Load Balancing Method in Edge Computing

Marian Kyryk
Department of Telecommunication
Lviv Polytechnic National University
Lviv, Ukraine
mkyryk@gmail.com

Nazar Pleskanka
Department of Computer Aided Systems
Lviv Polytechnic National University
Lviv, Ukraine
n.pleskanka@gmail.com

Mariana Pleskanka
Department of Telecommunication
Lviv Polytechnic National University
Lviv, Ukraine
mariana__p.m.v.9@ukr.net

Petro Nykonchuk
Department of Telecommunication
Lviv Polytechnic National University
Lviv, Ukraine
petro.nykonchuk.gl@gmail.com

Abstract - This paper considers the concept of Edge computing, which is a distributed computing paradigm that is using to improve response times and save bandwidth and brings computation and data storage closer to the Edge location. The main principles of CDN architecture and global concept content delivery network (CDN) system were considered in this paper. The algorithm to solve the load balancing problem in the edge computing network was investigated in this paper.

Edge computing provides compute and storage resources with adequate connectivity (networking) close to the devices generating traffic.

Keywords – Edge computing, Edge Cloud, load balancing, QoS, data processing, content distribution.

I. Introduction

It seems that the term “cloud computing” only recently "entered into our life” and another computing model appeared on the horizon - this time it is edge computing (peripheral, or boundary computing). As a real example, we can say software as a service (SaaS) instances, such as Google Apps, AWS, FaceBook, and BigFlix, have been widely used in our daily life.

The first mention of IoT was in 1999 [1], and even then this concept began to quickly adapt to other areas such as health, home, environment and transport [2, 3]. Now with IoT, a large quantity of data is generated by things that we are using in everyday life. A lot of applications will be also deployed at the edge to consume these data in the nearest future. Data generated by people and machines can reach 500 zettabytes by 2019, as estimated by Cisco Global Cloud Index [4] And by 2020, 45% of IoT-created data will be stored, processed, analysed at the edge of the network. [5] Some IoT devices or applications might require very short response time and some might produce a large quantity of data which could be a heavy load for the networks. In this case, cloud computing is not efficient enough to support all these things.

The main goal of this abstract was to define difference between CDN and Edge Compute, present and describe Edge Compute global architecture. An algorithm that allows migrating excess load from overloaded nodes to nodes with enough capacity was proposed. The goal of the problem is to distribute the hosting of services across the Edge Compute nodes in such a way that all loads can be successfully served. As a result, we will not have overloaded nodes in specific locations.

II. The Main Principles Of Information Distribution

The CDN technology consists of a huge number of Edge servers connected with adequate networking and placed as close as possible to the customers. A good working CDN should be based on two main principles:

1. Moving sources of information close to end users’ nodes.

2. CDN Cache Servers sharing.

A Content Delivery Network (CDN) is a globally distributed network of geographically dispersed servers. Each of them is located in various locations closer to customers than the origin server. The content is replicated and stored throughout the CDN. Thus CDNs improve network performance and improve accessibility by content replication [6-8].

There are different types of information: static and dynamic. Transmission and caching principles can be different, depending on the type of information.

In general, request routing in CDN consists of the following steps [9]:

(1). The end-user request will be sent to the CDN Provider Name Server.

(2). CDN Name Server based on user geolocation replies to the customer with the list of SLB nodes IPs.

(3a). The next request from the end-user, to get static content, will be sent to the closest SLB.

(3b) The Load Balancer will redirect this request to the Edge server with the lowest load.

(4). Edge server will try to get content from a local cache. If the content does not exist in the local cache, the request will be sent to the Origin server.

2020 IEEE 15th International Conference on Advanced Trends in Radioelectronics, Telecommunications and Computer Engineering (TCSET) 978-1-7281-5566-1/20/$31.00 ©2020 IEEE 10.1109/TCSET49122.2020.235584

(5). The Origin Content server returns requested static content to the Edge server.

(6). The Edge server saves content in the local cache and at the same time sends it to the user.

(7). All the next requests to this SLB for the same content will be processed from Edge server local cache.

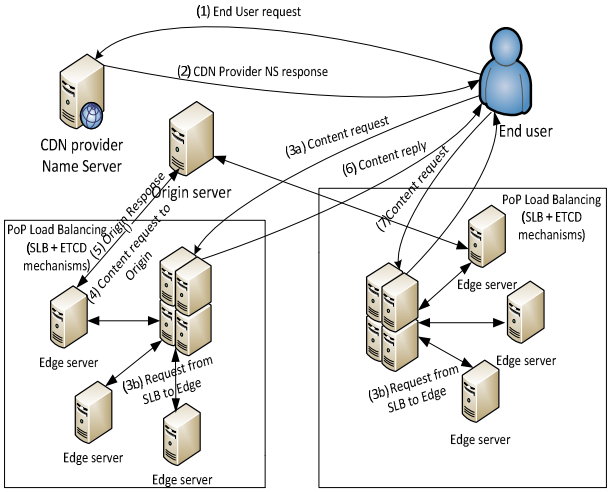

Fig. 1.Typical requests routing in CDN workflow

Edge Cloud Computing is a paradigm for improving the performance of cloud computing and distribution systems. It's based on CDN and performing data processing at the edge of the network, closer to the users and sources of data.

Edge Cloud Architecture and components are described in the next sections.

III. Edge Cloud platform Components

In Cloud paradigm, data is processing in the large data centers, which are located far from end-users. The main goal of Edge computing - reduce the latency between location of data processing and end-user. It allows to improve overall performance as well [10].

As the number of Internet-of-Things and mobile devices is constantly increasing, this becomes to be more and more important [11-12].

Edge Compute Cloud platform consists of the next components:

- ✓ **Network** - provide access for smart devices to the Compute platform and define Compute Engine close to IoT devices location.
- ✓ **Computing/Processing** - geographic distribution systems are built based on distributed servers provide data processing, computing, and data storage capabilities.
- ✓ **Storage** - a distributed storage used for backup and data protection, or distributing data objects to users.
- ✓ **Applications** - applications require integration with different platforms, devices, and technology (like Virtual Reality, Artificial Intelligence, IoT, etc.).

The Cloud platform components model is present in Fig. 2.

All of these components and the interaction between them represent the Edge Computing Global Platform.

IV. Edge Computing Platform Architecture

Edge computing is a distributed computing paradigm that is using to improve response times and save bandwidth and brings computation and data storage closer to the Edge location. It allows smart applications and devices to respond to data almost instantaneously, eliminating lag time. Edge Computing looks like a bridge between the digital and physical worlds.

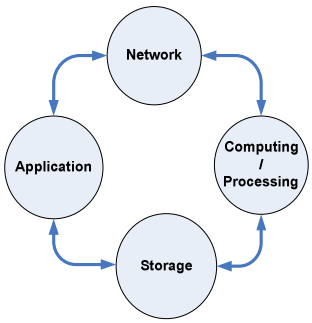

Fig.2. Cloud platform components model.

Edge computing allows to processed data near to the source and on the same time reducing Internet bandwidth usage. Edge computing platform plays a very important role in the next scenarios:

- ✓ real-time services and applications
- ✓ short-term data
- ✓ edge decision-making.

These eliminate costs and provide the ability to process data without ever putting it into a public cloud. This is very important for sensitive data.

In the Edge Computing Concept, there are a large number of geographically distributed Processing locations. The main goal is to decrease latency, processing time and improve QoS.

Edge computing architecture with a different module is presented in Figure 3.

The architecture includes bunch devices such as wireless sensors and smart devices that transmit data to the Edge Processing Center. These Centers can process data in real-time and respond to client requests. Edge Cloud has geographically distributed Services that manage all the Edge Computing and Storage infrastructure. Storage centers are using to store data and to create some analytics. The compute and storage resources migrated from the large data centers to compute

resources centers that are closer to the end-user devices at the edges of the network.

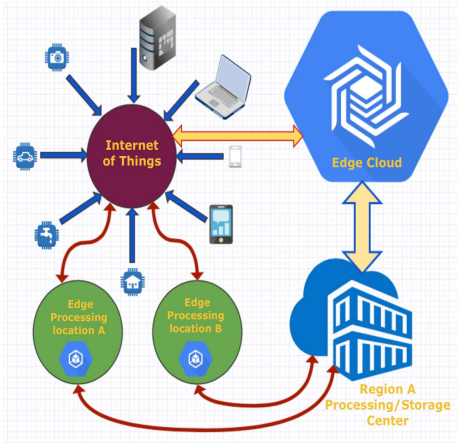

Fig.3. Edge Cloud platform Architecture.

But in this case, the entire load will be delegated to Edge Processing locations. And in this situation, some of the processing nodes can be overloaded. So, we need to define some load balancing mechanisms to prevent this situation.

V. Load Balancing in Edge Computing Concept

In this chapter we model the load balancing problem on a simple topology for edge computing [13-17] and propose an algorithm, to solve it.

Load balancing in Edge Cloud networks refers to the efficient distribution of the incoming workload across a group of processing compute nodes. The proposed algorithm is presented in Figure 4.

According to the presented Load balancing algorithm, there are a few steps that allow to migration of excess load from one node (overloaded) to another node in the same location (underloaded):

a. Overloaded node send message to all neighbors with the amount of excess load and hop_countr = 1;

b. Comparison of available units with the received amount of units from the overloaded node. If the node doesn't have enough capacity it will share information with its neighbors and increase the hop counter by 1;

c. If the node has enough capacity, this information will be aggregated, based on available capacities and hop counter, and sent to the parent node. All the neighbors should reply.

d. After that, the overloaded node has information about a list of nodes that have enough capacity and are ready to accept an extra load.

e. Overloaded node migrate migrates extra load to the node with smaller hop counter and node that has more than enough available capacity.

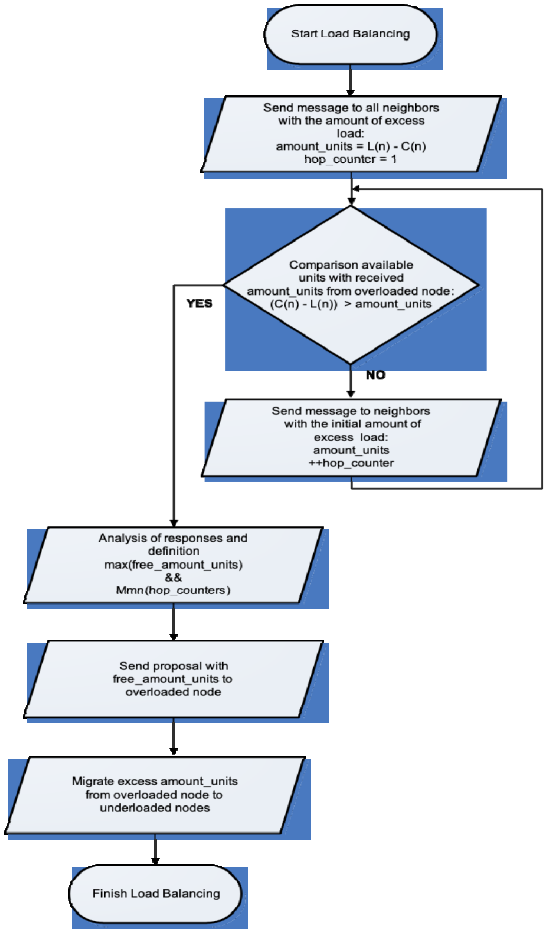

Fig.4. Load balancing algorithm.

Every Edge Compute node has some capacity C(n) and current load L(n). Each overloaded node (L(n) > C(n)) identifies a subset of other nodes that it should shed its excess load with. Fig. 4 illustrates a few steps that describe the proposed algorithm. In Fig. 5, black numbers in circles are the initial loads of the nodes; red numbers are the capacities and red numbers in circles - the load at the end of the algorithm flow. Workflow with a detailed state description is presented in Figure 5.

Every Edge Compute node has some capacity C(n) and current load L(n). Each overloaded node (L(n) > C(n)) identifies a subset of other nodes that it should shed its excess load with. Figure 4 illustrates a few steps that describe the proposed algorithm. In figure 5, black numbers in circles are the initial loads of the nodes; red numbers are the capacities and red numbers in circles - the load at the end of the algorithm flow.

As we can see in figure 5, node A is overloaded, for some nodes (B, C, G) load is an equal capacity and some nodes (D, E, F) are underloaded. The main goal of the proposed algorithm is to migrate extra load from overloaded nodes (node A) to underloaded nodes, which are closest to node A in this particular case. The description of the workflow for this algorithm is below.

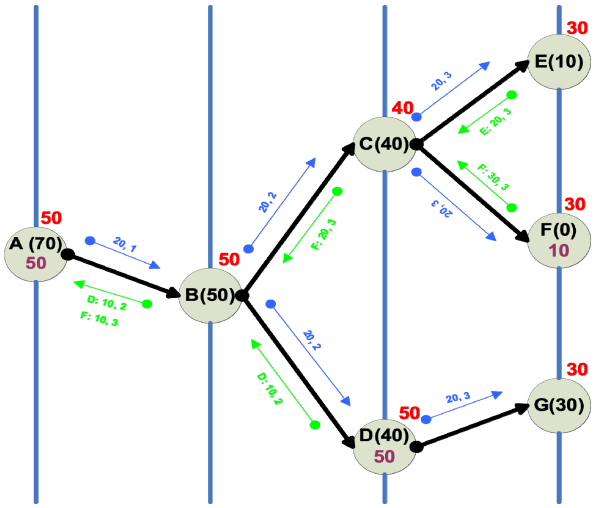

Fig.5. Load balancing workflow.

State 1. Overloaded nodes, in our example node A, will send a message to all neighbors with the amount of excess load. This message also includes a hop counter. When any node receives this message for the first time, it will share information with its neighbors and increase the hop counter by 1. If any node receives the same message type from neighbors, it will be ignored. This transmission of information continues until it reaches any node with enough capacity to accept the excess load from the overloaded node. At the end of this state, the message from node A will reach nodes that have enough capacity. In figure 5, these messages are marked as blue lines.

State 2: Each node that receives a message from the neighbor will replay with the node id (or name), available capacity and the number of hops to node A. When a node receives this reply from all neighbors, this information will be aggregated, based on available capacities, and sent to the parent. The nodes, with a smaller hop counter, will have a preference. In our example, these messages are marked as a green line. As can be seen in figure 5, node D has the capacity = 10 and hop counter = 2. This message will be sent to node B. Node C aggregates information from F and E and based on available capacity and hop counter also sends message to node B. It will aggregate received messages and forward to node A. At the end of this state, node A has information about list of nodes that have enough capacity and are ready to accept extra load.

State 3: Node A migrates extra load (20 units) to node D (10 units) because it has smaller hop counter and node F(10 units) because it has more than enough available capacity. All these actions, proposed in the load balancing algorithm can be optimal if they provide minimal processing time and a minimum overall overload probability.

VI. Conclusions

The main principles of CDN architecture and global concept content delivery network (CDN) system were considered in this paper. Request routing flow has been presented and described in this paper.

The main principles, the architecture of Edge computing, Edge model were considered. Edge computing is a distributed computing paradigm that is using to improve response times and save bandwidth and brings computation and data storage closer to the Edge location. It's based on CDN and performs data processing at the edge of the network, closer to users and sources of data. Serving as a bridge between physical and digital worlds, edge computing enables smart assets, smart gateways, smart systems, and smart services.

There has been proposed a load-balancing algorithm, which uses extra load migration from overloaded nodes (node A) to underloaded nodes, closest to the overloaded node.

References

[1] K. Ashton, "That Internet of Things thing," RFiD J., vol. 22, no. 7,pp. 97–114, 2009

[2] H. Sundmaeker, P. Guillemin, P. Friess, and S. Woelfflé, "Vision and challenges for realising the Internet of things," vol. 20, no. 10, 2010.

[3] J. Gubbi, R. Buyya, S. Marusic, and M. Palaniswami, "Internet of Things (IoT): A vision, architectural elements, and future directions," Future Gener. Comput. Syst., vol. 29, no. 7, pp. 1645–1660, 2013

[4] "Cisco global cloud index: Forecast and methodology, 2014–2019 white paper," 2014

[5] D. Evans, "The Internet of Things: How the next evolution of the Internet is changing everything," CISCO White Paper, vol. 1, pp. 1–11, 2011

[6] G. Pallis, and A. Vakali, "Insight and Perspectives for Content Delivery Networks," Communications of the ACM, Vol. 49, No. 1, ACM Press, NY, USA, pp. 101-106, January 2006

[7] A. Vakali, and G. Pallis, "Content Delivery Networks: Status and Trends," IEEE Internet Computing, IEEE Computer Society, pp. 68-74, November-December 2003.

[8] G. Peng, "CDN, "Content Distribution Network," Technical Report TR-125, Experimental Computer Systems Lab, Department of Computer Science, State University of New York, Stony Brook, NY, 2003.

[9] Pleskanka N., Kyryk M., Pleskanka M. The analysis of the optimal data distribution method at the content delivery network // The experience of designing and application of CAD systems (CADSM) : proceedings of the 15th International conference (Polyana (Svalyava), Ukraine, February 26 – March 2, 2019). – 2019. – C. 8779328-1–8779328-3.

[10] S. Yi, C. Li, and Q. Li, "A Survey of Fog Computing: Concepts, Applications and Issues," in Proc. 2015 Workshop on Mobile Big Data. ACM, 2015, pp.37-42

[11] Evans, D.: The internet of things: How the next evolution of the internet is changing everything. White Paper 2011; Cisco Internet Business Solutions Group (IBSG), Cisco Systems, Inc.:, 2011. Available online: http://www.cisco.com/web/about/ac79/docs/innov/IoT_IBSG_0411FINAL.pdf (accessed on 27 July 2018)

[12] A. A. Fuqaha, M. Guizani, M. Mohammadi, M. Aledhari, and M. Ayyash, "Internet of Things: A Survey on Enabling Technologies, Protocols, and Applications". IEEE Commun. Surveys Tuts., vol. 17, no. 4, 2015

[13] Pleskanka Nazar, Kyryk Maryan, Pitsyk Mariana. QOS mechanism in content delivery network. Proceedings of the XIIIth International Conference TCSET'2016. February 23 – 26, 2016 Lviv-Slavske, Ukraine.- P. 658-660.

[14] Rabinovich, M., Xiao, Z., Aggarwal, A.: Computing on the edge: A platform for replicating internet applications. In: Web Content Caching and Distribution, pp. 57–77. Springer (2004)

[15] Rust, P., Picard, G., Ramparany, F.: Self-organized and resilient distribution of decisions over dynamic multi-agent systems. In: Proceedings of the International Workshop on Optimization In Multi-Agent Systems (2018)

[16] Shi, W., Dustdar, S.: The promise of edge computing. Computer 49(5), 78–81 (2016)