Poster: A SDN-based Network Layer for Edge Computing

An Wang
an.wang@case.edu
Case Western Reserve University

Zili Zha
zzha@gmu.edu
George Mason University

Yang Guo
yang.guo@nist.gov
National Institute of Standards and Technology

Songqing Chen
sqchen@gmu.edu
George Mason University

ABSTRACT

Driven by big data analytical capabilities and ever-expanding Internet-of-Things (IoT) devices and applications, the new computing paradigm, Edge Computing significantly improves network performance by collecting and processing data locally or in a nearby edge data center. It offers a far less expensive route to scalability, allowing service providers to expand their computing capability as needed. Meanwhile, emerging network technologies, such as Software Defined Networking (SDN) and programmale data plane, can facilitate cost-efficient networking and direct network management. Such advancements stimulate recent efforts to adopt these technologies by Edge Computing to further improve efficiency and reduce latency. However, albeit a lot of efforts have been spent on the edge computing, little has been developed from the SDN perspective. In particular, a generic framework that integrates programmable network control is still missing. In this poster, we discuss the relevant challenges and propose an initial design of such a framework.

CCS CONCEPTS

• **Networks** → **Network architectures**; **Network services**.

ACM Reference Format:
An Wang, Zili Zha, Yang Guo, and Songqing Chen. 2019. Poster: A SDN-based Network Layer for Edge Computing. In *SEC '19: ACM/IEEE Symposium on Edge Computing, November 7–9, 2019, Arlington, VA, USA.* ACM, New York, NY, USA, 3 pages. https://doi.org/10.1145/3318216.3363333

1 INTRODUCTION

Fig 1 depicts a typical architecture of many Edge Computing applications today and potentially in the future as well. As shown in the figure, this architecture can be organized in three layers: application (service) layer, network layer and device layer, as shown in Fig 1. The application layer aims to organize and provide various services for system operators and/or interface with end users. Exemplary services include QoS management, security applications, data analytics and billing management, etc. The device layer is composed of heterogeneous Internet capable devices, such as mobile devices, smart home appliances, and vehicles. The network layer connects these two ends, and often consists of two parts: local area network

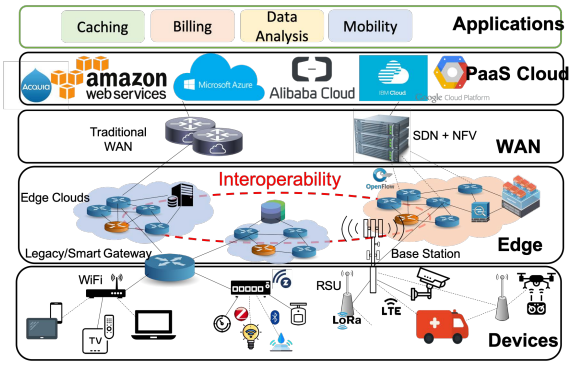

Figure 1: A general architecture of Edge Computing Environment

(LAN) and wide area network (WAN). LAN offers Internet access for devices while WAN acts as the interplay between the edge servers and remote cloud provider. Therefore, the network layer needs to be equipped both southbound interfaces and northbound interfaces.

Such an architecture faces several challenges from the network control perspective. (1) **Heterogeneity.** One of the ultimate goal of Edge Computing is to accommodate many devices and to provide various services. As a result, the heterogeneity increases along time. Such heterogeneity includes not only the devices and sensors used for different applications (e.g., auto-driving uses a different set of sensors and actuators than these used for smart and connected health), but also the way how these devices communicate with each other and with their service providers, e.g., using bluetooth, ZigBee, wifi, cellular direct interfaces and running HTTP/TCP or UDP protocols. While OpenFlow [4][1] is one of the standards for programmable data plane to communicate with their control plane, there is no such standard for the communications from the devices to the access points.

(2) **Interoperability.** The existing literature shows that a lot of edge computing innovations, platforms, and architectures are application driven. That is, they are designed or built for a specific type of applications, such as auto-driving with RSUs or smart-homes with smart gateways. In the future, applications may be composed on such existing services, which will require the interoperability and coordination among different edge platforms and edge computing

SEC '19, November 7–9, 2019, Arlington, VA, USA

ACM ISBN 978-1-4503-6733-2/19/11.
https://doi.org/10.1145/3318216.3363333

[1]Certain commercial equipment, instruments, or materials are identiïñĄed in this paper in order to specify the experimental procedure adequately. Such identiïñĄcation is not intended to imply recommendation or endorsement by the National Institute of Standards and Technology, nor is it intended to imply that the materials or equipment identiïñĄed are necessarily the best available for the purpose.

nodes. Moreover, different edge computing nodes may belong to different organizations. In addition, interoperability and coordination are essential to address the mobility of different devices. Ultimately, we envision that specialized and generic edge computing platforms would co-exist.

(3) **Mobility/connectivity.** In the application layers, some IoT devices naturally demand mobility support from the edge computing nodes. For example, with auto-driving, when the vehicle is traveling, the vehicle is interacting with the RSUs for traffic light information and other road condition information. Existing studies often focus on the smooth handover, with different prediction strategies, to make sure the shortest connection break. From the information or data perspective, such communications may be intermittent or sporadic depending on weather or other external factors.

(4) **Expressiveness.** The degree of complexity of the application logic could skyrocket with the increasing number of applications and services. To enforce the correct management policies, network operators need to carefully program and compose their applications to satisfy various service requirements. Thus, an expressive programming language that provides powerful abstractions and syntax that are compatible with mainstream programming languages is especially helpful. There are several domain-specific programming languages proposed to support SDN applications, such as Pyretic [5], Frenetic, and SNAP [2]. A similar platform is also necessary for Edge Computing frameworks.

(5) **Intelligent Network Management.** AI and machine learning have been successfully applied to various domains. They are behind the drive for Edge Computing. On the other hand, the innovations in the area of AI and big data analytics have the potential to simplify network management, such as monitoring and congestion control. It is crucial to leverage these powerful tools to manage a dynamic network system that is highly demanded by Edge computing. Certain technologies, such as approximation algorithms [6] and data collection techniques [3, 7], have been proposed to be adopted by the programmable data plane of SDN. Nonetheless, they are far from adequate for managing Edge Computing frameworks.

To address the above mentioned challenges, we propose an initial design of SDN-based network layer for Edge Computing.

2 DESIGN

We mainly focus our design on the network layer. Fig 2 illustrates such a design. We follow a similar event-driven paradigm with the existing SDN control platforms with several extensions. First of all, we propose to extend the southbound APIs with extra support for Wifi, ZigBee and Z-WAVE protocols. These are the dominant wireless protocols equipped by edge devices. The network drivers communicate with upper layers via an OpenFlow (or other counterpart) agent. The agent translates and enforces policies initiated by the control plane. It could be implemented either on the end devices or on a gateway router that is specifically designed for Edge Computing.

Secondly, we propose to introduce configuration handler and states handler in the control plane. The configuration handler is utilized to communicate with the data plane via different mechanisms, such as OpenFlow, gRPC, and Thrift. Such a handler allows

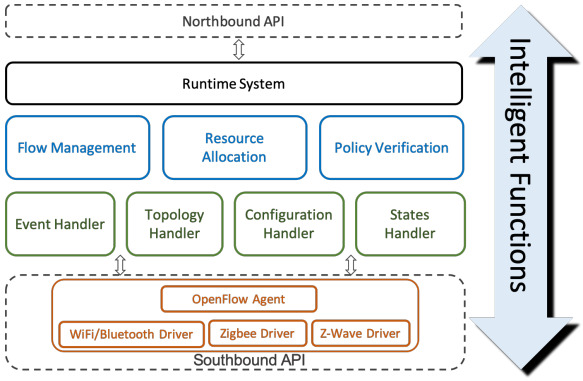

Figure 2: The design of SDN-based network layer for Edge Computing

the control plane to accommodate many different types of devices. The states handler is designed as a distributed storage system that maintains states of different flows from different clients. These two handlers are introduced to address the mobility and interoperability issues.

Additional management functions, such as flow management, resource allocation and policy verification, are designed in the upper layer of the control plane. Since these functions only have local visibility, they do not perform global optimization and management except for policy verification. Policy verification relies on the centralized cloud computing to perform the task by exchanging local information.

Thirdly, a runtime system is proposed and located at the uppermost of the control plane. It interprets syntax of high-level programming languages and compiles them into low level commands that could be understood by the lower layer functions. Existing SDN programming languages, such as Pyretic and Frenetic, only abstract the match+action forwarding model in the data plane. We plan to extend data plane oriented languages to service oriented languages by designing new service related queries and filters. The high-level programming language not only help reduce the burden of network programmers, but also facilitates formal verification of policies. Furthermore, it enables seamless integration with the existing Edge Computing services and applications.

Finally, an important component in the design are the intelligent functions. Depending on the specific models and algorithms, they could involve in multiple layers of the control plane. For example, sketch-based statistical algorithms could be implemented within the programmable data plane switches to detect abnormal flows. Another example is that the controller could leverage the information collected from the data plane to learn the optimized resource allocations for different flows in the data plane. It is extremely challenging if not impossible to design standard APIs for intelligent functions in different layers to make it work efficiently.

For implementation, we are going to implement our system over the existing mainstream SDN control platforms, such as Floodlight and Ryu. After the implementation is done, we plan to conduct experiments via simulators and public available testbeds, such as Living Edge Lab and EdgeNet [1].

3 CONCLUSION

Edge Computing demands cost-efficient networking, which could be provided by SDN. In this poster, we discuss the main challenges faced by Edge Computing to leverage advanced network technologies for more cost-efficient network management. To address these challenges, we have proposed a new network layer framework specifically for Edge Computing platform. It extends the existing framework with several modules. Such extensions are transparent to the clients and are mainly contained within the network layer. We also propose to leverage the advanced AI and machine learning techniques to manage the network resources.

4 ACKNOWLEDGEMENT

We appreciate the constructive comments from the reviewers. This work is partially supported by a NIST grant 70NANB18H272 and an NSF grant CNS-1524462. This work is also supported in part by the Institute for Smart, Secure and Connected Systems at Case Western Reserve University through a grant provided by the Cleveland Foundation.

REFERENCES

[1] 2018. https://edge-net.org. https://edge-net.org/. (2018).
[2] Mina Tahmasbi Arashloo, Yaron Koral, Michael Greenberg, Jennifer Rexford, and David Walker. 2016. SNAP: Stateful network-wide abstractions for packet processing. In *Proceedings of the 2016 ACM SIGCOMM Conference*. ACM.
[3] Zaoxing Liu, Ran Ben-Basat, Gil Einziger, Yaron Kassner, Vladimir Braverman, Roy Friedman, and Vyas Sekar. 2019. Nitrosketch: Robust and general sketch-based monitoring in software switches. In *Proceedings of the ACM Special Interest Group on Data Communication*. ACM.
[4] Nick McKeown, Tom Anderson, Hari Balakrishnan, Guru Parulkar, Larry Peterson, Jennifer Rexford, Scott Shenker, and Jonathan Turner. 2008. OpenFlow: enabling innovation in campus networks. *ACM SIGCOMM Computer Communication Review* (2008).
[5] Joshua Reich, Christopher Monsanto, Nate Foster, Jennifer Rexford, and David Walker. 2013. Modular sdn programming with pyretic. *Technical Reprot of USENIX* (2013).
[6] Naveen Kr Sharma, Antoine Kaufmann, Thomas Anderson, Arvind Krishnamurthy, Jacob Nelson, and Simon Peter. 2017. Evaluating the power of flexible packet processing for network resource allocation. In *14th {USENIX} Symposium on Networked Systems Design and Implementation ({NSDI} 17)*.
[7] Tong Yang, Jie Jiang, Peng Liu, Qun Huang, Junzhi Gong, Yang Zhou, Rui Miao, Xiaoming Li, and Steve Uhlig. 2018. Elastic sketch: Adaptive and fast network-wide measurements. In *Proceedings of the 2018 Conference of the ACM Special Interest Group on Data Communication*. ACM.