercomputing, although technically a supercomputer is a system that performs at or near the currently highest operational rate for computers. Some supercomputers work at more than a petaflop or 10^{15} floating-point operations per second. High Performance Computing most generally refers to the practice of aggregating computing power in a way that delivers much higher performance than one could get out of a typical desktop computer or workstation in order to solve large problems in science, engineering, or business.

IOT Technologies are being sandwiched with High performance computing for improving the performance of the IOT Networks. In this paper a high performance computing architecture is presented that help minimizing the latency at the services layer thereby improving the performance of the network drastically.

2. PROBLEM DEFINITION

An IOT system involves in different layers of networking and each layer of networking contributes to some latency in effecting the communication through the layer. An IOT network must perform in real-time. The total time required for either to transmit data or receive a request from the user and respond to the query must be as minimum as possible. One has to minimize the latency at each of the layer especially in the services layer such that the performance of an IOT network is optimized.

PROTOTYPE MODEL

IOT network typically is built using different layers of networking that include Device layer, controller layer, Restful services layer, Gateway layer and cloud computing Layer. A proto-type IOT network that was developed to sense the departure and control an AC and a FAN unit and send the status data which is time stamped to be stored in remote storage location preferably within cloud computing system. Users are allowed to make requests from remote locations to know the status of the devices or control the working of the devices.

At each of the layer data or request reception and data or response transmission takes place. Certain amount time is combined at intermittent stage of transmission of data one either forward or backward flow. Typical networking diagram of the proto-type IOT network is shown in Figure 1. The details of the equipment used for networking are shown in the Table 1. It cane bee seen that a different technology is used which include sensing, controlling, Service rending, heterogeneous communication and storing and retrieving the data from databases.

Table 1 : Node Identification within Proto type IOT network

S. No	DEVICE NUMBER	NAME OF THE DEVICE	S. No	DEVICE NUMBER	NAME OF THE DEVICE
1	NOO1	FAN	7	N007	RESTFUL SERVER
2	N002	Light	8	N008	ACCESS POINT
3	N003	AC	9	N009	ROUTER
4	N004	CLUSTER1	10	N010	GATEWAY
5	N005	CLUSTER2	11	N011	CLOUD
6	N006	CONTROLLER	12	N012	USER

The prototype model is primarily aimed at sensing the temperature in the neighborhood and controls the functioning of a FAN and an AC. Typically the application is implemented related to a home automation system.

The prototype model is used as an experimentation model for determining the performance and also modifying the same to shown how the performance can be improved by either changing the topologies or adding the interfaces.

3. RELATED WORK

High performance computing requirements have been felt necessary for building clouds as the customers requires fast response of the order of few seconds. Several investigations have been presented explaining the way HPC is used to build cloud related infrastructures and the way the services or

rendered by those cloud providers. The contributions made by the researchers helped many to know the way HPC can be included into the overall framework for providing the cloud computing based services.

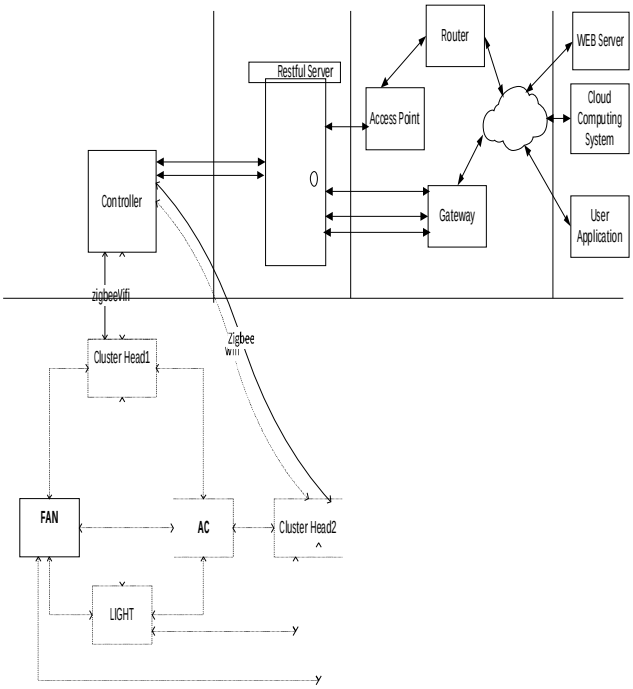

Figure 1: Typical IOT prototype Network

The performance of an IOT network is dependent on the way the interaction takes place between the Cloud computing system and the gateways that connect the rest of the system. There are many interoperability issues that must be addressed have been presented by Tharam Dillon et al,. [1].

Applications that must be built related to automotive engineering, Aerospace, High performance computing (HPC) is becoming, entertainment, business analytics, manufacturing, oil and Gas exploration, pharmaceutical development, drilling and mining etc. needs high processing speeds. These kinds of systems must be modelled simulated extensively to identify the risks. These kinds of system are designed and developed to run on super-computing systems which have many processors and fast interconnect between the processors. This kind of a system does not support elastic provisioning and dynamic scalability. These systems cannot be tailored for meeting the processing demands from time to time. Cloud computing offers various aspects that include provision, scalability, and dynamic adaptability etc. which are required to be supported along with high performance computing. Moustafa AbdelBaky and Manish Parashar et al., [2] have presented how HPC can be implemented within cloud computing systems so that all the desired computing requirements can be built along with supporting HPC.

Virtualisation is the key concept implemented within cloud computing for making available different kinds of computing resources on shared basis to different users. Many enterprise wide applications have been delayed on cloud computing platforms all of which have suffered from performance point of view even though most of the cloud computing features such as scalability, elasticity and schedulability. Albert Reuther et al., [3] have presented an analysis how HPC can be implemented in virtualised environment. They have shown how virtualisation can be added into existing HPC platforms.

HPC is quite required for implementing many scientific, industrial and research based applications. HPC is expensive due to the requirement of investment as a result; HPC is used for implementing large scale systems. Giuseppe Carlino, Rossella Prandi et al., [4] have implemented a project called " HPC Cloudpills" in which they have shown how HPC can be implemented within the clouds, leveraging the benefits of cloud computing, especially in reducing the cost of implementing HPC for middle range applications at the least cost possible.

Victor Eijkhout [5] in his book on "'Introduction to High Performance Scientific Computing" have HPC architectures, frameworks and different types of HPC implementations that one can use for implementing different types of applications.

Didier El Baz [6] has presented the connection between IoT and HPC. New paradigms and devices that can be used with HPC implemented within IOT layers have been presented. He also has presented the how IOT can be supported with HPC for implementing applications such as smart building management, smart logistics and smart businesses. He in his presentation has discussed several combinatorial and optimisation related problems that must be investigated so as to implement HPC within IoT.

Li Luxingzi [7] in his thesis titled "High performance computing applied to Cloud computing" submitted to Lapland University of applied science have details the way HPC can be implemented within the Cloud. He has detailed different challenges one must meet for implementing HPC within Cloud computing system.

Performance is the key issue when it comes to IOT networks. Sushma Satpute et al., [8] presented that the performance of the IOT networks generally suffers due to the need to deal with much of unwanted data than the actual data that must be transmitted. Maintenance of redundancy within the IOT networks to make the network fault tolerance also leads to poor performance. They have also suggested including another layer within the IOT network, actually called Local IOT controller layer for improving the performance of the IOT network.

Dr. Hassan Rajaei et al., [9] have presented the way HPC as a service can be implemented for a user. They have shown how education can be imparted by implementing HPC as a service within Cloud computing system.

D. BoobalaMuralitharan1 et al., [10] different ways of optimising the performance of cloud computing systems

when HPC is implemented as a part of the cloud. The process of optimisation of performance when HPC is offered as service is demonstrated through implementation of a different kind of task of scheduling.

Akhila Prabhakaran et al., [11] have presented the way HPC is implemented @ IISc Bangalore for facilitating research. The HPC centre @IIsc centre facilitates OpenMP, MPI, CUDA and OpenCL based HPC jobs for academic research. They have carried cost benefit analysis of providing HPC through cloud in comparison to the stand-alone HPC. They have presented that in-house supercomputing is cheaper compared to cloud computing. While that being the case the, HPC implemented @ IISc does not support the concepts like provisioning, scheduling etc.

Stephen Lien Harrell et al., [12] have reported that a set of applications have been tested by a set of students on a cloud or on the in-house clusters which are built on HPC platforms in relation to the budget provided for the purpose. They have arrived at different strategies that aim to use Standalone HPC and Cloud computing (Hybrid use of stand-alone HPC and cloud computing.

Many architectures and adaptive frameworks have been presented in the literature that aims at providing proven connection between the IOT devices and IOT systems. Service oriented architecture is seen to me most suitable architecture that suiting to the implementation of IOT based networks and systems. The SOA based architecture aims to provide loosely coupled systems to leverage the use and reuse of IoT services at the middle layer to minimise the system integration problems. Despite the flexibility offered through use of middle layer, the issues related to integrating, scaling and ensuring the resilience still persisted. The main reason being lack of intelligent connection aware framework that is needed to support interaction among things existing in the IoT based networks. Onoriode Uviase et al., [13**]** have suggested implementation of micro services based middleware that implement intelligent API layer that implements various components that includes external service assembler, service auditor, service monitor and service router component to coordinate service publishing, subscription, decoupling and service combination within the architecture.

An number of performance improvement techniques have been presented in the literature [14] [15] [16] [17] [18] [19] [20] [21] [22] [23] [24] [25]that include performance monitoring of lathes through IOT, IOT traffic flows, performance monitoring UPS Batteries through IoT, Implementation of asymmetric processing on multi core processors to implement IOT applications on GNU/Linux framework, urban climate monitoring through IOT, testing of message scheduling middleware algorithm with SOA for message traffic control in IoT environment, Development of hybrid execution service oriented architecture (HESOA) to reduce response time for IoT application, and Automation of an IoT hub using artificial intelligence techniques,

4. INVESTIGATIONS AND FINDINGS

4.1 Overview on High Performance Computing

The requests from the remote clients have to be served by the RESTful services server. The request must be serviced as fast as possible such that the SLA conditions are met.

Due to the requirements of heavy speeds of processing there is a requirement of implementing parallel processing with the RESTful services layer

Instruction pipelining, has been in use quite for some time for enhancing the speed of processing by a Mono processor. Based on the concept of pipelining, parallel processing has been pursued.

Three Categories of parallel processing has been investigated

- Single Instruction, Multiple Data (SIMD) system: A single machine instruction controls the simultaneous execution of a number of processing elements on a lockstep basis. Each processing element has an associated data memory, so that each instruction is executed on a different set of data by the different processors. Vector and array processors fall into this category
- Multiple Instruction, Single Data (MISD) system: A sequence of data is transmitted to a set of processors, each of which executes a different instruction sequence. This structure has never been implemented.
- Multiple Instruction, Multiple Data (MIMD) system: A set of processors simultaneously execute different instruction sequences on different data sets. SMPs, clusters, and NUMA systems fits into this category

With the MIMD organization, the processors are general purpose; each is able to process all of the instructions necessary to perform the appropriate data transformation. Symmetric Multi-processing is a kind of MIMD organisation that is cost effective that can be easily implemented over quadra core processor

In an SMP, multiple processors share a single memory or a pool of memory by means of a shared bus or other interconnection mechanism. A distinguish feature is that the memory access time to any region of memory is approximately the same for each processor.

A symmetric multiprocessor (SMP) can be defined as a standalone computer system with the following characteristic:

1. There are two or more similar processor of comparable capability.
2. These processors share the same main memory and I/O facilities and are interconnected by a bus or other internal connection scheme.

3. All processors share access to I/O devices, either through the same channels or through different channels that provide paths to the same device.
4. All processors can perform the same functions.
5. The system is controlled by an integrated operating system that provides interaction between processors and their programs at the job, task, file and data element levels. The operating system of a SMP schedules processors or thread across all of the processors.

SMP has potential advantages that include performance, availability, internal Growth and sealing

A system with multiple processors will perform in a better way than one with a single processor of the same type if the task can be organized in such a manner that some portion of the work done can be done in parallel. The availability of SMP is high since all the processors can perform the same function in a symmetric multiprocessor, the failure of a single processor does not stop the machine. Instead, the system can continue to function at reduce performance level. The performance of a SMP can be incrementally enhanced by adding additional processors. Typical architecture of a SMP system is shown in the Figure 2.

In the diagram shown above there are two or more processors. Each processor is self-sufficient, including a control unit, ALU, registers and cache. Each processor has access to a shared main memory and the I/O devices through an interconnection network. The processor can communicate with each other through memory (messages and status information left in common data areas). It may also be possible for processors to exchange signal directly. The memory is often organized so that multiple simultaneous accesses to separate blocks of memory are possible. In some configurations each processor may also have its own private main memory and I/O channels in addition to the shared resources.

The organization of multiprocessor system can be classified as Time shared or common bus, multiport memory and control Central control unit.

Time shared bus is the simplest mechanism for constructing a multiprocessor system. The bus consists of control, address and data lines. The time shared bus organisation must address the issues related to addressing which must address the modules on the bus to determine the source and destination. The bus arbitration system should allow any of I/O modules to temporarily function as a master. The bus arbitration method should allow for competing to get bus control using some sort of priority scheme. The time sharing of the bus must be achieved through locking out all the devices while one of the devices is given control of the bus. The bus organisation is simple, flexible and reliable,

The main drawback of the bus organization is performance. The speed of the system is limited by bus cycle time. To improve performance, each processor can be equipped with local cache memory. The use of cache leads to a new problem which is known as cache coherence problem. Each local cache contains an image of a portion of main memory. If a word is altered in one cache, it may invalidate a word in another cache. To prevent this, the other processors must perform an update in its local cache.

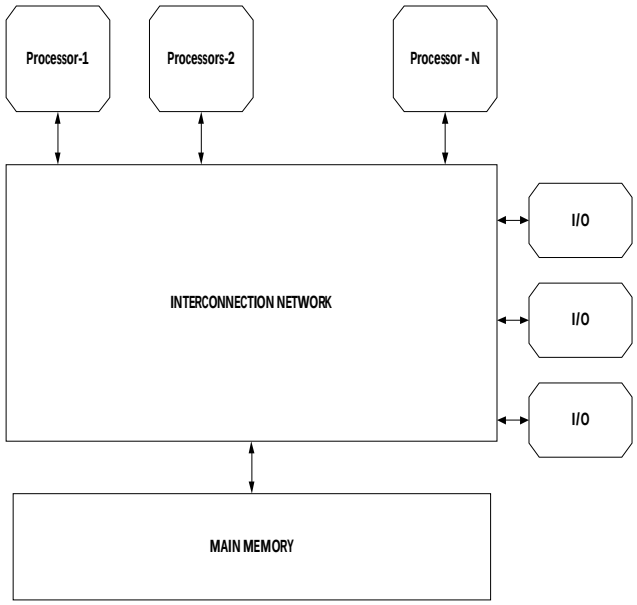

Figure 2: Typical architecture of SMP system

The multiport memory approach allows the direct, independent access of main memory modules by each processor and IO module. The multiport memory approach is more complex than the bus approach, requiring a fair amount of logic to be added to the memory system. Logic associated with memory is required for resolving conflict. The method often used to resolve conflicts is to assign permanently designated priorities to each memory port.

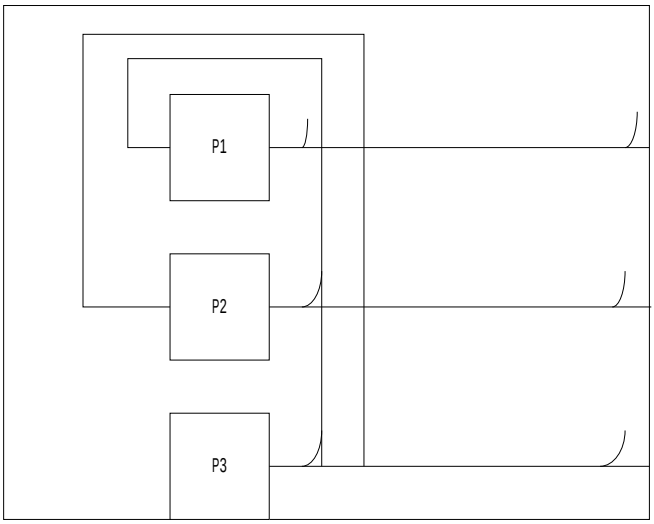

Figure 3: Cross Interconnection Network

Crossbar switch is a versatile switching network. It is basically a network of switches. Any module can be connected to any other module by closing the appropriate switch. Such networks, where there is a direct link between all pairs of nodes are called fully connected networks.

In a fully connected network, many simultaneous transfers are possible. If *n* sources need to send data to *n* distinct destinations then all of these transfers can take place concurrently. Since no transfer is prevented by the lack of a communication path, the crossbar is called a no blocking switch.

The crossbar interconnection network is shown in the Figure 3, a single switch is shown at each cross point. In actual multiprocessor system, the paths through the crossbar network are much wider.

If there are modules in a network, than the number of cross point is in a network to interconnect modules. The total number of switches becomes large as increases. In a crossbar switch, conflicts occur when two or more concurrent requests are made to the same destination device. These conflicting requests are usually handled on a predetermined priority basis.

The crossbar switch has the potential for the highest bandwidth and system efficiency. However, because of its complexity and cost, it may be cost effective for a large multiprocessor system.

The networking of the multiple processors is achieved through cross bar connections due heavy performance reasons. The bus and crossbar systems use a single stage of switching to provide a path from a source to a destination.

In multistage network, multiple stages of switches are used to setup a path between source and destination. Such networks are less costly than the crossbar structure, yet they provide a reasonably large number of parallel paths between source and destinations.

A three-stage network called a shuffle network that interconnects eight modules is shown in the Figure 4. The term "shuffle" describes the pattern of connections from the outputs of one stage to the inputs of the next stage. The switchbox in the figure is a switch that can route either input to either output. If the inputs request distinct outputs, they can both be routed simultaneously in the straight through or crossed pattern. If both inputs request the same output, only one request can be satisfied. The other one is blocked until the first request finishes using the switch.

A network consisting of stages can be used to interconnect modules. In this case, there is exactly one path through the network from any module to any module. Therefore, this network provides full connectivity between sources and destinations. Many request patterns cannot be satisfied simultaneously. For example, the connection from *P2* to *P7* can not be provided at the same time as the connection from *P3* to *P6*. A multistage network is less expansive to implement than a crossbar network. If nodes are to be interconnected using this scheme, then we must use ($s= \log_2 n$) stages with n/2 switches per stage.

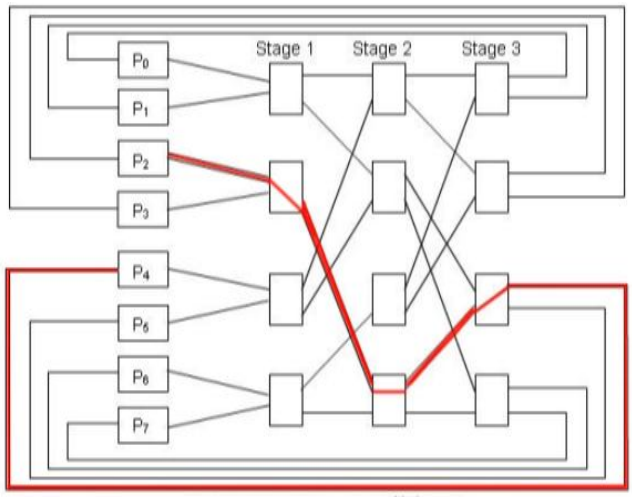

Figure 4: Multi-Stage HPC implementation

A 8 Processor computer is selected that connects the processing units using the Multi-Stage network keeping in view of achieving higher performance of the RESTful services server

4.2 Performance Assessment of Prototype

The performance assessment of an IOT network can be undertaken using several parameter that include response time, throughput etc. For the project response time is used as the parameter for assessing the performance of an IOT network. However the performance assessment of an IOT network is of composite in nature.

The performance of an IOT network is as such bidirectional. In one direction, data packets move from device to storage level in cloud computing or data storage and retrieval setup. In the other direction, a service request is initiated from a remote client and the request is moved to the level of a controller and the response to the request is transmitted back to the client who initiated the request.

In the first direction, to track response time, every data packet has to be identified along with device which initiated the data packet. All the devices must be identified with an Identification number, its longitude, latitude and any salient features required. The data packets moves across the network and the time stamp of receiving and transmitting data must be recorded to identify latencies at each of the location. Following tables needs to be maintained to understand the time taken to receive, store and transmission of the data at every stage.

Packet numbers are generated by the source that captures the data. At every node both reception and transmission time stamps are collected and added to the received packets. At every node its own header shall be added to the data that it has received. The data packet received shall have all the headers of the nodes that have been traversed before receiving at a node. Latency s computed considering the received and transmitted date and time. The last node is the server at which the data is received and stored. The final data packet thus will have several rows of the data that gives the details of data transmission and reception. The total time taken for transmitting the can be computed by knowing the date and time of receiving the data at the starting node and date and time of receiving the data at the final node. Latency time can be computed by adding latency time at each and every node. Time stamps at the device level can be gotten through of an RTC connected to device and through system time maintained at different servers in the network.

A typical IOT network is shown in the Figure 1. The elements identified with a node number and the function of each node is shown in the Table 1.

The data transmission details are captured into the Table 2. This will allow us to get the response time for moving the data from originated device till the data is received at storage server. From the Table 2 it could be seen that it takes 1 Second and 10 Micro Seconds of the time to move the packet from the Device to the storage server. Thus it can be construed that the response time is 1 second and 10 Micro seconds in the bottom up approach.

Table 2: Total response time Calculations (Micro seconds

Serial	Type of transaction	Number of transaction	Response time for transaction	Total Time required
1	Data Transfer	10	100,010	1,000,100
2	User Transfers	200	50	10,000
	Total	210	100,060	1,010,100
		Average	476	4810

In similar lines we need to consider the response time when a request for the device status is initiated by a user who is interfaced through internet. In this case the request is initiated from the user, reaches the RESTful service server to controller and back to the user. The total time it takes to traverse from the user and back to the user is called response time. The typical response time computations when a request is initiated by the user and the response received by the user is shown in the Table 3. The response is computed as 50 Micro seconds for initiating a request till the response is received by the user.

In a typical scenario one can expect 200 data related transactions that must be answered due to user requests. Using this as the basis the response computations are as shown in the Table 4. From the table it can be seen that average response time of the typical IOT network works out to 4810 Micro Seconds which is generally not acceptable. The acceptable response time form an IOT network is around 0.3 seconds. Thus it becomes necessary to improve on the response time of the IOT network through proper mechanisms.

It can be seen from the Table 4 that highest latency is at the RESTful services level and the next highest latency is at the gateway level. Thus it becomes necessary to improve the latency at the RESTful serves level as the latency issues related to the cloud computing shall be taken care of by cloud vendor.

From Table 4 it can be seen that the real bottleneck is at the RESTful services level at which location the response time must be improved by enhancing the processing power at that layer.

4.3 Building SMP within IOT Based Network and Performance Assessment

An 8 CPU server is selected for implementing an SMP system which is connected through a Multi-Stage network. Each processor is assigned to provide service. All 8 services can be simultaneously processed so that any service requested by a remote user is processed with the same time even though 8 simultaneous requests have been initiated by the remote user.

A service association to a sever is configured through UNIX operating systems through which parallel processing is implemented. A light weight server is developed that receives the request from the remote user and makes a request to the OS to schedule the request to be processed by a server that is associated with the kind of service that must be processed by it.

The IOT network is made operational and the response time calculations are shown in the Table 5, Table 6 and table 7, it can be seen that the response time of the IOT network has been tremendously improved.

Table 5: Total response time Calculations (Micro seconds

Serial	Type of transaction	Number of transaction	Response time for transaction	Total Time required
1	Data Transfer	10	100,010	1,000,100
2	User Transfers	200	29	5,800
	Total	210	100,039	1,005,900
		Average	476	4790

The average response time considering 200 user requests and 10 data transfer request work out to 4790 Micro seconds which is an improvement over 4890 micro seconds required for processing a mixture of data transactions and the processing required to answer user requests. The average time required for processing the data requests works out to 1 second and 10 Micro seconds while the time required to process the user requests works out to 10,000 Micro seconds which is clear reduction from mono server based RESTful service implementation.

5. CONCLUSIONS

RESTfull service based processing is the real bottleneck in undertaking an IOT based processing. The Load on an IOT network must be considering the data that flows from bottom to TOP and the requests that floe from Top to Bottom. The flow of data happens in both the direction to answer the user requests initiated by the user. The response time greatly improves through implementing the RESTful services through High Performance computing.

REFERENCES

[1] Tharam Dillon and Chen Wu and Elizabeth Chang, Cloud Computing: Issues and Challenges, 24th IEEE International Conference on Advanced Information Networking and Applications, 2010, IEEE Computer Society, 1550-445X/10 $26.00 © 2010 IEEE https://doi.org/10.1109/AINA.2010.187

[2] Moustafa AbdelBaky and Manish Parashar, Hyunjoo Kim, Kirk E. Jordan, Vipin Sachdeva, James Sexton, Hani Jamjoom, and Zon-Yin Shae, Gergina Pencheva, Reza Tavakoli, and Mary F. Wheeler, Enabling High computing Performance as a service, IEEE Computer Society, 0018-9162/12/$31.00 © 2012 IEEE

[3] Albert Reuther, Peter Michaleas, Andrew Prout, and Jeremy Kepner, HPC-VMs: Virtual Machines in High Performance Computing Systems, 2012 IEEE Conference on High Performance Extreme Computing, 978-1-4673-1576-0/12/$31.00 ©2012 IEEE https://doi.org/10.1109/HPEC.2012.6408668

[4] Pietro Ruiu, Olivier Terzo and Alberto Falzone, Paolo Maggi, HPC CloudPills: on-demand deployment and execution of HPC application in cloud environments, Ninth International Conference on P2P, Parallel, Grid, Cloud and Internet Computing, 978-1-4799-4171-1/14 $31.00 © 2014 IEEE

[5] Victor Eijkhout and Edmond Chow, Robert van de Geijn, Text Book - Introduction to High-Performance Scientific Computing, 2nd Edition, 2014, Distributed under a Creative Commons Attribution, The Saylor Foundation http://www.saylor.org.

[6] CNRS, LAAS, IoT and the Need for High Performance Computing, International Conference on Identification, Information and Knowledge in the Internet of Things, 2014

[7] Li Luxingzi, Thesis - High performance computing applied to cloud computing, Technology, Communication and Transport Programme in Information Technology, Lapland University of applied sciences, 2015

[8] Sushma Satpute, Dr. Bharat Sing Deora, Improving performance of Internet of Things by Using local IOT controller unit, International Conference on Green Computing and Internet of Things (ICGCIoT), 2015, 978-1-4673-7910-6/15/$31.00_c 2015 IEEE https://doi.org/10.1109/ICGCIoT.2015.7380672

[9] Dr. Hassan Rajaei, Bowling, Mr. Saba Jamalian, HPC as a Service in Education, ASEE's 123rd Annual conference and expositions, 2016

[10] D. Boobala Muralitharan, S. ArockiaBabi Reebha, D. Saravanan3, Optimization of performance and scheduling of HPC applications in cloud using cloudsim and scheduling approach, International Conference on IoT and Application (ICIOT), 2017 https://doi.org/10.1109/ICIOTA.2017.8073634

[11] Akhila Prabhakaran, Lakshmi J, Cost-benefit Analysis of Public Clouds for offloading in-house HPC Jobs, 11th IEEE International Conference on Cloud Computing, 2018 https://doi.org/10.1109/CLOUD.2018.00015

[12] Stephen Lien Harrell, Andrew Howard, Hybrid HPC Cloud Strategies from the Student Cluster Competition, IEEE 11th International Conference on Cloud Computing, 2018 https://doi.org/10.1109/CLOUD.2018.00031

[13] Onoriode Uviase, Gerald Kotonya, IoT Architectural Framework: Connection and Integration Framework for IoT Systems, First workshop on Architectures, Languages and Paradigms for IoT EPTCS 264, 2018, pp. 1–17 https://doi.org/10.4204/EPTCS.264.1

[14] Murty, A.S.R., Teja, K., Naveen, S., Lathe performance monitoring using IoT, 2018 International Journal of Mechanical Engineering and Technology 9(4), pp. 494-501© IAEME Publication

[15] Rambabu, K., Venkatram, N., Traffic flow features as metrics (TFFM): Detection of application layer level DDOS attack scope of IOT traffic flows, 2018, International Journal of Engineering and Technology (UAE) 7(2), pp. 203-208 © 2018 https://doi.org/10.14419/ijet.v7i2.7.10293

[16] K., Prabu, A.V., Sai Prathyusha, M., Varakumari, S., Performance monitoring of UPS battery using IoT Vijaya Manasa, 2018 International Journal of Engineering and Technology(UAE), 7, pp. 352-355© 2018 https://doi.org/10.14419/ijet.v7i2.7.10717

[17] Poonam Jain, S., Pooja, S., Sripath Roy, K., Abhilash, K., Arvind, B.V. Implementation of asymmetric processing on multi core processors to implement IOT applications on GNU/Linux framework,2018, International Journal of Engineering and Technology (UAE), 7, pp. 710-713© 2018. https://doi.org/10.14419/ijet.v7i2.7.10928

[18] Rajasekhara Naidu, G., Venkat Ram, N., Urban climate monitoring system with IoT data analytics

2018 International Journal of Engineering and Technology(UAE), 7(2), pp. 5-9
https://doi.org/10.14419/ijet.v7i2.20.11732

[19] Gupta, P., Satyanarayan, K.V.V., Shah, D.D., Development and testing of message scheduling middleware algorithm with SOA for message traffic control in IoT environment, International Journal of Intelligent Engineering and Systems 11(5), pp. 301-313
https://doi.org/10.22266/ijies2018.1031.28

[20] Gupta, P., Satyanarayan, K. V. V., Shah, D.D., IoT multitasking: Development of hybrid execution service oriented architecture (HESOA) to reduce response time for iot application, 2018Journal of Theoretical and Applied Information Technology 96(5), pp. 1398-1407

[21] Yasaswini, A., DayaSagar, K.V., Shri Vishnu, K., HariNandan, V., Prasadara Rao, P.V.R.D., Automation of an IoT hub using artificial intelligence techniques, 2018 International Journal of Engineering and Technology(UAE), 7(2), pp. 25-27
https://doi.org/10.14419/ijet.v7i2.7.10250

[22] Ramaiah, C.H., Parimala, V.S., Kumar, S.P., Reddy, G.B., Rahul, Y. Remote monitoring through tab, 2018 International Journal of Mechanical Engineering and Technology 9(1), pp. 490-498

[23] Sai, Y. S., Kumar, K.K., Internet of things and its applications, 2018 International Journal of Engineering and Technology(UAE) 7, pp. 422-427
https://doi.org/10.14419/ijet.v7i2.7.10758

[24] N.Saritha Devi, K.S.R.Raju, A.Madhu and R.Raja Sekhar, Safety and Security for School children's Vehicles using GPS and IoT Technology, International Journal of Advanced Trends in Computer Science and Engineering, Vol. 7, Iss. 8, pg. 91-95
https://doi.org/10.30534/ijatcse/2018/03762018

[25] S.V.R.K.Rao, M.Saritha Devi, A.R.Kishore and Praveen Kumar, Wireless sensor Network based Industrial Automation using Internet of Things (IoT), International Journal of Advanced Trends in Computer Science and Engineering, Vol. 7, Iss. 6, pg. 82-86

Table 3: response time computation – Moving data from device to storage server

Data Packet Number	Node Number	Longitude	Latitude	Receiving date	Receiving Time	Transmission date	Transmission Time	Latency Time	Data	Receiving Node
1.	N001	80.62°36’13.0’’E	16.44°19’10’’N	1/3/19	14:04:32:45:01	1/3/19	14:04:32:45:04	00:00:00:00:03	F	N004
2.	N004	80.62°36’13.1’’E	16.44°19’11’’N	1/3/19	14:04:32:45:04	1/3/19	14:04:32:45:08	00:00:00:00:04	F	N006
3.	N006	80.62°36’13.1’’E	16.44°19’11’’N	1/3/19	14:04:32:45:08	1/3/19	14:04:33:45:11	00:00:00:00:03	F	N007
4.	N007	80.62°36’13.2’’E	16.44°19’12’’N	1/3/19	14:04:33:45:11	1/3/19	14:04:33:45:43	00:00:00:00:33	F	N010
5.	N010	84.62°78’13.20’’E	18.45°19’40’’N	1/3/19	14:04:33:45:43	1/3/19	14:04:33:45:47	00:00:00:00:04	F	N011
6.	N011	84.62°78’13.20’’E	18.45°19’40’’N	1/3/19	14:04:33:45:47	1/3/19	14:04:32:45:51	00:00:00:00:04	F	N012
7.	N012	84.62°78’13.20’’E	18.45°19’40’’N	1/3/19	14:04:33:45:51	1/3/19				

Table 4 response time computation – Moving data from User to Back

Data Packet Number	Node Number	Longitude	Latitude	Receiving date	Receiving Time	Transmission date	Transmission Time	Latency Time	Data	Receiving Node
1.	N012	80.62°36’13.0’’E	16.44°19’10’’N	1/3/19	14:04:32:45:01	1/3/19	14:04:32:45:04	00:00:00:00:03	B,25C	N011
2.	N011	80.62°36’13.1’’E	16.44°19’11’’N	1/3/19	14:04:32:45:04	1/3/19	14:04:32:45:08	00:00:00:00:04	B,25C	N010
3.	N010	80.62°36’13.1’’E	16.44°19’11’’N	1/3/19	14:04:32:45:08	1/3/19	14:04:33:45:11	00:00:00:00:03	B,25C	N007
4.	N007	80.62°36’13.2’’E	16.44°19’12’’N	1/3/19	14:04:33:45:11	1/3/19	14:04:33:45:13	00:00:00:00:02	B,25C	N006
5.	N006	84.62°78’13.20’’E	18.45°19’40’’N	1/3/19	14:04:33:45:13	1/3/19	14:04:33:45:17	00:00:00:00:04	B,25C	N007
6.	N007	84.62°78’13.20’’E	18.45°19’40’’N	1/3/19	14:04:33:45:17	1/3/19	14:04:32:45:43	00:00:00:00:22	B,25C	N010
7.	N010	80.62°36’13.1’’E	16.44°19’11’’N	1/3/19	14:04:32:45:43	1/3/19	14:04:33:45:47	00:00:00:00:03	B,25C	N011
8.	N011	80.62°36’13.1’’E	16.44°19’11’’N	1/3/19	14:04:32:45:47	1/3/19	14:04:32:45:53	00:00:00:00:06	B,25C	N012

Table 6: response time computation – Moving data from device to storage server

Data Packet Number	Node Number	Longitude	Latitude	Receiving date	Receiving Time	Transmission date	Transmission Time	Latency Time	Data	Receiving Node
1.	N001	80.62°36’13.0’’E	16.44°19’10’’N	1/3/19	14:04:32:45:01	1/3/19	14:04:32:45:04	00:00:00:00:03	F	N004
2.	N004	80.62°36’13.1’’E	16.44°19’11’’N	1/3/19	14:04:32:45:04	1/3/19	14:04:32:45:08	00:00:00:00:04	F	N006
3.	N006	80.62°36’13.1’’E	16.44°19’11’’N	1/3/19	14:04:32:45:08	1/3/19	14:04:33:45:11	00:00:00:00:03	F	N007
4.	N007	80.62°36’13.2’’E	16.44°19’12’’N	1/3/19	14:04:33:45:11	1/3/19	14:04:33:45:13	00:00:00:00:02	F	N010
5.	N010	84.62°78’13.20’’E	18.45°19’40’’N	1/3/19	14:04:33:45:13	1/3/19	14:04:33:45:17	00:00:00:00:04	F	N011
6.	N011	84.62°78’13.20’’E	18.45°19’40’’N	1/3/19	14:04:33:45:17	1/3/19	14:04:32:45:21	00:00:00:00:04	F	N012
7.	N012	84.62°78’13.20’’E	18.45°19’40’’N	1/3/19	14:04:33:45:21	1/3/19				

Table 7: response time computation – Moving data from User to Back

Data Packet Number	Node Number	Longitude	Latitude	Receiving date	Receiving Time	Transmission date	Transmission Time	Latency Time	Data	Receiving Node
1.	N012	80.62°36’13.0’’E	16.44°19’10’’N	1/3/19	14:04:32:45:01	1/3/19	14:04:32:45:04	00:00:00:00:03	B,25C	N011
2.	N011	80.62°36’13.1’’E	16.44°19’11’’N	1/3/19	14:04:32:45:04	1/3/19	14:04:32:45:08	00:00:00:00:04	B,25C	N010
3.	N010	80.62°36’13.1’’E	16.44°19’11’’N	1/3/19	14:04:32:45:08	1/3/19	14:04:33:45:11	00:00:00:00:03	B,25C	N007
4.	N007	80.62°36’13.2’’E	16.44°19’12’’N	1/3/19	14:04:33:45:11	1/3/19	14:04:33:45:13	00:00:00:00:02	B,25C	N006
5.	N006	84.62°78’13.20’’E	18.45°19’40’’N	1/3/19	14:04:33:45:13	1/3/19	14:04:33:45:17	00:00:00:00:04	B,25C	N007
6.	N007	84.62°78’13.20’’E	18.45°19’40’’N	1/3/19	14:04:33:45:17	1/3/19	14:04:32:45:21	00:00:00:00:04	B,25C	N010
7.	N010	80.62°36’13.1’’E	16.44°19’11’’N	1/3/19	14:04:32:45:21	1/3/19	14:04:33:45:24	00:00:00:00:03	B,25C	N011
8.	N011	80.62°36’13.1’’E	16.44°19’11’’N	1/3/19	14:04:32:45:24	1/3/19	14:04:32:45:30	00:00:00:00:06	B,25C	N012