5G Edge Computing

Usha Chauhan
Department of Electronics and Communication Engineering,
Galgotias University, Yamuna Expressway
Greater Noida, Uttar Pradesh
Email ID: gu1413912423@Galgotiasuniversity.edu.in

ABSTRACT: The new 5 G telecommunication network is set to deliver high data speeds and ultra-low latency to meet future users ' complex demands. Multi-access Edge Computing (MEC) is a key part of the solution by supplying the edge of the radio access network with a cloud computing system. Edge computing is an evolving technology that allows evolution to 5 G by getting cloud technologies closer to end users to solve the conventional cloud's fundamental problems, such as high delay and lack of safety. The paper provides an overview of existing state-of - the-art 5 G edge computing technologies based on goals, technical frameworks, features, 5 G functions, output metrics and responsibilities. Author is addressing other important factors including the main criteria for its efficient 5G rollout and 5G edge computing applications. Then, he investigates, highlight and identify recent advances in 5G edge computing. It also exposes the fundamental precepts of various 5 G edge computing concepts.

KEYWORDS: 5G, Cloud Computing, Edge Computing.

INTRODUCTION

Edge computing is a computer model that requires edge servers in mini clouds to broaden cloud resources at the edge of the network to perform computer-intensive tasks and store massive amounts of data near to user equipment (UEs). Standard cloud computing, a centralized computing paradigm that provides consistent access to critical capable data centers, has been adopted to enable UEs to discharge data center computing and storage. Future applications may require high bandwidth, high durability, and/or low latency communication for UAVs, driverless cars, the Industrial Internet and the Internet of Things (IOT)[1]. To fulfill these needs, the revolutionary 5 G communication system not only promote an innovative air interface but will also introduce new technologies in the network – Software- Defined Networking (SDN), Network Function Virtualization (NFV), and, most notably, Multi-access Edge Computing (MEC)[2]. MEC is analogous to cloud computing but operates at the edge of the network, i.e. close to the base stations (in 5 G terms: gNB; next-generation NodeB), and is therefore far closer to the client than a traditional cloud server. The basic principle of spreading processing capacity at the edge of the network is also known as edge clouds, cloudlets and fog computing[3]. Figure 1 shows the Edge computer process.

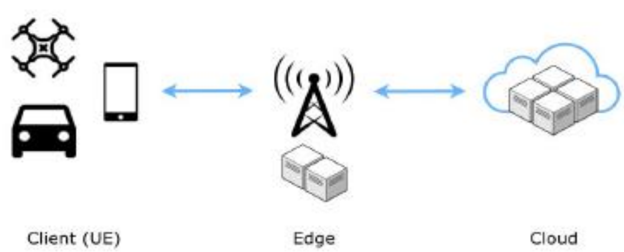

Fig. 1: Edge Computing Happens Near Base Station.

Edge Computing forms part of a fundamental shift from the traditional store-and-forward service-agnostic strategy to a smart network. Running applications at the edge of the network allows for services of low latency which comes at the cost with complexity[4].

SIGNIFICANCE OF EDGE COMPUTING

5 G is intended to support interactive devices with low latency and high demands on throughput. In order to reduce latency, edge computing adopts a decentralized model that takes cloud computing technologies closer to UEs. Edge computing can act as a single computing platform or interact with other elements, including the cloud. Edge computing is required because the conventional cloud computing model is not appropriate for interactive, computation-intensive applications with high QOS specifications including low bandwidth and high throughput[5]. That's because cloud might be a long way from UEs that also raises energy usage. In certain terms, cloud services are generally located in the core network, and mini cloud edge servers are located at the edge of the network[6]. Fig.2 shows the edge computing overview.

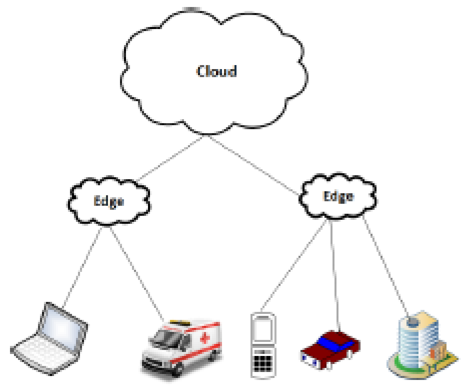

Fig. 2: Edge Computing.

TRANSPARENT EDGE SERVICES

Developers can develop edge services as quickly as traditional cloud services, and need not distinguish between cloud or edge service development. The sections to be performed in the edge can be further determined either manually or automatically by implementing machine learning models, or on - the-fly by observing current use of the network. The approach suggested tackles this goal with the use of SDN. The fundamental principle is that SDN-enabled connections in the 5 G network assign user requests for authorized services to the nearest computing node hosting the requested service (as determined by network latency). The method works this way: First, the customer requests a service, as a conventional cloud service would be required[7]. The network then intercepts the message, and redirects it to the nearest edge node available. At last, the edge node procedures the request and the client reacts. As interception occurs automatically through SDN interfaces in the network, it looks like the reply came from the cloud instance addressed to the clients. Note that perhaps edge application as a whole may use the cloud for the request processing[8]. Figure 3 shows the perceived cloud with respect to real cloud.

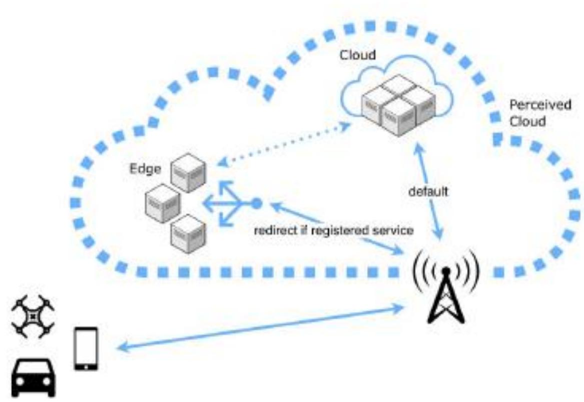

Fig. 3: Perceived Cloud vs. Real Cloud.

The system makes use of Open Flow's packet filtering and rewriting capability to achieve clarity towards the UE. Firstly, in its flow tables the gNB switch match an incoming packet against the active rules. When it notices a similar flow entry the packet will be forwarded accordingly. In event of a miss the packet is forwarded as a PacketIn message to the SDN controller. The SDN manager (keeping the domain-to-IP mappings track) matches the PacketIn message's destination IP / port combination to the list of approved edge services. The controller assigns the UE to the nearest edge host delivering the requested edge function, if the request is intended for one of those services. If there is no similar entry, the packet will need to be redirected to the cloud according to the default switching guidelines. In both cases the controller returns a Flow-Mod message which contains the switch's Redirect instructions[9]. Consequently, the message includes an idle timeout value that triggers an invalidation of the redirect entry when no contrasting packet is received in the given period. This enhances the forwarding efficiency by maintaining tables lean. Fig.4 shows the routing process within a registered service IP

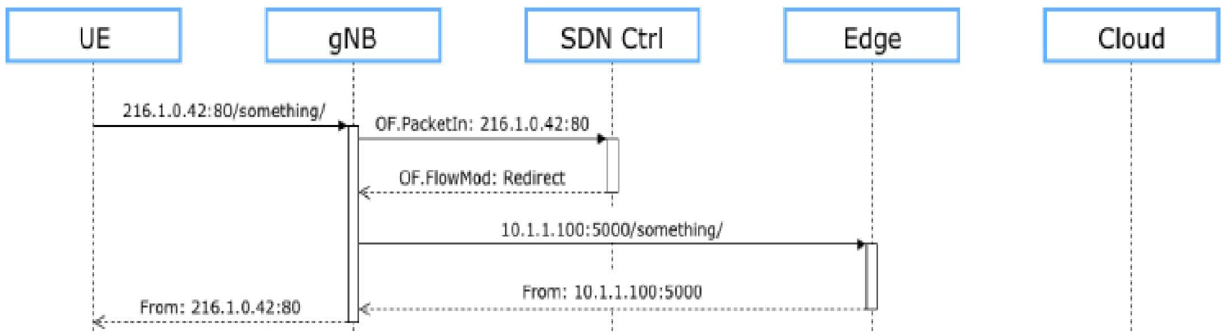

Fig. 4: Routing With a Registered Service IP

UE Migration:

When clients switch and connect to various base stations (gNBs), the network needs to determine repeatedly new best edge hosts for the given edge application. In such scenario, the local SDN switch for the newly arrived UE would have no flow entry now, and therefore transfer packet to the SDN handler. Because the SDN operator tracks each UE's connection point, it identifies the migration instantly, and adjusts the device's current position. In the case of an UE migration, the SDN controller can proceed in one of the following ways, depending on the edge service requirements. It might choose to link UE to the last instance of edge program, move the UE to a program running in a nearby edge host (as determined by network latency), or use any other host selection strategy based on completely different criteria. They assume stateless edge applications in this prototype and therefore forward the UE to nearest edge host providing the service needed. As a consequence the request would be routed in such a way as to achieve the minimum latency.Further formulas for host recruitment may be used using different criteria. Whereas the idea of transparent edge computing could be accomplished by intercepting and modifying the related DNS queries, the combination of a DNS-based approach with UE migration flops. The UE would have to issue new DNS requests with each such migration in order to find out the new optimal edge host. So although DNS provides the required isolation of service recognition and service location, it is not normally used to provide continuous updates of a service's current location.

LIMITATIONS OF PROTOTYPE

While the prototype does not limit the functional spectrum of edge services, there are limitations for the transportation layer protocol used. When an UE has to be allocated to a specific edge host, it cannot transfer open TCP connections. There needs to be a new link between the UE and the new edge host. As a response the model performs best for UDP links and short-lived TCP. They did not investigate how to manage secure communication (TLS), either. Compared to current methods used by CDNs, it may be a option to have the edge client keep a copy of the private key. Also, they have not yet incorporated a process with the edge network operator to sign the cloud services, which will be routed to edge applications.

PERFORMANCE AND SCALING OPTIMIZATIONS

Given that 5G is oriented towards supporting one million IOT devices / km2, outstanding performance and usability are critical. User has to pay attention to two possible bottlenecks in relation to these aspects:

- The number of flow inputs per SDN interrupter.
- UE number per SDN controller.

The easiest solution to tackle the amount of stream inputs per switch would be to evenly distribute the UEs among several switches per gNB. That would reduce the Open Flow rules that belong to a transition to a percentage of the overall set of rules instantly. A further solution would be to differentiate among switches that handle generic information and those that process edge-related traffic. As in the fig. 5, it would result in two step filtering. On the first point they were only able to check if combination of destination IP / port matched a recorded service.

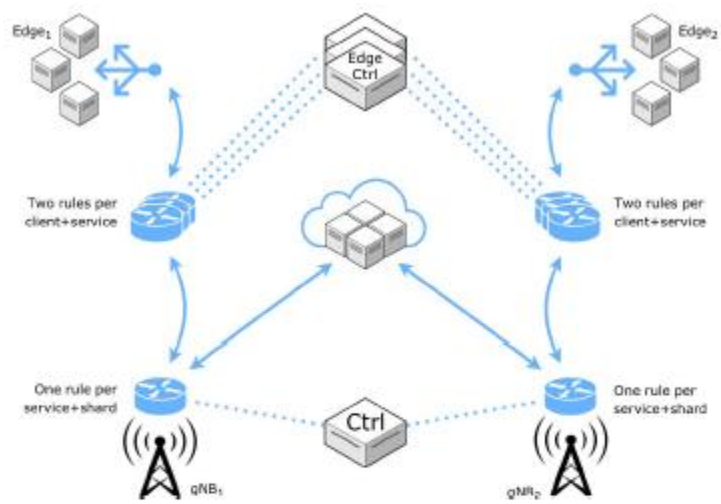

Fig. 5: Two-Stage Filter and Sharing of UEs to Increase Scalability

Nevertheless, if at first stage the packet fits a recognized cloud service, it is redirected to a second switch devoted to edge processing only. The latter includes all sessions of active edge applications connected to this gNB. In case of a match (i.e. a rule exists with the existing UE IP and network), the UE request will be redirected to its respective host and port edge. In case of a loss, the request is transmitted as a Packet In message to the SDN controller. The controller then assigns the UE to the nearest edge host which provides the edge application in question. As an added benefit of separating standard and edge switches, multiple SDN controllers could be assigned to these. That would enable the latter to be better scaled and thus help to mitigate the second slowdown. The core network handler would only have to manage the distinction between normal and edge traffic, whereas the edge handler would have to allocate UEs to a specific edge. Because the core operator would only manage the distinction between normal and edge-related traffic (rules that only change occasionally), they could continuously add the flow entries for registered services in the switches, instead of using timeouts[10]. Only when the DNS lookup shows a new IP address for one of the registered services may over dated rules be removed on requests. The main controller could also use shading to spread the edge connections equally among multiple edge switches for additional scalability. In addition, those edge switches could be delegated to different edge controllers, which would further spread the load. Nevertheless, if the aim is to speed up the filtering without increasing the amount of switches, some improved method might use two (instead of only one) flow tables within a switch. The Open Flow framework allows different tables of flow that could be used in many phases to match packets. As in the above approach, they could do matching in same two phases; this time, though, within a single switch. Rather than transferring to the next switch a packet allocated for an edge transaction, the rule would be to start processing in a follow-up row. This design would be flexible enough to allow one controller to start and disperse the load to multiple controllers when necessary.

CONCLUSION

This prototype needs fewer flow inputs in the switches, leading to better output and scalability compared with similar solutions. The current design and the Mini-net test platform include a solid foundation for further study and progress. They are currently working on a modular version which enables the various scaling optimizations to be compared quantitatively. The goal is to update the design and incorporate it on a physical test bed. The transparent edge service approach provides an interesting side benefit: If edge application vendor doesn't have an arrangement with an edge platform supplier, and therefore the product domain has not been recorded in the specified network, the UE demand would be transmitted to the intended cloud service. This optimistic approach to deployment has proved a precondition for effective adoption of emerging technologies. Edge application providers and network operators gain a lot of versatility and liberty by reducing the required changes to one edge network at a time.

REFERENCE

[1] Y. Mao, C. You, J. Zhang, K. Huang, and K. B. Letaief, "A Survey on Mobile Edge Computing: The Communication Perspective," *IEEE Communications Surveys and Tutorials*. 2017, doi: 10.1109/COMST.2017.2745201.

[2] T. Taleb, K. Samdanis, B. Mada, H. Flinck, S. Dutta, and D. Sabella, "On Multi-Access Edge Computing: A Survey of the Emerging 5G Network Edge Cloud Architecture and Orchestration," *IEEE Commun. Surv. Tutorials*, 2017, doi: 10.1109/COMST.2017.2705720.

[3] K. Roopakumar, "Multi-access edge computing," *J. Inst. Telecommun. Prof.*, 2018.

[4] H. Tanaka, M. Yoshida, K. Mori, and N. Takahashi, "Multi-access edge computing: A survey," *J. Inf. Process.*, 2018, doi: 10.2197/ipsjjip.26.87.

[5] E. Ahmed *et al.*, "Bringing Computation Closer toward the User Network: Is Edge Computing the Solution?," *IEEE Commun. Mag.*, 2017, doi: 10.1109/MCOM.2017.1700120.

[6] N. Hassan, K.-L. A. Yau, and C. Wu, "Edge Computing in 5G: A Review," *IEEE Access*, 2019, doi: 10.1109/access.2019.2938534.

[7] H. Guo, J. Ren, D. Zhang, Y. Zhang, and J. Hu, "A scalable and manageable IoT architecture based on transparent computing," *J. Parallel Distrib. Comput.*, 2018, doi: 10.1016/j.jpdc.2017.07.003.

[8] J. Hammer, P. Moll, and H. Hellwagner, "Transparent access to 5G edge computing services," in *Proceedings - 2019 IEEE 33rd International Parallel and Distributed Processing Symposium Workshops, IPDPSW 2019*, 2019, doi: 10.1109/IPDPSW.2019.00147.

[9] R. Munoz, R. Vilalta, R. Casellas, A. Mayoral, and R. Martinez, "Integrating optical transport network testbeds and cloud platforms to enable end-to-end 5G and IoT services," in *International Conference on Transparent Optical Networks*, 2017, doi: 10.1109/ICTON.2017.8025035.

[10] R. Alvizu, G. Maier, S. Troia, V. M. Nguyen, and A. Pattavina, "SDN-based network orchestration for new dynamic Enterprise Networking services," in *International Conference on Transparent Optical Networks*, 2017, doi: 10.1109/ICTON.2017.8025147.