Comparing caching approaches with Software-defined Networking (SDN) for Internet of Things (IoT) applications

Florian Weidner
Philipps-University Marburg, Germany
Department of Mathematics and Computer Science, Distributed Systems Group
February 09, 2024

***Abstract*—The abstract goes here.**
***Index Terms*—SDN, IoT, caching**

I. Introduction

With the development of the internet and the increasing complexity of networks, the management and configuration of them become more complex and time consuming. Technologies like moblie networks, cloud computing, multimedia applications and virtualization have a high need of bandwidth, high accessibility and dynamic network configurations. These requirements are a challenge for traditional networks.

Traditional Networks are very hardware-centric. Routers and switches are used to manage the network traffic. The control plane is very tightly coupled with the forwarding by the data plane. Since both are happening on the local device, the configuration and management of the network is very time consuming. Software-defined Networking (SDN) addresses these issues. It uses a centralized approach for managing the network devices. This leads to easier configuration, more flexible forwarding, enhanced performance and reduced costs. [4] [7]

In this paper we will first summerize the concept of SDN and look at applications and challanges In Section III we will focus on the usage of SDN in IoT applications. In Section IV we then deep dive even more into caching approaches with SDN for IoT applications. Multible caching strategies will be compared. Finally, we will conclude our findings in Section V.

II. SDN

> "Software-defined Networking (SDN) is an emerging network architecture where network control is decoupled from forwarding and is directly programmable" [9]

This definition is by the Open Networking Foundation (ONF) from 2012. Software-defined Networking (SDN) decouples the control form the data plane. It uses a centralized controller, which has a global view to the network to manage the control plane. The controller can manage and adjust the network and the forwarding configuration of the network devices. There exist different implementations of SDN controllers. In II you can see the difference in the architectures

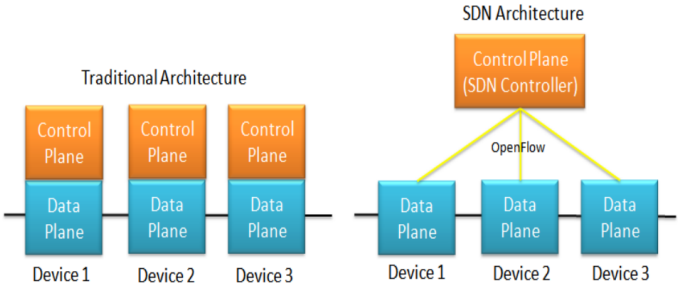

Fig. 1. Traditional Architecture and SDN Architecture [4]

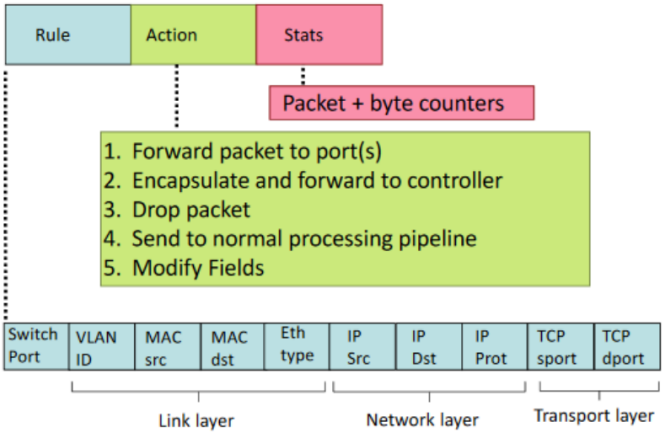

Fig. 2. Flow Table Entry Representation [8]

of traditional networks and SDN networks. In the traditional architecture, each device has its own control plane to manage the device. In SDN the control plane is centralized using the SDN controller.

The main responsitblity of the data plane is forwarding the network traffic. For that, it uses flow tables to determin the forwarding desitnation, which are more complex forwarding tables of traditional routers or switches. More complex decisions based on the information of incoming packtes are possible. In II you can se the representation of a flow table entry. An Entry contians three columns. The first one contains the rules to match the incoming packages. The rules can

be applied to any part of the datagram. The second one are the actions, that should be executed if the rule matches. The third column is used to store performance metrics on their corresponding rule and action field. [8] The dataplane can also be used, to enable various functions like network inspection, anomaly detection or traffic engineering. [7] The third plane is the application plane. On that plane software is used to manage the network over the SDN controller. There complex funtions can be performed to configure or automate the network trafic, based on the customer needs. Application Programming Interfaces (APIs) are used to comunicate with the hardware in the network.

The three planes use dedicated interfaces to comunicate. The southbound interface is used to comunicate between the control plane and the data plane. The northbound interface is used to comunicate between the control plane and the application plane. The OpenFlow Protocol, maintained by the ONF is a commonly used open-source protocol defining an interface for the southbound communication between the controller and the network devices. It defines guidelines and uses Transmission Control Protocol (TCP) to update the flow table entires from the control plane. If the controller is distributed the east-westbound APIs can be used to comunicate between the controllers. [8]

A. Advantages and Challenges of SDN

Software-defined Networking (SDN) has compared to traditional networking several advantages. Here the some of them are summerized.SDN provides a better and easier management of the network. All network devices can be controlled from a single point. Also newlly added devices can be easily integrated into the network. [4] Also the performance of the network will be imporved. It is possible to orchistrate the network traffic centrally. This leads to a better dynamic utilization of the network resources. This also leads to reduced costs. The management of the network is more efficient by using a central software, since there is less need to acces the individual network devices directly. [4] The forwarding network devices can be simplified. They only need to be able to forward the network traffic and have basic functions to be able to execute the instructions of the contorller, which takes over the management logic. [1] With SDN the network gets programmable with applicaitons that are installed ot the control plane. The control plane can be dircetly programmed, since it is seperated from the data plane. That also makes automation possible. [1]

But SDN also faces some challenges. Research into SDN mainly focused on the control plane. The programability of the dataplane is not as advanced as the control plane. With OpenFlow, there is no solution provided for data plane customization. [7] For forwarding devices have a tradoff in flexibilty and performance. General purpose processors provide the highest flexibilty wheras specific standard products are specialized for high performance but lower flexibilty. [4]New SDN switches are using hardware combinations to achive a better balance between flexibilty and performance. [7] SDN networks are dependent on OpenFlow compatible switches, which limits the scalability. Also the controller needs to be distributed to achive further scalabiltiy, over the limits of a single controller. Splitting the controller leads to typical distributed system problems like latency, fault tolerance, consistency and load balancing. On the other and it also leads to more resilicance, performance and availability. [7] Since SDN is widely being adopted and used, security is getting very important. Controllers are a central target for security threads. With unauthorized access to the controller, the whole network can be compromised. Authentication between controllers and their network devices with Transport Layer Security (TLS) lighten these threads. To achive a secure network protection, an effective security model is mandatory. [4]

B. Applications of SDN

SDN networks are used for data centers, enterprise networks, optical networks and even homes and small businesses. SDN enables customization and deployment of new services or policies, because of the independence of the control and the data plane. Therefore SDN can be used in various network environments. Data centers operate large-scale networks with high traffic, traffic management and many policies. Here SDN can be used to manage the network traffic and to provide a better utilization of the network resources. Generally, the same works for enterprise networks. Also for optical networks, the ONF provides specialized protocols to integrate multiple network technologies. And even for small networks it turns out that using SDN is useful. Having a single point of control makes it easier to manage the network. [4]

III. SDN IN IOT

Internet of Things (IoT) connects devices with limited resources to create various services. IoT applications are created to mostly collect data and execute tasks for multiple domains, like industrial process systems, traffic monitoring and a large variety of end-user applications. Often they result in large-scale networks with many heterogeneous devices and protocols. [6] Jazaeri et al. identified that IoT faces the following problems:

- Difficulties in control and management, due to the geographically distributed heterogeneous devices in various domains.
- Difficult to program and configure, due to different devices with different capabilities, like memory constraints, bandwidth and energy usage.
- Long service provisioning, due to the need for a full development cycle, including installing, configuring and testing the devices.
- Resources are not fully used. Devices haven't been completely considered as network resources. [12]

Missing flexibility, intelligence and application-specific controls, lead to these problems. SDN brings a global view on the network and provides capabilities to use network resources efficiently. It reduces the management and brings flexibility to mitigate the problems of IoT.

IoT devices need to be managed a lot, due to the need for configurations, reconfigurations, resource allocations and communication between the devices. [12] The concept of a central controller of a SDN network fits the need for central management for all devices in an IoT network. The controller can be used to activate and deactivate sensors, based on the current needs. Also, the routing of sensor data can be optimized. [6] There exist multiple frameworks for managing IoT devices with SDN. [12] The integration is not trivial, since SDN mainly focuses on controlling traffic. It lacks the ability to control sensor hardware and IoT applications. [6]

Another application of SDN for IoT is for cellular networks. There are multiple proposals for SDN-based cellular networks. With policies and a central controller, abstractions in geographical areas, load balancing, packet inspection and packet processing can be achieved. [12]

The most common device in IoT are sensors. For sensor networks, there are also proposals for SDN-based solutions. One example is the Software-Defined Wireless Sensor Network Framework (WSNSDN). It consists of a Base Station (BS) and several sensor nodes in a classical architecture. The BS is controlled by the SDN controller, managing the routing instead of the sensor nodes. The sensor nodes also contain flow tables like switches in the SDN architecture. [12] There also exist architectures using reconfiguration based on customer needs. That enables sharing a single infrastructure for multiple applications. Sensor OpenFLow (SOF) also propose reprogramming and retasking the sensor nodes with the control plane. [6]

SDN based IoT networks are also used for improved security, enabling authentication and authorization of the devices on the controller. With a global view over the network, approvals of connections can be handled securely by the controller. It also helps run distributed firewalls or detect unauthorized malicious devices. [12] It limits the impact caused by security threads for the most devices in the network. The controller on the other hand promotes to a central target for security threads. [6]

There are also challenges for SDN based IoT applications. [11] [12]

IV. Caching in IoT with SDN

[5] [3] [2] [10]

V. Conclusion

SDN is great. It brings a lot of advantages...

References

[1] Mudassar Hussain et al. "Software-defined networking: Categories, analysis, and future directions". en. In: *Sensors (Basel)* 22.15 (2022), p. 5551.

[2] Seyedeh Shabnam Jazaeri et al. "An efficient edge caching approach for SDN-based IoT environments utilizing the moth flame clustering algorithm". In: *Cluster Computing* 27.2 (May 2023), 1503–1525. ISSN: 1386-7857. DOI: 10.1007/s10586-023-04023-9. URL: https://doi.org/10.1007/s10586-023-04023-9.

[3] Seyedeh Shabnam Jazaeri et al. "Composition of caching and classification in edge computing based on quality optimization for SDN-based IoT healthcare solutions". en. In: *J. Supercomput.* 79.15 (2023), pp. 17619–17669.

[4] Abigail Jefia, S Popoola, and Atayero. "Software-defined networking : Current trends , challenges , and future directions". en. In: *Popoola, S. (2018, September). Software-Defined Networking: Current Trends, Challenges3rd North American International Conference on Industrial Engineering and Operations Management* (2018).

[5] Naga Katta et al. "CacheFlow: Dependency-aware rule-caching for software-defined networks". In: *Proceedings of the Symposium on SDN Research*. New York, NY, USA: ACM, 2016.

[6] Yuhong Li et al. "Enhancing the Internet of Things with knowledge-driven Software-defined Networking technology: Future perspectives". en. In: *Sensors (Basel)* 20.12 (2020), p. 3459.

[7] Rahim Masoudi and Ali Ghaffari. "Software defined networks: A survey". In: *Journal of Network and Computer Applications* 67 (2016), pp. 1–25. ISSN: 1084-8045. DOI: https://doi.org/10.1016/j.jnca.2016.03.016. URL: https://www.sciencedirect.com/science/article/pii/S1084804516300297.

[8] Alexander Nunez et al. *A Brief Overview of Software-Defined Networking*. 2023. arXiv: 2302 . 00165 [cs.NI]. URL: https://arxiv.org/abs/2302.00165.

[9] Open Networking Foundation. *Software-Defined Networking: The New Norm for Networks*. White Paper. Open Networking Foundation, Apr. 2012. URL: https://opennetworking.org/wp-content/uploads/2011/09/wp-sdn-newnorm.pdf.

[10] Giuseppe Ruggeri et al. "Caching popular transient IoT contents in an SDN-based edge infrastructure". In: *IEEE Trans. Netw. Serv. Manag.* 18.3 (2021), pp. 3432–3447.

[11] Ali Haider Shamsan and Arman Rasool Faridi. "SDN-Assisted IoT Architecture: A Review". In: *2018 4th International Conference on Computing Communication and Automation (ICCCA)*. 2018, pp. 1–7. DOI: 10.1109/CCAA.2018.8777339.

[12] Sahrish Khan Tayyaba et al. "Software Defined Network (SDN) Based Internet of Things (IoT): A Road Ahead". In: *Proceedings of the International Conference on Future Networks and Distributed Systems*. ICFNDS '17. Cambridge, United Kingdom: Association for Computing Machinery, 2017. ISBN: 9781450348447. DOI: 10.1145/3102304.3102319. URL: https://doi.org/10.1145/3102304.3102319.