International Journal on Recent Researches in Science, Engineering & Technology (IJRRSET)
A Journal Established in early 2000 as National journal and upgraded to International journal in 2013 and is in existence for the last 10 years. It is run by Retired Professors from NIT, Trichy. Journal Indexed in JIR, DIIF and SJIF.
Research Paper
Available online at: www.jrrset.com

ISSN (Print) : 2347-6729
ISSN (Online) : 2348-3105

Volume 7, Issue 4
April 2019.
JIR IF : 2.54
SJIF IF : 4.334
Cosmos: 5.395

Received on: 22.03.2019 Published on: 14.04.2019 Pages : 37 - 41

EVALUATING PERFORMANCE OF DYNAMIC SCALING MECHANISM USING FOG COMPUTING

[1]K. Aruna, [2]M. Poonguzhali, [3]R. Divya
[1]Assistant professor. [2,3]UG Scholars
Department of Information Technology
AVC college of Engineering, Mannampandal,Mayiladuthurai, Tamil Nadu, India
Email:avcce.aruna@gmail.com

ABSTRACT

Amount of internet users are increased day by day. the internet has been faced some problem based on the performance like network traffic, high latency, low scalability, low performance etc, in our project we done a process to improve the network performance and scalability for reducing the network traffic and also improve the scalability using fog computing . in project we have to plan to implement the fog node to produce interaction between the fog nodes. and specifically use algorithm to improve the scalability .the algorithm uses A mechanism to dynamically scale in/out the serving instances of the middle nodes in order to
make the whole M2M platform more scalable. Comparing the performance of our dynamic scaling mechanism with those based on a static and fixed pool of serving instances Inside the fog computing.
Keywords - scalability;fog computing;IOT.

Introduction

The Internet of Things (IoT) will become ubiquitous in the near future. Gartner, Inc. estimates that there will be 20 billion IoT/M2M devices connected to the Internet in 2020. It is foreseeable this huge amount of IoT/M2M devices will generate the data traffic to the IoT/M2M platforms normally deployed in the Cloud. Though more resources in the cloud may be allocated to alleviate such overloading issues, this research Proposes the alternative of utilizing “Fog Computing” to extend the scalability of IoT/M2M platforms in the cloud. When comparing the cloud and fog due to issues of data traffic and maintaining the performance of scalability. To Implement the scalability platform in the cloud has been investigated where the resources in the cloud can be scaled up or down according to the loading of cloud traffic. The scaling server automates the scale-in and scale-out procedure. Compared to virtual machine, containers provide lightweight virtual environment and thus are more suitable to achieve fast response in the fog network. Due to its locality and proximity, Fog computing, when compared to cloud computing, can provide lower latency and quicker response. In the Fog network, the topology could be hierarchical, which means the data from the end-users or the devices can be processed at several levels; each level carrying out specific data filtering and analytics and deciding whether to pass to the next level for further Fog/Cloud processing. In fog network, node to node transmission will be occurred and it will be visualized. We introduced two approaches to extend the scalability of M2M systems in the fog, it will be easily transmitted through the node but the number of M2M systems are transmitted through the clusters. To finding the shortest path in between the node to transmitted the data easily between the nodes. In this way of fog network, we using hierarchical method of implementation through the several layers of fog. However,

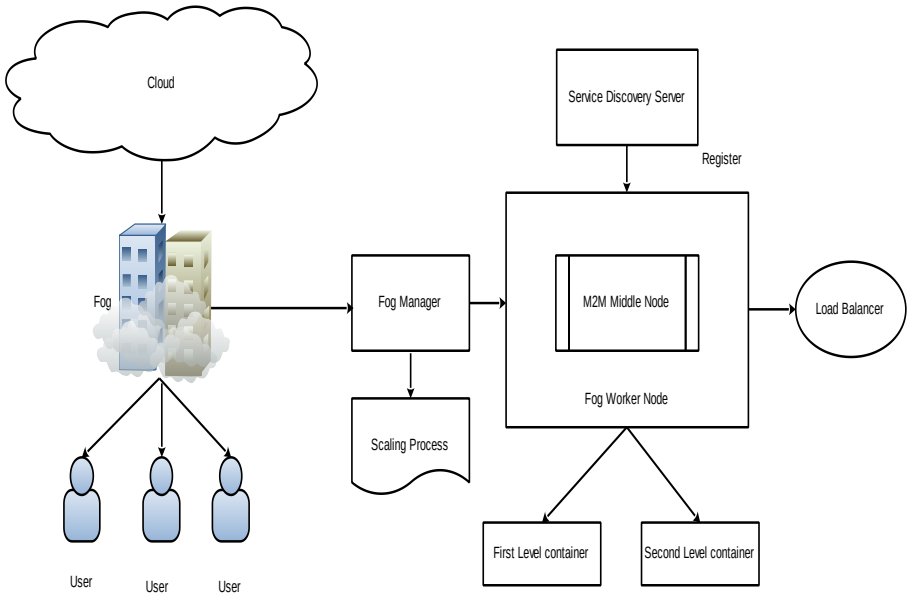
dealing with IoT traffic solely in the fog will be completely solve the scalability problems. To improve the scalability of IoT/M2M platform, the IoT traffic must be managed. This means Fog network as a continuum between the Cloud and the things must provide the scalable capacity, i.e. be able to flexibly and dynamically scale in/out the serving instances across Fog nodes to match with the incoming traffic and it will be reducing the network traffic and improving the performance of scalability. The rest of this paper is organized as follows. Section II introduces the background. Section III presents our use case. Section IV describes the design and implementation of our proposed hierarchical, dynamically scalable container-based Fog architecture in detail. Section V compares the proposed architecture with the static system. Finally, Section VI presents our conclusion and future work.

II. Background

In this section, we introduce the concept of Fog, oneM2M and their integration. In addition, how to solve scalability issues in the Fog is discussed.

Implementation

Fog computing intends to move the cloud capacity (compute, network, and storage) to the continuum from the cloud to the things. Due to its locality and proximity, Fog computing, when compared to cloud computing, can provide lower latency and quicker response .Therefore, Fog can be seen as an extension of the cloud. In the Fog network ,the topology could be hierarchical ,which means the data from the end-users or the devices can be processed at several levels ;each level carrying out specific data filtering and analytics and deciding whether to pass to the next level for further Fog/cloud processing.



When comparing the cloud and fog computing gives the system good elasticity under different traffic loads and doesn't create too much overhead with an additional scaling server. The scaling server automates the scale-in and scale-out procedure .compared to the static approach allows the system capacity to dynamically grow and shrink as needed. We leverage virtualization enabled by the container technology to create high scalable Fog-based M2M system. Compared to the virtual machines, containers provide lightweight virtual environment and thus are more suitable to achieve fast response in the Fog network. Due to this system, fog contains advancement of cloud computing .It will reduce data traffic, avoiding data loss and quicker response.

Experimentation

In this paper, Signature Driven Load Management (SigLM) algorithm will be used. It works by capturing system's signature like available RAM, current CPU bandwidth available and other resources .Once captured,that value is compared with default threshold value and accordingly load like incoming requests is

shifted to target node machine using Dynamic Wrap Technique works by considering source node as given by SigLM algorithm and makes some calculations to predict target nose to which the load is to be shifted.

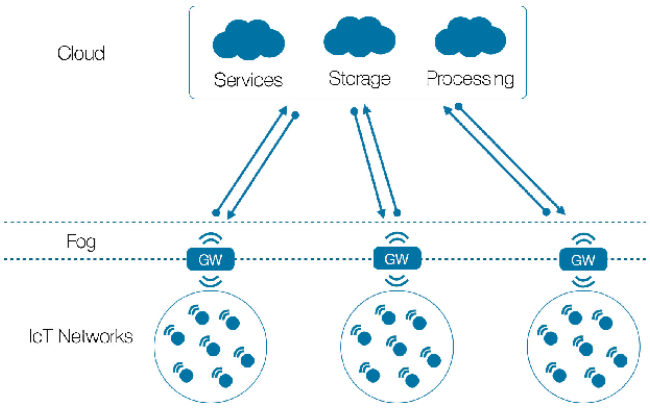
$V = \{v_1, ..., v_n\}$: nodes in the cloud system
t_i: a task that needs to be placed in the cloud system
W: signature sliding window
f_i: pre-filtering qualifying function for resource type $r_i \in R$
DHT: P2P signature lookup system

UpdateResourceSignature(V, $|W|$, DHT)
1. **for** every signature window $|W|$ **do**
2. **for** each node v_i in V **do**
3. **for** each resource attribute r_k in R **do**
4. Construct the resource signature Sig_{r_k}
5. Construct the index MBRs for Sig_{r_k}
6. Insert the MBRs into R-trees
7. Push Sig_{r_k} and its index into DHT

MatchTaskSignature(t_i, V, DHT)
1. Construct the MBRs for the load signature Sig_L of t_i
2. Send load matching request to DHT nodes
3. **for** each DHT node **do**
4. **for** each cloud node resource signature Sig_R **do**
4. flag = TRUE;
5. **for** each resource type Sig_{r_i} **do**
6. flag = flag $\wedge$ $f_i(Sig_{l_i}, Sig_{r_i})$ /* MBR matching*/
7. if (flag == TRUE)
8. insert the cloud node signature into a DTW list
9. return the DTW list to the initiating node for t_i
4. merge DTW lists received from all DHT nodes
7. **for** every node resource in the DTW List **do**
8. Invoke DTW algorithm to get a matching score α
9. Sort the DTW List based on α
10. **for** every node v_j in the sorted DTW list **do**
11. Invoke admission control func. between t_i and v_j
12. **if** admission control func. returns **TRUE**
13. Allocate t_i to v_j

A. Resource Monitoring

Open Fog Consortium was launched in 2015 that aimed to promote the concept of Fog Computing and define a set of system-level architectural framework. Open Fog introduced a major component in Fog Computing called Fog Node, which provides the computing, networking, storage and acceleration capacity for the end-devices.
The performance of resource discovery and subscribe newly discovered resources of interest to the first level of MN-CSE. In addition, it will keep monitoring the sensor data in the first level MN-CSE's resource tree. By performing data analysis, this AE can detect abnormal events and notifies the second level MN-AE. The performance of resource discovery and subscribe newly discovered resources of interest to the first level MN-CSE. In addition, it will keep monitoring the sensor data in the first level MN-CSE's resource tree. By performing data analytics, this AE can detect abnormal events and notifies the second level MN-AE.



B. Load Balancing

There are many common features between the Middle Nodesin oneM2M and the Fog Nodes in OpenFog. In addition to supporting the hierarchical processing model, oneM2M Middle Nodes resemble a localized IN that can offload the processing load of the IN normally residing in Cloud by providing compute, storage and network closer to the IoT/M2M devices. Consequently, Middle Nodes can assume the role of the Fog Nodes. The load balancer then will distribute the traffic to all the serving instances.In addition,a load balancer is deployed between IOT/M2M devices and the fog cluster to distribute all the incoming requests from the IOT/M2M devices to the first level MNAE/MN-CSE according to a choosen load balancing policy.

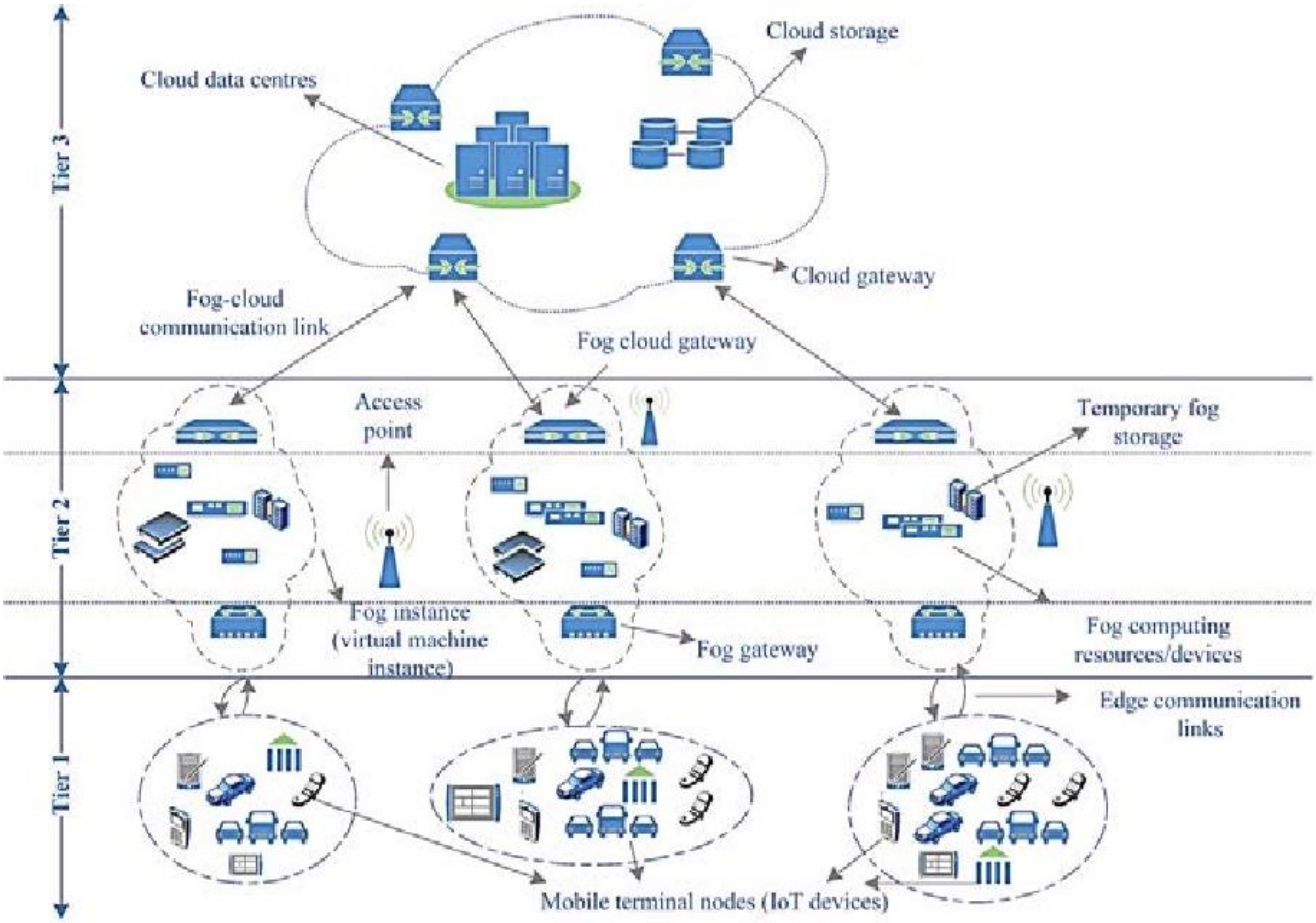


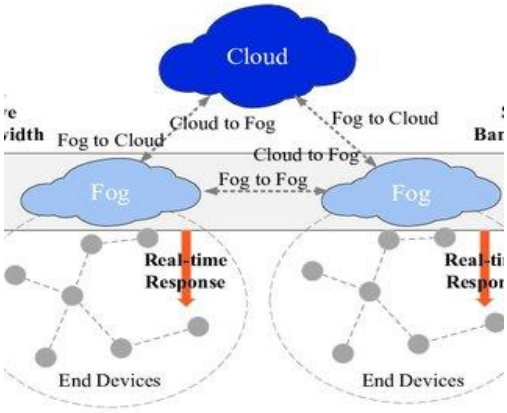
Fig. 2. Fog computing architecture

Clusters will be distributed the data due ro corresponding nodes and balanced the several nodes. It will be deployed at Iot/M2M devices it contains implemented as load balancer. As the router optimization, the nodes will be balanced due to corresponding nodes. The node will be balanced at load balancer because of using clusters. IOT/M2M devices are connected to the internet, we can't using clusters but number of IOT/M2M devices are connected together, using cluster. It will be distributed the nodes due to implemented by load balancing policy. This is the traditional approach where a fixed number of oneM2M Middle Node container instances is deployed among the Fog Nodes before the service starts to work. The load balancer then will distribute the traffic to all the serving instances.

Cloud setup

cloud computing can be efficient alternative to owing and maintaining computer resources and applications for many organization due to pay-as-you-go model and other characteristics includes on-demand,self-service,resource pooling and rapid elasticity. It will be implemented they contains many drawbacks like network traffic,data loss,reducing response time. Although both cloud and fog offer similar resources and services, the latter is characterized by low latency with a wider spread and geographically distributed nodes to

support mobility and real-time interaction. Cloud and fog computing shares overlapping features,but fog computing has additional attribures such as quicker response,maintaining scalability



Fog Manager

Fog can be seen as an extension of the cloud .In the Fog network ,the topology could be hierarchical ,which means the data from the end-users or the devices can be processed at several levels ; each level carrying out specific data filtering and analysis and deciding whether pass to the next level .

Performance analysis

In order to evaluate the performance of our proposed architecture, we compare it to the static approach with a fixed pool of serving instances. The scaling server automates the scale-in and scale-out procedure. To enhance our scaling server to be able to scale in/out the one M2M instances across different levels of the Fog hierarchy .The performance of our dynamic scaling mechanism with those based on a static and fixed pool of serving instances.

Conclusion and future work

The scalability hierarchical container-based Fog architecture with auto-scaling mechanism . We containerize the one M2M Middle Nodes using NS2 technology .Our approach leverages Fog networking to extends the scalability of the one M2M platform .The scaling server automates the scale-in and scale-out procedure .compared to the static approach allows the system capacity to dynamically grow and shrink as needed.

References

[1] Anand, F. Dogar, D. Han, B. Li, H. Lim, M. Machadoy, W. Wu, A. Akella, D. Andersen, J. Byersy, S. Seshan, and P. Steenkiste, “XIA: an architecture for an evolvable and trustworthy internet”, HotNets '11 Proceedings of the 10th ACM Workshop on Hot Topics in Networks, 2011.
[2] Seskar, K. Nagaraja, S. Nelson, and D. Raychaudhuri, “MobilityFirst future Internet architecture project”, AINTEC '11 7th Asian Internet Engineering Conference, 2011.
[3] D. G. Andersen H. Balakrishnan N. Feamster T. Koponen D. Moon and S. Shenker, “Accountable Internet Protocol (AIP)”, ACM SIGCOMM, 2008.
[4] F. Teraoka, S. Kanemaru, K. Yonemura, “ZNA: A network architecture for new generation network - Focusing on the session layer”, In Ubiquitous and Future Networks, 2011.