

THE INTEGRATION OF EDGE COMPUTING INTO IOT APPLICATION USING ADVANTEDGE PLATFORM, CASE STUDY: CONNECTION PROTOCOL (UDP, TCP)

Dayoub A.¹
¹HSE university

Abstract

As the number of connected devices in the Internet of Things (IoT) increases, the amount of data is exploding. Therefore, this growth requires increased Internet speed during data transfer and more requirements for computing power and storage in central servers. To address these challenges and facilitate real-time data processing while optimizing network bandwidth utilization, edge computing has emerged as a promising paradigm. This study focuses on the integration of edge computing into IoT applications, facilitated by an edge simulation platform called AdvantEdge. We discuss the case of connectivity protocols (UDP, TCP) used by user devices (UEs) to connect to edge servers to solve problems such as latency and network bandwidth consumption in IoT applications.

Keywords: Multi-access Edge Computing (MEC), Internet of things IoT, IoT application, UDP, TCP.

Introduction

Multi-access edge computing (MEC) has emerged as a powerful strategy to solve the limited resources problems in IoT. In the MEC framework, compute-capable servers are strategically placed at the edge of the network, near IoT endpoints, thereby providing compute services to the requirements of IoT devices.

According to predictions presented in Cisco's Annual Internet Report (2018-2023) [1], the popularity of IoT devices accessing the Internet will increase, reaching approximately 30.9 billion by 2023. To In 2025, IoT devices are expected to generate approximately 73.1 zettabytes (ZB) of data, representing a significant increase from the 18.3 ZB recorded in 2019. This data includes countless number of sources, extended sensors, cameras and various connected devices. The growing volume of data, coupled with the evolving IoT device landscape, is driving growing demand for low-latency applications and robust bandwidth infrastructure. MEC technology presents itself as the optimal solution to meet the changing needs of IoT applications, a focus of this research.

Although conventional cloud computing offers advantages such as abundant computing resources, scalability, and cost-effectiveness, it presents several challenges, including single-point-of-failure risks and significant latency (potentially hundreds of milliseconds) between the user and the centralized cloud. In contrast, many IoT applications require decentralized systems that incorporate location awareness, reliability, and low latency. For time sensitive IoT services, latency must not

exceed 1 millisecond, and reliability must be no lower than 99.99% [2].

Main part

IoT edge computing refers to the processing of data generated by IoT devices that are close to the data source, thus avoiding the need to transmit all the data to a centralized server. This method speeds up data analysis and response time while minimizing data transfer costs.

The conventional IoT architecture usually consists of three layers: the perception layer, network layer, and cloud platform layer [3]. The perception layer is primarily responsible for data acquisition and system control, while the network layer facilitates the transmission of acquired data and control instructions. The platform layer performs a variety of functions, including device and connectivity management, application provisioning, and business analytics.

Integrating edge computing into IoT models is a powerful addition to traditional IoT architectures. Unlike traditional IoT frameworks, the IoT Edge architecture integrates an additional layer, namely the Edge layer, which sits in the middle between the perception layer and the platform layer, as illustrated in fig 1.

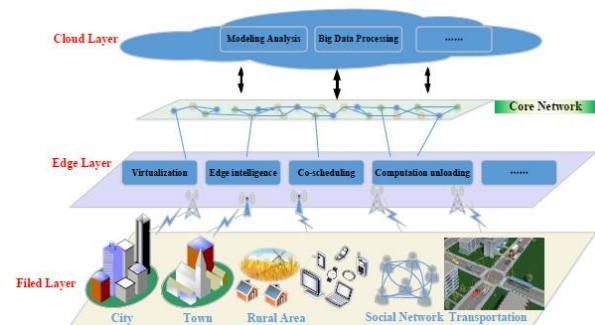


Fig 1: IoT Edge Architecture.

The edge layer serves as a functional sink for the cloud platform, capable of performing real-time data processing and analysis. It also facilitates real-time control and decision-making within the system, helping to reduce unnecessary data usage and conserve network bandwidth, minimizing system latency and minimizing energy consumption due to data transmission. Additionally, the edge IoT system operates as a closed system, thereby significantly improving system durability and reliability [4].

The integration of edge computing with IoT application

Our system has three main parts, which are the IoT sensors, the Edge computing platform and the cloud as shown in fig 2.

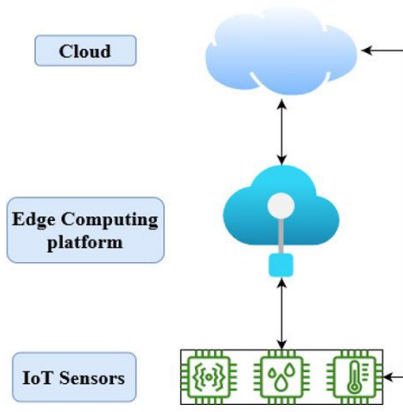


Fig 2: The schema of the IoT edge application.

This process involves deploying IoT applications on an edge computing platform, where IoT devices transmit data to the edge platform using protocols such as TCP or UDP, followed by further processing of the data.

To implement this scenario, we will use AdvantEDGE [5], a Mobile Edge Emulation Platform (MEEP) based on Docker and Kubernetes. The platform provides a simulation environment conducive to evaluating and testing advanced IT technologies, applications, and services.

To determine performance metrics, a multitude of parameters must be evaluated, including latency, bandwidth, and error rates. The comprehensive monitoring capabilities provided by AdvantEDGE make measuring these parameters easy. Then, a comparative analysis will be performed based on the scenario where IoT applications are executed in a cloud environment, thereby shedding light on the differences between the two methods.

The initial development phase includes designing the network model shown in Fig 3. Our network architecture consists of three main components: user equipment (UE), edge servers, and cloud. Additionally, backend network components such as zones, points of access (PoA), service providers, and Internet connections are also integrated.

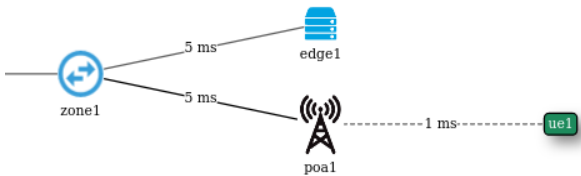


Fig 3: Network design.

In our network infrastructure, we have uplink/downlink throughput parameters set to 1000/1000 Mbps, while meeting platform-specific latency recommendations between each pair of components. regulations, shown in Fig 3.

After the network design phase, we proceed with integrating the application responsible for simulating IoT functions. To do this, we chose iPerf, a

well-known tool used to evaluate and optimize network performance. iPerf offers cross-platform compatibility and provides a comprehensive feature set to create standardized performance metrics across different network environments. This flexible tool includes both client and server functionality, making it easy to set up data streams to evaluate throughput between endpoints. Typical output provided by iPerf includes a time-stamped report describing data transfer volume and measured throughput [6].

In a technical context, our implementation involves deploying three iPerf instances: one runs as a client, while the other two run as servers, as shown in Fig 4.

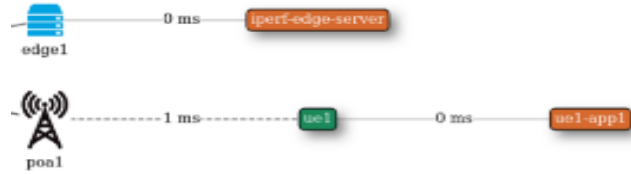


Figure 4: Network design with the iPerf applications.

UDP protocol

The application that runs on the UE works as a client and sends the data to the server and the edge server, it uses UDP protocol, and it sends 50 Mbps. The applications that run on edge and cloud work as a server and receive the data from the client, it used UDP protocol and listen on port 80.

The last step of the development is to deploy the scenario and see the results of the simulation, after the design of the network and the configuration of the network elements and the applications. So first we apply the scenario using UDP as protocol to send the data. Fig 5 indicates the latency between UE and the edge, also between the UE and the cloud.

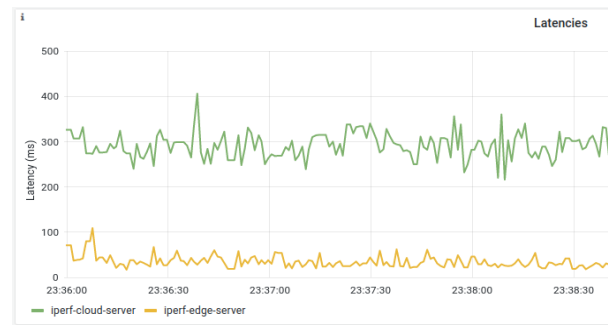


Fig 5: Latencies from UE to cloud and edge using UDP.

From the latencies chart we can see the big difference between the edge and the cloud in case of sending the data to the server. The latency between UE and cloud is in the range 250-350 ms, we can take 300 ms as a middle value. For the latency between the UE and edge it is around 30-70 ms so we can take 50 ms as a middle value.

We also measured the UL throughput for the UE, as shown in Fig 6 the UL is as indicated in our

command which is 50 Mbps with small variation, the DL is almost zero only the control signals. We did not set a DL because in most cases the data flow is going from the sensors or UEs to the cloud or edge.

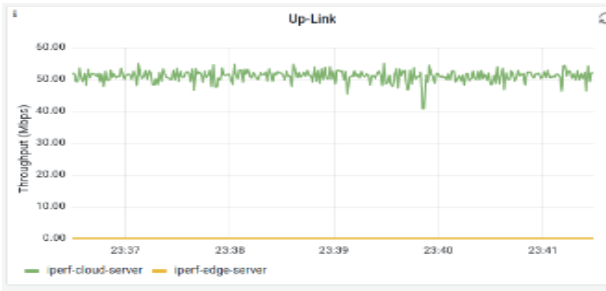


Fig 6 - UL from UE to cloud and edge using UDP.

TCP protocol

Now the deployment using the TCP protocol, in Fig 7 we can see the latencies from the UE to the cloud and to the edge using TCP. It shows a similar result to the case where we used UDP with a little variation.

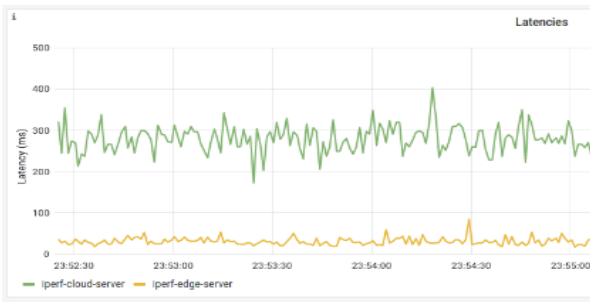


Fig 7 - Latencies from UE to cloud and edge using TCP.

The latencies chart is assuring that despite the using protocol the latency between UE and cloud way greater than the latency between the UE and the edge.

For the UL between the UE and the edge as you can see in Fig 8 there is a big variation when sending data using TCP and that is because of the way TCP work which is three handshakes first, even the DL has a spike of data that represent the packets send from the cloud to the UE for starting the connection.

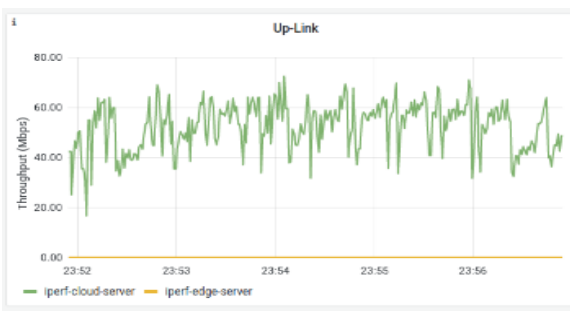


Figure 8 - UL from UE to cloud and edge using TCP.

We can clearly see that the latency between the UE and the cloud is very big for IoT applications (250 - 350) ms, but the latency from the UE to the edge server is (20-70) ms which is better for IoT applications.

Conclusion

In this paper we have presented a simulated model of an IoT application developed on an edge computing platform using two connection protocol (TCP, UDP), and we come across the fact that using MEC with IoT we will be able to achieve latency around (20-70 ms) rather than (250-350 ms) using only cloud and reduce the data load over the network. This model will open the doors to other studies on the idea of developing the IoT applications with the help of MEC, because the MEC is one of the main enabler technologies for IoT. What we can recommend after this study is to use edge server for critical application and use cloud for non-critical applications.

References

1. Annual C., Report I. White paper Cisco public. 2018.
2. Mahbub M. et al. Multi-Access Edge Computing-Aware Internet of Things: MEC-IoT // 2020 Emerging Technology in Computing, Communication and Electronics (ETCCE). IEEE, 2020. P. 1–6.
3. Jabraeil Jamali M.A. et al. IoT Architecture. 2020. P. 9–31.
4. Pan J., McElhannon J. Future Edge Cloud and Edge Computing for Internet of Things Applications // IEEE Internet Things J. 2018. Vol. 5, № 1. P. 439–449.
5. AdvantEdge Platform [Electronic resource]. URL: <https://interdigitalinc.github.io/AdvantEDGE/> (accessed: 08.05.2023).
6. IPerf tool [Electronic resource]. URL: <https://iperf.fr/> (accessed: 08.05.2023).