MARS: Machine learning based Adaptable and Robust Network Management for Software-defined Networks

Farah Kandah  
Computer Science and Engineering  
University of Tennessee at Chattanooga  
Chattanooga, TN  
farah-kandah@utc.edu

Ilker Ozcelik  
Info. Systems Engineering & SimCenter**  
Recep Tayyip Erdogan University  
Rize, Turkey  
ozcelikilker@ieee.org

Brennan Huber  
Computer Science and Engineering  
University of Tennessee at Chattanooga  
Chattanooga, TN  
khsl616@mocs.utc.edu

Abstract—Traditional networks were initially designed to scale fast, but in turn are harder to monitor and manage. The rise in the Internet of Things (IoT) has caused an increase in the number of mobile nodes and thus the topology changes constantly. This compels researchers to explore more efficient methods to monitor and manage the network. Software Defined Networking (SDN) have become the primary focus of the research community due to the flexibility it enables by the separation of the data and the control plane. However, the centralized nature of SDN causes a scalability and a single point of failure problems. To combat this problem, we propose an adaptable and robust network management approach using machine learning while considering the control plane architecture for software-defined networks. Our system aims to enhance the network resource utilization and increase the SDN’s scalability by using multiple controllers and assigning the switches among them autonomously, based on network traffic patterns.

Keywords—Software Defined Network, Machine Learning, Internet of Things, Control Plane, traffic pattern

I. INTRODUCTION

The emergence of Software-Defined Networking (SDN) has brought along a wave of new developments in the field of networking with hopes of dealing with network resources more efficiently [1]. SDN enables flexibility by separating the control and data planes in a network environment and virtualizing the network [2] [3]. Through the programmability features of SDN, handling data transfer between hosts become more efficient. A number of leading companies such as Google and Amazon have adopted SDN to efficiently build private, public and hybrid clouds to advance the application development agility aiming to adapt to business needs. SDN enables the ability to monitor and control the networking resources in a deeper level to facilitate the IT as a service through the infrastructure control.

The rapid growth of connected devices and the Internet of Things increased the level of urgency of designing a scalable and adaptable networking infrastructure that can cope with high demands. Software Defined Networks (SDN) presents a promising solution and as a replacement for the current network design [4]. With its separation of data and control plane it allows for a better network monitoring and management.

Although, SDN provides a number of benefits such as breaking the vertical integration, separating the network’s control logic from the underlying routers and switches, promoting centralization of the network control and introducing the ability to program the network, it suffers from a number of challenges related to its implementation and its ability to function efficiently including the single-point-of-failure and the scalability [5].

The issue of single-point-of-failure occurs due to the network relying on a single, centralized authority. This singular controller has all the power when it comes to handling traffic in a network and if it were to experience any problem, then the entire network suffers as a result. Moreover, if, for any reason, the controller needs to be reset or shut down for a period of time, the entire network will be forced to experience the downtime as well.

On the other hand, with increase in number of nodes on the networks and advancement of mobile communication technologies; managing and securing traditional networks become even more difficult. Clean slate network design efforts has been increased in the past decade [6]–[8], with SDNs playing a significant role in most of these efforts. In addition to management benefits, gaining a global visibility of a network and being able to react incidents immediately on the network would also improve network security significantly. However, because of the centralized nature of SDN, these networks require a fail-safe mechanism and an approach to sustain scalability.

To address the aforementioned issues, we in this work, propose an adaptable, scalable and robust software defined network management approach aiming to handle the network traffic more efficiently, taking into consideration the network demand and its resource utilization at any given time.

Managing a network with a highly dynamic nature requires a dynamic control plane to efficiently utilize and effectively secure network resources. With the network demand increase, there is both a critical and urgent need to design, prototype,
validate, and demonstrate an efficient management approach aiming to reduce the network latency, increase the network scalability and efficiently utilize the network resources. Furthermore, we believe that through the analysis of network traffic and the extraction of certain user behavior as well as machine to machine interactions, it is possible to optimize the network scalability and speed.

The remainder of the paper is organized as follows: We present the related work in Section II, followed by our motivations and list the contributions in Section III. We present our Machine Learning based Adaptable and Robust Network Management approach is Section IV, followed by a discussion on the use of our system as well as future work in Section V. We present the system design and implementation in Section VI and finally, we conclude the paper in Section VII with a closing statement about where our system stands in the current field.

II. RELATED WORK

Related work presented in this section covers two complementary components of this work.

A. Software-defined networks

A number of distributed control plane designs for a scalable SDN network proposed in the literature [9]–[15]. A distributed cluster based control plane architecture proposed by Yazici et. al. in [12] is based on synchronizing controllers using JGroups membership notification and messaging infrastructure. This architecture aims to pose as a single controller to applications and switches and enables dynamic addition and removal of controllers. Dixit et. al. in [10] proposed an Elastic Distributed controller architecture that can dynamically grow and shrink the control plane based on controller system load with the ability to migrate switches between controllers to provide load balancing on the control plane network.

Another design (DIstributed SDN Control plane (DISCO)) was presented in [9] aiming to manage distributed and heterogeneous networks. The network split into multiple control domains controlled by one controller each. Controllers exchange status information with each other using Advanced Message Queuing Protocol (AMQP). Bari et. al. in [13] proposed a control framework to minimize flow setup time by adjusting number of controllers in the control plane and delegating the switches. They also proposed two heuristic approaches to (re)distribute switches among controllers. Their first approach uses greedy knapsack algorithm to utilize controller capacity efficiently. While the second approach focuses on determining and eliminating overloading controllers, than optimizing the switch distribution to improve performance.

Karakus and Durresi proposed a hierarchy-based network architecture to reduce control plane communication and increase scalability. In the proposed system, each domain controlled by its controller and controllers rely on a broker (higher level controller) for inter domain routing [14]. Yeganeh and Ganjali proposed a framework for distributed control plane that uses two layers of controllers to reduce control plane overhead. While bottom level controllers handle local events and applications, upper level controller takes care of rare events, like elephant flows [15].

In this work, we also considered the elasticity on the control plane and the redistribution of switches among the controllers. However, best of our knowledge, our work is the first approach using data mining and machine learning techniques to design efficient and dynamic SDN control plane.

B. Machine Learning

SDN increases the network efficiency through its programmability features which allows configurations and optimizations to be dynamically applied throughout the network. This idea has been expanded with machine learning implementations to optimize next generation networks.

Studies have shown that implementing dynamic routing using neural networks and Dijkstra’s algorithm can be used to improve the quality of service [16]. In essence each machine learning approach has a common strategy as many other studies which have been conducted to identify network optimization [17], [18].

Studies have shown that implementing machine learning on top of the network model allows for classification and feature extraction that will provide an ideal configuration of the network under high load environments [19], [20]. This has proven to be quite a difficult challenge that is not unique to SDN [19]–[22]. One of the common complications with automated optimizations using machine learning is the high variety of data which goes through a network. While it has been proven that optimizations can be made to increase efficiency, this high variety of data increases the complexity of the machine learning model in order to most efficiently predict and optimize the network [22]. Given a multi-controller network, it can be expected to require the same topology changes that have been proven to increase efficiency, but also will require further analysis in order to optimize the controller architecture.
In the study, we will not only apply machine learning to configure the network to be optimal within a single controller network, but we will also apply machine learning to optimize the topology of the controllers that will exist in our multi-controller network. Furthermore, machine learning will be applied to reconfigure and recover the network in case of a controller(s) failure. This way, we will not only maintain the efficient use of resources on the network but also we will establish a fail-safe mechanism to ensure that the software defined network is always operating.

III. OUR MOTIVATIONS AND CONTRIBUTIONS

Software-Defined Networking (SDN) has profoundly affected the field of networking with hopes of dealing with network resources more efficiently and also provides a foundation for programmability. SDN separates the control and data plane to observe and control the entire network state from a central vantage point, hosting features such as routing protocols, access control, energy management, and new prototype features. This shift in network management from a decentralized to more centralized approach caused a significant scalability and a single point of failure issues.

Scalability is the capability of a system to perform under increasing workload. Similarly, in networking scalability can be defined as maintaining the efficiency of a network as it grows. In the traditional networks, performance depends on and most of the time limited to link and switching equipment capacities. On a software defined network, on the other hand, network has an additional dependency; controller, since the decision (control plane) and the switching (data plane) processes run in separate locations and there is a unified controlling entity.

Although, separating control and data planes give more flexibility to manage a network, it also creates an additional communication overhead. It limits network performance with controller or control system capacity such as network bandwidth and computational power. Additionally, network congestion and delay between controller and switches degrades the overall network performance. These problems can be addressed by implementing proactive flow rules or using multiple controllers to reduce average controller load. However, while the former approach reduces the network management flexibility, the latter one would create a synchronization problem on the control plane. Therefore, on an SDN network, scalability problem can be addressed by deciding trade-offs between many dynamic parameters.

Karakus and Durresi classified SDN control plane approaches based on their topologies as centralized and distributed approaches [23]. In centralized scenario, the controller needs to make all flow decisions and becomes a performance bottleneck. Also, it creates a single point of failure on the network. In comparison, a distributed approach, where more than one controller are utilized to monitor/manage the network.

Distributed approaches are further classified as Distributed Flat, Hierarchical and Hybrid. While distributed flat and hierarchical approaches use multiple controllers to reduce the average controller workload, hybrid distributed approach offloads some of the controller tasks back to switches. The fundamental difference between the distributed flat and the hierarchical approaches is the level of network view of controllers. While each controller has a global view of the network in distributed flat approach, only master controller has this information in hierarchical approach.

In this study we focus on distributed control plane approaches, since the centralized approach has scalability and single point of failure problems. Additionally, we eliminated the hybrid distributed approach due to the fact that it reduces the flexibility of the control plane by handing over some of the control functions back to network switching equipment.

Due to the incredibly high demand that the world has placed on today’s networks, our motivations for this work are derived from the two primary problems. One being the single point of failure and scalability in Software Defined Networks, the second being efficient utilization of network resources, therefore, in this project and to fulfill these needs, we propose an network management approach through combining SDN control plane, data mining and machine learning technologies with a long term goal of creating a scalable SDN control plane that is robust and adaptable taking into consideration the efficient use of network resources.

We summarize our contributions as follows:

- Utilize data mining and machine learning techniques to better understand the network dynamics.
- Design a network control plane that can autonomously adopt to network changes.
- Design an SDN control plane architecture that can utilize network resources efficiently and can scale.
- Design an elastic control plane to prevent single point of failure.

IV. ADAPTABLE AND ROBUST NETWORK MANAGEMENT

In this section we present two potential control plane typologies for the proposed system: MARS, the Machine learning based Adaptable and Robust Network Management for Software-defined Networks. We also present the workflow of our proposed approach.

A. Control Plane Topology

The topology of the SDN’s control plane will have an impact on the overall system performance, therefore, it is critical to analyze multiple control plane architectures. The distributed flat and the hierarchical designs were selected in this study, which are presented in Figure 2. To determine which design offers the highest optimizations each will be critically analyzed based upon latency, resource utilization and controller overload including flow requests, and flow setup latency.

(a) Distributed (Flat) controller

(b) Hierarchical controller

Fig. 2: SDN control plane architecture
1) Distributed (Flat) Controller Design: As depicted in Figure 2a, the controllers in the distributed (flat) control plane architecture will be coordinating together, where each controller in the network will have to be synchronized to have the same global view of the network. This approach has a great advantage to manage the network efficiently, since all of the controllers have an up-to-date state information of the complete network. However, synchronization of the controllers is a significant challenge, this is due to the time and resources necessary to handle all of the controller’s routine tasks while additionally working to communicate with all other controllers to update the global view. The distributed controller design mitigates the existing SDN problem of being centralized with a single point of failure, but at the added cost of additional tasks for controller synchronization.

2) Hierarchical Controller Design: As depicted in Figure 2b, the hierarchical controller design is implemented through the use of a master controller and several secondary controllers. The master controller oversees the entire network, and performs real-time data collection of the network traffic which allows for reconfiguration of the secondary controllers. The secondary controllers on the other hand would monitor and control non-overlapping subsections of the network. This in turn gives the master controller the capability to quickly and easily manage the network, while the secondary controllers do not add any additional overhead necessary for implementation of multiple controllers.

B. MARS

We use a three phases iterative approach: Logging and data collections, Machine learning model, Decision making and Network update. An overview of our proposed approach is presented in Figure 3.

1) Logging and data collection: Logging and data collection are critical regardless of the controller architecture selection. As stated, the use of machine learning requires vast amounts of data and analysis to reconfigure the network in the most advantageous manner. To accomplish this there will be multiple forms of logging and data collection operating in parallel such that the appropriate data is being collected. Data collection will be performed on the data plane as well as on the control plane, due to the multiple levels that exist in this design. Multiple forms of logging and data collection will be explored to aid in the determination of the amount of data and the level of detail that is necessary, both of which will provide insight to potential reconfigurations that need to be performed. To achieve this, we will use OpenFlow statistic calls, which will capture the flow statistics of SDN switches. Openflow statistic calls can be executed by either individual controller or the application itself and allows for easy implementation across the infrastructure. Also, we will use the traffic analysis tool (Zeek) [24], which runs on the network switching equipment and will be used to collect and analyze switch port statistics.

2) Pattern recognition through machine learning: In order to conduct network optimization, system must derive and analyze frequent patterns from the traffic that was collected from the network. Our plan is to identify frequent networking events using a Frequent Pattern Mining (FPM) model [25] followed by the gradient boosted tree model for event classification [26]. In our case, these events would be occurrences such as source and destination IP addresses, action types along with how often they occur with each other. This will allow us to understand the users’ and machine to machine behavior, and allow us to further identify traffic patterns, which will aim in better network reconfiguration. Thus, between FPM and GBT, data that is collected from the SDN can be analyzed and used in the next phase of the process.

3) Decision making and network update: When the initial setup stage is completed, the logging and data collection process is defined, and the machine learning model trained, then the system enters the decision making and reconfiguration phase. The decision making will be followed by network update process. The machine learning algorithms will process network traffic and generate clusters consisting of nodes communicate frequently. This information will be used together with other network statistics including network delay, congestion and link failures, to decide switch - controller pairs. When a more efficient configuration is determined, it will be pushed to switches to reflect the latest network topology. Our system will continuously collects network statistics to keep up with any changes in the network demand and to make sure to utilize network resources efficiently to satisfy end user needs.

V. DISCUSSION

To accomplish a robust and adaptable network control plane that utilizes network resources efficiently is needed. Our system can be optimized for different network needs

0589
to increase overall performance of any system that relies on effective network communication. It can enable a better network management for Wide Area Networks to utilize available bandwidth efficiently and reduce network delay. It can be used to reduce energy consumption in low power - narrow band networks, such as IoT, by minimizing network control plane communication overhead. It can increase the overall performance of a datacenter or a cloud system by increasing throughput of its internal network. It can help efficient utilization of heterogeneous wireless network resources to service mobile nodes in 5G networks by adopting continuous network (topology) changes. Additionally, proposed system provides a better visibility of the network to use for faster and effective network threat detection and mitigation by clustering frequently communicating nodes under one controller.

Our research presents a novel network control plane design to establish smart networks by utilizing SDN and machine learning. This system will reconfigure network control plane based on topology changes, traffic distribution on the network and user behavior that is extracted using machine learning algorithms. It is anticipated that the design will provide a better infrastructure management and visibility and increased efficiency in network resource utilization such as bandwidth. Additionally, it is expected to increase the SDN control plane scalability and energy efficiency by reducing control plane communication overhead. Last but not least, the system will add an auto-disaster recovery ability to the network in use.

To achieve a better understanding of the system and how it will perform under different scenarios, we will conduct multiple phases of evaluation and experimentation, starting with advanced simulation and later be extended to real-life experiments through a lab test-bed. We will study different control plane topologies, therefore, we will setup different use cases to measure how the network will perform under different traffic conditions as well as how it will perform in case of a controller failure. These simulations can put the system under the harshest conditions and will provide the evidence of network demands of the entire system, as well as providing insight into the computational challenges. We will quantify scalability of different architectures by using metrics commonly used to measure SDN controller performance such as throughput, number of flow requests handled per second, and low setup latency, delay to respond a flow request. We will investigate the pros and cons of these approaches and decide the best one for a dynamic network control plane.

VI. SYSTEM DESIGN AND IMPLEMENTATION

We developed a testbed that bridges software-defined networking (SDN) capabilities, through the use of Raspberry Pis including Raspberry Pi SDN based switches, SDN based controllers. Our testbed can be separated into two components based on SDN architecture. The first component is the control plane, which hosts our SDN Floodlight controllers that run on Raspberry Pi 3s. The data plane consists of both inner switches and outer switches on the network. Inner switches are devices that only contain switch to switch connections while outer switches contain switch to host connections. All network switches are hosted on Raspberry Pi 3s using USB to gigabit adapters for link connections. Hosts on our test bed consisted of both regular PCs and Raspberry Pi computers. Hosts performed a variety of common network tasks to create a baseline for traffic analysis. Our SDN-based testbed is shown in Figure 4.

The Raspberry Pi SDN will serve as the base for our testbed. It will consist of multiple Raspberry Pi’s connected through Ethernet. On the software side of the network, we will be using Floodlight controller. The Floodlight system will be deployed on server computer and Raspberry Pis to ensure consistent performance. Other Raspberry Pis will serve as OpenFlow switches for our network and hosts. We will revise the controller design to implement multiple controller scenarios to cover the distributed controller design and the hierarchical controller design. Through our evaluation, will be using the Internet of Things as our use case. Beside the SDN-testbed, our lab includes sensor nodes that are constructed using an Arduino Uno ATMega328P micro-controller connected to sensors capable of measuring: temperature, pressure, humidity, and illumination. We will use the sensor collected data to generate traffic on the network, which will be used by hosts (Raspberry Pis) running and connected to the SDN testbed.

VII. CONCLUSION AND FUTURE WORK

While the current solution of networks have proven to be functional over the last several decades, it is time to enter the modern era of networking to allow for an easy to manage and scalable, but most importantly an adaptable and robust networks. Software defined networks are significantly better than the networks currently implemented, with the greatest benefit being the ability to reconfigure the network in order to allow for greater performance and efficiency in routing. But it has seen that SDNs have a few critical flaws including the scalability of the network and the obvious central design which can lead to an inoperable network if the SDN controller were to fail.

To combat these critical flaws, we discussed the implementation of two forms of a multi-controller SDN design. The distributed (flat) design and the hierarchical design, both of
which aim to solve the scalability issues that a single controller SDN is prone to having. The two implementations will be critically analyzed and compared based upon the latency, resource utilization and controller overload including flow requests, and flow setup latency aiming to reduce network latency, increase network scalability and efficient resource utilization.

Then taking the optimizations and scalability one step further by applying different models of machine learning to both multi-controller designs. Through the analysis of live network traffic, features can be extracted such as common user behavior and machine to machine interactions. This will lead the machine learning algorithms to implement real-time configuration changes in the network. This not only allows for single controller optimizations, but also allow for the multiple controllers in the network to be optimized such that utilization of resources can be more efficient.

Development and implementation of the proposed system on our testbed is a work in progress. Machine learning models will be evaluated further and network traffic feature extraction and classification will begin to take way. Upon which we will introduce many different forms of network traffic and furthermore we will introduce failures into the network controller(s) to test the adaptability and robustness of the network management system. Using the reporting dashboard and evaluation of which machine learning model (both supervised and unsupervised) lead to the greatest utilization of resources during network reconfiguration.

ACKNOWLEDGEMENT

The authors acknowledge support from the University of Tennessee at Chattanooga and The Scientific and Technological Research Council of Turkey (TUBITAK). Research reported in this publication was supported by the 2020 Center of Excellence for Applied Computational Science and Engineering grant competition (CEACSE) and The Scientific and Technological Research Council of Turkey (TUBITAK).

REFERENCES

[1] Cisco. The zettabyte era: Trends and analysis. Cisco, page 7, June 2017.
[2] D. Kreutz, F. M. V. Ramos, P. E. Veríssimo, C. E. Rothenberg, S. Azodolmolky, and S. Uhlig. Software-defined networking: A comprehensive survey. Proceedings of the IEEE, 103(1):14–76, Jan 2015.
[3] T. Wang, F. Liu, and H. Xu. An efficient online algorithm for dynamic sdn controller assignment in data center networks. IEEE/ACM Transactions on Networking, PP(99):1–14, 2017.
[4] Bob Lantz, Brandon Heller, and Nick McKeown. A network in a laptop: rapid prototyping for software-defined networks. In Proceedings of the 9th ACM SIGCOMM Workshop on Hot Topics in Networks, page 19. ACM, 2010.
[5] Fábio Botelho, Alysson Bessani, Fernando MV Ramos, and Paulo Ferreira. On the design of practical fault-tolerant sdn controllers. In 2014 third European workshop on software defined networks, pages 73–78. IEEE, 2014.
[6] Jianli Pan, Subbarthi Paul, and Raj Jain. A survey of the research on future internet architectures. IEEE Communications Magazine, 49(7):26–36, 2011.
[7] Alex Galis, Javier Rubio-Loyola, Stuart Clayman, Leifertis Mamatas, Slawomir Kukinski, Joan Serrat, and Theodore Zohariadis. Software enabled future internet–challenges in orchestrating the future internet. In International Conference on Mobile Networks and Management, pages 228–244. Springer, 2013.
[8] Barath Raghavan, Martin Casado, Teemu Koponen, Sylvia Ratnasamy, Ali Ghodsi, and Scott Shenker. Software-defined internet architecture: decoupling architecture from infrastructure. In Proceedings of the 11th ACM Workshop on Hot Topics in Networks, pages 43–48. ACM, 2012.
[9] K. Phemius, M. Bouet, and J. Leguay. Dsco: Distributed multi-domain sdn controllers. In 2014 IEEE Network Operations and Management Symposium (NOMS), pages 1–4, May 2014.
[10] Advait Dixit, Fang Hao, Sarit Mukherjee, T.V. Lakshman, and Ramana Kompella. Towards an elastic distributed sdn controller. In Proceedings of the Second ACM SIGCOMM Workshop on Hot Topics in Software Defined Networking, HotSDN ’13, pages 7–12, New York, NY, USA, 2013. ACM.
[11] Soheil Yeganeh and Yashar Ganjali. Kandoo: A framework for efficient and scalable offloading of control applications. HotSDN’12 - Proceedings of the 1st ACM International Workshop on Hot Topics in Software Defined Networks, 08 2012.
[12] Volkan Yazici, M Oguz Sunay, and Ali O Ercan. Controlling a software-defined network via distributed controllers. arXiv preprint arXiv:1401.7651, 2014.
[13] Md Faizul Bari, Arup Raton Roy, Shihabur Rahman Chowdhury, Qi Zhang, Mohamed Fateh Zhan, Reaz Ahmed, and Raouf Boutaba. Dynamic controller provisioning in software defined networks. In CNMS, pages 18–25, 2013.
[14] Murat Karakus and Arjan Durresi. A scalable inter-as qos routing architecture in software defined network (sdn). In 2015 IEEE 29th International Conference on Advanced Information Networking and Applications, pages 148–154. IEEE, 2015.
[15] Soheil Hassas Yeganeh and Yashar Ganjali. Kandoo: a framework for efficient and scalable offloading of control applications. In Proceedings of the first workshop on Hot topics in software defined networks, pages 19–24. ACM, 2012.
[16] Abdelhadi Azzouni, Raouf Boutaba, and Guy Pujolle. Neuroute: Predictive dynamic routing for software-defined networks. CoRR, abs/1709.06002, 2017.
[17] Gagan Choudhury, David Lynch, Gaurav Thakur, and Simon Tse. Two use cases of machine learning for sdn-enabled ip/ optical networks: Traffic matrix prediction and optical path performance prediction, 2018.
[18] Junfeng Xie, Fei Richard Yu, Tao Huang, Renchao Xue, Jiang Liu, Chenmeng Wang, and Yunjie Liu. A survey of machine learning techniques applied to software defined networking (sdn): Research issues and challenges. IEEE Communications Surveys & Tutorials, 21:393–430, 2019.
[19] Álvaro López-Raventós, Francesc Wilhelmi, Sergio Barrachina-Muñoz, and Boris Bellalta. Machine learning and software defined networks for high-density wans. 04 2018.
[20] Pedro Amaral, Joao Dinis, Paulo Pinto, Luis Bernardo, Joao Tavares, and Henrique S Mamede. Machine learning in software defined networks: Data collection and traffic classification. In 2016 IEEE 24th International Conference on Network Protocols (ICNP), pages 1–5. IEEE, 2016.
[21] Tam N. Nguyen. The challenges in SDN/MPLS based network security : A survey. CoRR, abs/1804.03539, 2018.
[22] S. Nanda, F. Zafari, C. DeCusatis, E. Wedaa, and B. Yang. Predicting network attack patterns in sdn using machine learning approach. In 2016 IEEE Conference on Network Function Virtualization and Software Defined Networks (NFV-SDN), pages 167–172, Nov 2016.
[23] Murat Karakus and Arjan Durresi. A survey: Control plane scalability issues and approaches in software-defined networking (sdn). Computer Networks, 112:279–293, 2017.
[24] The Zeek Project. Zeek manual. https://docs.zeek.org/en/stable/. Accessed: 2019-10-15.
[25] Charu C Aggarwal, Yan Li, Jianyong Wang, and Jing Wang. Frequent pattern mining with uncertain data. In Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining, pages 29–38. ACM, 2009.
[26] Si St, Huan Zhang, S Sathiya Keerthi, Dhruv Mahajan, Inderjit S Dhillon, and Cho-Jui Hsieh. Gradient boosted decision trees for high-dimensional sparse output. In Proceedings of the 34th International Conference on Machine Learning-V olume 70, pages 3182–3190. JMLR. org, 2017.