Hybrid SDN Networks: A Survey of Existing Approaches
Rashid Amin, Martin Reisslein, and Nadir Shah

Abstract—Software Defined Networking (SDN) decouples the control plane from the data plane of forwarding devices. This separation provides several benefits, including the simplification of network management and control. However, due to a variety of reasons, such as budget constraints and fear of downtime, many organizations are reluctant to fully deploy SDN. Partially deploying SDN through the placement of a limited number of SDN devices among legacy (traditional) network devices, forms a so-called hybrid SDN network. While hybrid SDN networks provide many of the benefits of SDN and have a wide range of applications, they also pose several challenges. These challenges have recently been addressed in a growing body of literature on hybrid SDN network structures and protocols. This article presents a comprehensive up-to-date survey of the research and development in the field of hybrid SDN networks. We have organized the survey into five main categories, namely hybrid SDN network deployment strategies, controllers for hybrid SDN networks, protocols for hybrid SDN network management, traffic engineering mechanisms for hybrid SDN networks, as well as testing, verification, and security mechanisms for hybrid SDN networks. We thoroughly survey the existing hybrid SDN network studies according to this taxonomy and identify gaps and limitations in the existing body of research. Based on the outcomes of the existing research studies as well as the identified gaps and limitations, we derive guidelines for future research on hybrid SDN networks.

Index Terms—Hybrid SDN network, Network management, Software Defined Networking (SDN), SDN controller, Traffic engineering.

I. INTRODUCTION

SOFTWARE Defined Networking (SDN) [1]–[4] has emerged as a new networking paradigm that decouples the network control (management) plane from the network data plane. In networks based on the SDN paradigm, which are commonly referred to as “SDN networks”, the controller distributes forwarding rules to the network devices to handle flows initiated by users [5]. The control of routers/switches is typically implemented at a central device that is called controller [6] (distributed controllers are possible, see Sec. VII-H). The switches have a uniform programming interface, e.g., an interface based on the OpenFlow protocol [7]–[10], and have the capability to observe and forward the network traffic flows according to the rules prescribed by the controller. This separation of the network control and data planes eases the network control and management [11]–[13].

SDN networks have many advantages over traditional networks, e.g., ease of network management and enforcement of security policies [14]. However, SDN is typically not fully deployed in networks due to several reasons. One major reason is the limited budget for new network infrastructure. Organizations are often reluctant to invest large budgets on installing a new network infrastructure from scratch [15]. Another reason is fear of downtime during the transition to SDN. One possible solution to address these concerns is to deploy a limited number of SDN-enabled devices [16] alongside the traditional (legacy) network devices; thus, incrementally replacing traditional network devices by SDN devices. A network containing a mix of SDN and legacy network devices is commonly referred to as a hybrid SDN network [15], [17]. Fig. 1 shows an example hybrid SDN network consisting of OpenFlow SDN switches alongside legacy switches. The controller in Fig. 1 can be a hybrid SDN network controller [18] with specific features to adopt hybrid SDN network protocols, as surveyed in detail in this article. Table I contrasts the main characteristics of hybrid SDN networks from pure SDN networks and traditional networks.

A. Advantages of Hybrid SDN Networks

Hybrid SDN networks offer a range of advantages:

1) SDN network deployment is financially very costly. To replace all the existing legacy devices by SDN devices, large budget amounts are required to purchase SDN devices. After full deployment of SDN, i.e., after creating a pure SDN network, additional budget amounts are
required to train the operators to design, configure, and operate the SDN network [14]. Hybrid SDN networks ease these budget concerns.

2) Hybrid SDN networks can be used to reap some of the benefits of the SDN paradigm without deploying a full SDN network [19]–[22]. For example, an Internet Service Provider (ISP) usually handles millions of forwarding entries; however, SDN-enabled switches can typically support only tens of thousands of forwarding entries [21]. As illustrated in Fig. 2, the ISP access network can use legacy devices while the ISP distribution network can use SDN devices. Thus, the ISP can use a hybrid SDN network to handle millions of forwarding entries in the access network through the legacy devices while SDN devices are used in the distribution network to reap the benefits of SDN.

3) SDN provides fine-grained control for data traffic flows. If fine-grained control in only required for a small portion of the network, then a hybrid SDN network can be implemented by executing SDN for that small portion of network [21] requiring fine-grained control, while the rest of the network uses traditional networking. For example, if an organization with the network setup shown in Figure 3 requires fine-grained control for the portion of the network in the green rectangle, then only that portion will be upgraded to SDN.

4) Traditional routing protocols are very effective for some tasks, e.g., for in-band connectivity between forwarding devices and the SDN controller, as well as for connectivity among SDN controllers to control different parts of the network. Thus, a hybrid SDN network can be deployed to relieve the SDN controller from tasks that can be effectively conducted by traditional routing protocols.

5) The SDN architecture has emerged recently. Thus, SDN devices are not as mature as legacy networking devices. Network operators may therefore hesitate to replace the complete traditional network at once. Hybrid SDN networks ease the transition from legacy to SDN network devices. With a hybrid SDN network, a network operator can incrementally deploy more and more SDN devices and evaluate SDN performance in practice. For instance, Google [23] has adopted SDN in multiple stages over several years for their network management and control. Similarly, other large, as well as small and medium size organizations may want to adopt SDN technology in a gradual manner.

6) Scenarios where two SDN networks are interconnected by legacy network devices require hybrid SDN network mechanisms to properly assign resources for the SDN network interconnection.

B. Industry and Standardization Perspectives

Motivated by the advantages of hybrid SDN networks that have been summarized in the preceding section, several industry white papers and documents of standardization groups, such as the Open Networking Foundation (ONF) and the Internet Engineering Task Force (IETF), have advocated the study and deployment of hybrid SDN networks. This section briefly summarizes these industry white papers and documents from the ONF and IETF, which have been mainly published in the 2014–2016 time frame. These white papers and IETF and ONF documents have helped spur the extensive study of hybrid SDN networks in the time frame from about 2014 to the present, as surveyed in the subsequent sections of this article.

Major network equipment manufacturers as well as network consulting firms and network operators have advocated for hybrid SDN networks. For instance, the Allied Telesis white paper [24] recommends that the migration from a traditional network to an SDN based network should occur gradually. Allied Telesis recommends a migration to an SDN based network in multiple steps (phases). Each step towards SDN control should addresses a particular need, and provide a commensurate benefit. The gradual migration helps avoid potential problems and outages while providing the benefits.
of SDN control, such as reduced costs, increased flexibility, enhanced security, and improved user experience, in a step-by-step manner. Along the same lines, the network analytics and management company Entuity noted in a white paper [25] that many organizations do not adopt SDN due to concerns about the budget and downtime. Entuity advocated a gradual adoption of SDN in the form of hybrid SDN networks.

Huawei Technologies Co. Ltd. was one of the first communications equipment manufacturers to report on hybrid SDN networks in a newsletter in 2013 [26]. The newsletter reported about Huawei’s participation in an ONF PlugFest test in October 2012, where Huawei exhibited a Smart Network OpenFlow Controller (SOX) for controlling hybrid SDN networks.

The NEC Corporation white paper [27] has posed and addressed a common question about SDN: “Do we have to replace the entire existing network to introduce SDN?”. In response to this questions, NEC recommends hybrid SDN networks that partially introduce SDN without replacing the entire existing network. The hybrid SDN networks enable SDN to exert its control and achieve a wide range of benefits while coexisting with the existing network. This recommendation is based on NEC’s extensive experience in the area of SDN [27]. Similarly, a technical white paper from Hewlett-Packard (HP) [28] advocated hybrid SDN networks. HP advocated that the SDN controller should control all packet forwarding on the data plane; however, the SDN controller should delegate some forwarding decisions to the controlled switches to reduce the scope of the forwarding decisions made at the controller, reduce the control plane traffic, and utilize the existing network equipment. Moreover, the Cisco report [29] considers the evolution of MPLS networks to MPLS-SDN hybrid networks that employ MPLS in conjunction with SDN.

Turning to the network operator perspective, the report [30] sponsored by AT&T advocated hybrid SDN networks for cloud networks and big data applications. Hybrid SDN networks can enable enterprises to procure large amounts of on-demand bandwidth capacity in real-time at relatively low cost. The report for AT&T emphasized that hybrid SDN networks offer network programmability to provision network resources and to centrally manage performance and security aspects in a matter of minutes, while utilizing the installed distributed traditional networks. The very extensive report [31] from the network infrastructure planning group at Verizon specifies SDN-NFV reference architectures that outline a range of approaches for SDN deployment. The Verizon report noted the hybrid SDN network approach as a particularly viable solution. In Verizon’s reference architecture for hybrid SDN networks, SDN controllers cooperate with existing forwarding boxes and, optionally, vendor-specific domain controllers.

The Open Network Foundation (ONF) document [32] specifies hybrid SDN networks in the Section “Traditional Networking Coexistence and Migration (“Hybrid”). This section explains that hybrid SDN networks are a viable and acceptable solution for organizations to move towards SDN. The section also discusses some scenarios and issues that need to be addressed to ensure successful migration from and coexistence with traditional networking technologies. The IETF informational request for comments (RFC) [33] noted that SDN will likely be deployed progressively across the various network and service segments. The RFC also acknowledged that SDN devices will likely have to coexist with legacy networks in the form of hybrid SDN networks.

C. Different Forms of Hybrid SDN

Hybrid Software Defined Networking (hybrid SDN), i.e., the combination of legacy (pre-SDN) networking principles with SDN networking principles can take on different forms. The major categories of hybrid SDN are:

- Deployment of SDN switches in a legacy network, i.e., among legacy switches, to form a hybrid SDN network.
- Hybrid SDN switches having both SDN switching and legacy switching functionality.

1) Deployment of SDN Switches in Legacy Network: Hybrid SDN Network: In this form of hybrid SDN, SDN switches are placed in a legacy network, e.g., among legacy IP switches, to form a so-called hybrid SDN network. This form of hybrid SDN, i.e., the hybrid SDN network is the main focus of this survey. By forming a hybrid SDN network, old legacy switches can be used to realize SDN-like control and management in a legacy network. Thus, hybrid SDN networks offer several attractive advantages, as summarized in Section I-A.

2) Hybrid SDN Switches with Both SDN and Legacy Switch Functionalities: Network switches can be equipped with both SDN and legacy switch functionalities to form hybrid SDN switches.

Addressing concerns about network failure recovery and robustness, Tilmans and Vissicchio [34] have proposed an Interior Gateway Protocol (IGP) [35] based backup network for robustness in pure SDN networks. In particular, distributed legacy protocols quickly repair short-term network failures. An SDN controller then configures the routing paths while avoiding the limitations of the distributed legacy protocols so as to achieve optimal network operation in the long term. In each hybrid SDN switch, local agents are configured to exchange the routing information and back-up routes are established with the distributed legacy IGP. When a link failure occurs, these local agents use the IGP routing information to reconnect the network, i.e., to recover the network connectivity.
This way, link failures in pure SDN networks are quickly repaired with the back-up routes obtained through a legacy switch functionality.

Vissicchio et al. [36] have examined the cooperation of distributed (e.g., OSPF based) and centralized (SDN based) routing control planes in hybrid SDN switches. This cooperation can achieve several benefits, including improved robustness and traffic engineering. Moreover, hybrid SDN switches running both traditional and SDN routing protocols can improve routing flexibility as compared to a pure IGP network and can be used to deploy network function virtualization. However, the cooperation of the distributed and centralized control planes may give rise to novel forwarding anomalies. Vissicchio et al. [36] have presented a theoretical framework to address these forwarding anomalies and have derived sufficient conditions to achieve anomaly-free routing control plane cooperation. Similarly, Xu et al. [37] have employed hybrid switching for scalability.

A hardware SDN module can be installed in legacy switches to enable SDN functionalities. By installing the hardware SDN module, legacy switches can operate with both legacy network protocols and SDN-based protocols. Only some of the legacy switches in a given network may be upgraded with the hardware SDN module, resulting in a hybrid SDN network formed by legacy switches and hybrid switches, i.e., legacy switches that have been upgraded with the hardware SDN module. Accordingly, we include the hardware SDN module option in our survey, see Section II-C.

**D. Comparison with Existing Surveys**

The general principles of SDN have covered in several surveys that appeared as early as 2014, e.g., in [2], [38]–[52]. Recently, an extensive survey literature has covered the various aspects of SDN in detail. The control plane of SDN has been surveyed in [53], [54], related surveys covered SDN service orchestration [55] and quality of service in SDN networks [56]. Software engineering issues and the programmability of SDN networks have been surveyed in [57]–[59], while enhancements to OpenFlow in form of protocol-independent and protocol-oblivious forwarding have been covered in [60], [61]. A few surveys have covered the SDN aspects related to elementary functions in networks, ranging from topology discovery [62], [63] and network updates [64] to routing [65] and traffic engineering [66], as well as fault management [67], monitoring [68], and security [43], [69]–[74]. There have also been several surveys that cover SDN in different networking contexts; specifically SDN in wide area networks [75], access networks [76], mobile and wireless networks [77]–[83] optical networks [3], [84]–[86], as well as the Internet of Things (IoT) [87], [88], cyber-physical systems [89], and smart cities [90], [91]. SDN network testbeds have been surveyed in [92]. Several surveys have covered the virtualization of SDN networks [93], [94] and SDN network functions [95]. In particular, wireless network virtualization has been covered in several surveys, e.g., [96]–[99], while virtual network embedding has been surveyed in [100].

Cox et al. [101] have recently surveyed the range of opportunities and research challenges arising from the deployment of pure SDN networks in government, industry, as well as small and large-scale organizations. After a thorough survey of these opportunities and research challenges or pure SDN networks, Cox et al. outline the concept of hybrid SDN networks and briefly mention a few early hybrid SDN network studies, such as Levin et al. [17], [102] and Vissicchio et al. [14]. However, Cox et al. [101] do not conduct a detailed survey of hybrid SDN networks.
To the best of our knowledge the topic area of hybrid SDN networks has so far only been covered by two surveys. Vissicchio et al. [14] have briefly discussed the operation of traditional and SDN networks as well as the possible opportunities and research challenges in the hybrid SDN network area. Based on the respective roles assigned to the traditional and SDN networks, Vissicchio et al. described several potential hybrid SDN models, including the topology based model, service-based model, traffic class-based model, and integrated model. The topology-based model divides the network into different zones, whereby each zone is either SDN-based or based on a traditional network. The service-based model provides some services, such as network-wide forwarding, with SDN, while providing other services with traditional networking. The class-based model uses SDN for a certain traffic class, e.g., TCP traffic, while using traditional networking for other traffic classes. The overview of the hybrid SDN concept from the perspective of the topology-based, service-based, and traffic class-based modeling perspective has recently been elaborated in the survey by Sandhya et al. [103]. Sandhya et al. also provided a topology perspective on hybrid SDN network deployments and addressed topology discovery in detail. Complementarily to the two existing surveys [14], [103], we provide a general survey on hybrid SDN networks that is organized according to the elementary functions of network deployment, network control, network management, traffic engineering, as well as network testing, verification, and security.

Importantly, the two existing hybrid SDN network surveys present only a selected set of research studies. In particular, Vissicchio et al. [14] have only surveyed a few seminal hybrid SDN network studies, e.g., [15], [102]. Sandhya et al. [103] have covered a more extensive selection of studies, but have not surveyed the hybrid SDN network topic comprehensively. For instance, Sandhya et al. have included only six network control studies, whereas our comprehensive survey identifies and surveys 18 studies on hybrid SDN network control. Overall, our survey is the first to provide comprehensive up-to-date coverage of hybrid SDN network studies.

E. Contributions

Although hybrid SDN networks have many applications and benefits, they pose several challenges that need to be addressed by novel techniques. In particular, the architecture of a hybrid SDN network is different from the architectures of both pure SDN networks and traditional networks, as illustrated in Figure 4. Novel networking mechanisms and protocols are needed to design and operate these hybrid SDN networks. Recently, a rapidly growing literature has developed approaches for addressing the challenges posed by hybrid SDN networks. This article presents a comprehensive up-to-date survey of the research and development in the field of hybrid SDN networks. We organize the field of hybrid SDN networks according to the network design and operating aspects into five main categories that are illustrated in Figure 5:

- Network management techniques for hybrid SDN networks
- Traffic engineering mechanisms for hybrid SDN networks
- Testing/verification and security mechanisms for hybrid SDN networks

After thoroughly examining the existing hybrid SDN network research according to this taxonomy, we identify the gaps in the existing literature. We outline the future research directions that can address the gaps in the existing literature and advance the research area of hybrid SDN networks.

F. Paper Organization

The remainder of this paper is organized as follows. Section II surveys the different deployment strategies of SDN devices among traditional devices so as to form hybrid SDN networks. Section III surveys controllers for hybrid SDN networks, while network management techniques for hybrid SDN network are surveyed in Section IV. Section V surveys traffic engineering mechanisms for hybrid SDN networks. Section VI gives an overview of testing and verification mechanisms as well as security mechanism for hybrid SDN networks. Section VII outlines future research directions for hybrid SDN networks. Section VIII concludes the paper.

II. HYBRID SDN NETWORK DEPLOYMENT

A. Overview

To provide SDN-like control and management in a traditional network, a limited number of SDN devices can be placed among legacy devices, so as to form a hybrid SDN network. Hybrid SDN network deployment studies have examined different options for placing a limited number of SDN devices in a traditional network. In this section, we survey the existing studies on the deployment of SDN switches in a hybrid SDN network following the classification in Fig. 6.

B. Network Architecture and Design Focused Studies

The Panopticon approach of Levin et al. [15], [17] is a seminal approach for designing hybrid SDN networks. Panopticon forms a logical (virtual) SDN network by interconnecting legacy and SDN devices to enable the benefits of a pure SDN network. (The term “Panopticon” refers to a prison architecture where the prison cells are arranged along a circle so that they can all be observed and controlled from a central watch tower.) With Panopticon, all applications that have been designed for a pure SDN network can be installed on the logical SDN network. The main idea behind Panopticon is the waypoint enforcement technique, which forces every packet traversing from source to destination to traverse an SDN switch [118]. Upon reaching the SDN switch, the SDN controller handles the packet. Thus each packet is processed as it would be in a pure SDN network.

A Panopticon example with the physical and logical network structures is shown in Figure 7. The example has eight legacy switches and two SDN (OpenFlow) switches. Fig. 7(a) shows the Solitary Confinement Trees (SCTs) that are overlaid on the physical topology to connect every SDN controlled (SDNc)
Figure 6. Overview of hybrid SDN network deployment studies.

Figure 7. Example illustration of waypoint enforcement in Panopticon [15], [17]: Each packet is forced to traverse an SDN switch by overlaying Solitary Confinement Trees (SCTs) on the physical topology, see part (a), to connect every SDN controlled (SDNc) port to an SDN switch, resulting in the logical view illustrated in part (b). In part (b), each SDNc port is virtually connected to at least one SDN switch. An SCT is a spanning tree that is identified through a Virtual Local Area Network identifier (VLAN ID) [119], [120]. In Fig. 7(a), SCT (A) involves the routes 5 → 1 → 2 and 5 → 3 → 4 to both SDN switches, while SCT (B) has the route 6 → 2 to the top SDN switch. Fig. 7 (b) shows the corresponding logical view where each SDNc port is virtually connected to at least one SDN switch. Levin et al. [15], [17] have demonstrated the functionalities of logical SDN through the study of a few use cases and a set of experimental results. The experimental results have demonstrated that when 10% SDN switches are deployed among legacy switches then about 25% of the traffic is handled by SDN nodes. The results indicate a 40% performance increase compared to the original network.

Poularakis et al. [104] have examined the evolution of ISP networks towards SDN and have concluded that ISPs can gradually upgrade their entire networks to SDN-based networks over long time periods that may span several years. Based on different network topologies and traffic matrices, Poularakis et al. suggest that the upgrade procedure should not be in one step. They present a general model for different migration scenarios and the associated costs. For the hybrid SDN network created by the migration scenarios, network traffic is classified into two categories: programmable traffic traverses at least one SDN switch and non-programmable traffic does not traverse any SDN switch. Two main objectives are considered: first maximizing the programmable traffic and second maximizing the traffic engineering (TE) flexibility by providing alternative paths through SDN upgrades. A variant of the Budgeted Maximum Coverage Problem (BMCP) [121] is used to formulate the programmable traffic problem. The evaluation of this approach shows that there is an increase of 54% in programmable traffic which facilitates effective TE.

Xu et al. [105] have studied hybrid optical SDN deployment and the placement of SDN devices among legacy devices so that more and more SDN benefits can be achieved. In
most hybrid SDN deployments, SDN nodes are deployed by replacing legacy devices such that the maximum traffic portion traverses the SDN devices so that the maximum traffic portion can be managed by the SDN controller. Xu et al. [105] propose a duplicate deployment scheme that places SDN devices “in addition” and not “instead of’ legacy devices. This duplicate deployment does not change the legacy network; rather, new SDN devices are placed and are connected to one or multiple legacy devices. In this hybrid network, paths for all traffic flows are jointly decided by distributed routing protocols and the SDN controller. For throughput maximization under a limited budget, Xu et al. formulate a joint duplicate deployment and routing optimization problem. Xu et al. propose to solve this problem through traffic mapping and a randomized rounding based approximation algorithm. The approximation factor for this algorithm is $O(\log n)$ in the worst case, where $n$ indicates the total number of devices e.g., the total number of legacy routers and SDN switches. The proposed approach achieves 26% throughput improvement compared to traditional routing. The duplicate deployment proposed by Xu et al. [105] has the main benefit of improving network resource utilization and throughput without the need to alter the legacy network structure. Also, the capacity provided by the legacy devices continues to be utilized for network transport services.

Caria et al. [106] have presented a hybrid SDN network partitioning model for dynamic optical circuits. The partitioning model in [106] is based on an earlier model by Caria et al. [112] for partitioning a hybrid SDN network by dividing an OSPF network into several subdomains, see Section II-D. The OSPF subdomains are interconnected via SDN switches that are directly controlled by an SDN controller. The approach in [106] combines the SDN partitioning with dynamic optical circuits that operate as optical bypasses. That is, the original OSPF partitioning is extended by optical bypasses that provides efficient network control for network operators. By operating optical bypasses among SDN-based border nodes of sub-domains, these sub-domains can support heavy network traffic and achieve optimum resource utilizations. With the Caria et al. [106] approach, network operators do not need to completely migrate to optical transport or to SDN-only transport. Simulation results indicate that the proposed mechanism improves the network management and control and achieves efficient link utilization in the hybrid SDN network.

Martinez et al. [107] have presented a service-based model for a hybrid SDN wireless mesh backhaul network that combines the benefits of SDN centralization with the distributed network nature of traditional networks. The wireless mesh backhaul architecture consists of three layers. In a lower layer, low-cost capacity is aggregated through the cooperation of sub-6 GHz and millimeter wave technology with microwave links. Distributed non-delay tolerant services and centralized delay tolerant services form a model that functions as a control layer for this network. At the upper (application) layer, routing and energy efficient network services are deployed. The simulation results indicate that this service based hybrid SDN model has high performance as compared to a centralized SDN model by lowering latency to one sixth. This model operates smoothly with scalable distributed protocols even if the SDN controller is down.

He et al. [108] have developed a new SDN/IP network architecture by creating an overlay hybrid network with satellites in outer space. The architecture facilitates the resolution of the problems and inconsistencies of hybrid SDN by collaboratively managing both networks. The traditional IP based network is a terrestrial network created on the ground, and the SDN based centralized network is formed in the satellite network. Several IP networks are formed on the ground and are administered by SDN control at the overlay satellite level. A new routing model is adopted by using sockets and OpenFlow. First, the respective forwarding entries are installed by the SDN controller at the switches and the packets are forwarded according to a routing protocol that is managed by the routing engine server. Routing protocol packets are parsed by Quagga [122] and the routing information is forwarded to the SDN controller which has a global view of the entire network.

C. Hardware Shim

By installing a hardware SDN shim [109] in legacy switches, the legacy switches can operate with both legacy network protocols and SDN-based protocols. The hardware shim enables a legacy switch to exchange its routing and forwarding control information with the SDN controller. Thus, the SDN shim effectively provides SDN features in legacy switches and gives a hybrid SDN environment. The SDN shim design is illustrated in Figure 8. The legacy switch is pre-configured to connect to the shim and to access ports through VLAN trunks. Thus, all switch traffic can reach the shim and is handled by the shim according to the SDN control. The limitation of the shim module is that every specific type of legacy switch requires a corresponding specific SDN-shim device. Hence, the hardware shim solution requires extensive efforts and a high budget to accommodate a wide variety of legacy switch types.

D. ILP Optimization

Tamal et al. [110] have presented a complete migration trajectory for replacing legacy routers/switches in a network by
SDN switches/routers. If a network operator wants to replace a legacy router with an SDN router, then the operator should select the legacy router for replacement that yields the highest network performance increase. The migration trajectory presented in [110] provides the complete schedule to replace legacy switches/routers with SDN switches/routers. Based on this migration trajectory, Tamal et al. [110] discuss scenarios for Internet Service Providers (ISPs) with a limited investment budget for SDN devices. The presented migration trajectory provides the best traffic engineering gain during the migration towards the SDN architecture. A greedy algorithm is used to obtain the optimal traffic engineering gain and an integer linear program (ILP) for the deployment trajectory is formulated. The main limitation of this approach is that the greedy algorithm employed for calculating the deployment of SDN devices may not give an optimum solution in some scenarios.

Lukovszki et al. [111] have studied the conditions for a minimum number of SDN switches to be deployed in the hybrid SDN network so that a distance constraint and a capacity constraint on the overall network deployment is satisfied. Lukovszki et al. [111] have provided an algorithmic solution for the placement of SDN switches and have made the first attempt to mathematically solve the problem of SDN device deployment. The provided mechanism can also be used to place SDN switches in an initial network deployment or in an already deployed network. The proposed solution provides a deterministic and greedy $O(\log(\min\{k, n\}))$ approximation algorithm for the number of nodes $n$ and capacity $k$ in polynomial time. The proposed approach uses a submodular function that computes the maximum number of pairs with a prescribed number of deployed SDN switches. The submodular function is expressed efficiently using an augmented path on a bipartite graph. The main limitation of this approach is that it becomes computationally demanding for large networks.

Caria et al. [112], [113] have developed a hybrid SDN/OSPF deployment approach in which the entire network is divided into several small-size networks (sub-domains) that are interconnected with SDN devices. This method of Caria et al. is quite different from other SDN deployment methods. The Caria et al. mechanism partitions the entire OSPF domain into sub-domains. The subdomains are inter-connected via SDN devices and there are no other links between these sub-domains. With this Caria et al. mechanism, SDN devices can easily be placed in an operational OSPF network. Caria et al. introduce a traffic engineering engine that balances traffic loads through Integer Linear Programming (ILP). The ILP formulation assumes that all OSPF paths inside the sub-domains are known and constant. With $l$ denoting a link that belongs to the set $L$ of links attached to SDN switches and $Cost_l$ representing the utilization cost of link $l$, the objective function is

$$\text{Minimize } \sum_{l \in L} Cost_l.$$  

Figure 9 illustrates the network formed by the Caria et al. mechanism which places SDN devices at all sub-domain edge nodes. In this network, SDN devices begin the update process in their own sub-domain by flooding routing updates that are independently modified for each specific sub-domain. In this way, Link State Advertisement (LSA) flooding is limited by SDN switches, and each SDN switch participates in the OSPF protocol of its corresponding sub-domains.

Figure 9. Caria et al. [112], [113]: Partitioning OSPF with SDN: Link State Advertisement (LSA) flooding is limited by SDN switches, and each SDN switch participates in the OSPF protocol of its corresponding sub-domains.

E. Optimization Heuristics

Hong et al. [114] have introduced a model for the incremental deployment of SDN switches in a hybrid SDN network. The model selects a subset of the legacy switches/routers to be replaced by SDN switches/routers, without changing the network topology, nor the configuration of the existing legacy switches or routers. The following techniques are used to achieve the replacement:

![System architecture components for the optimum heuristic by Hong et al. [114]: The offline incremental SDN Deployment Planner decides which legacy devices to upgrade to SDN devices, the Global Topology Viewer maintains a global view of the hybrid SDN network through interactions between legacy and SDN devices, the TE Module meets TE goals by controlling forwarding paths, and the Failover Module avoids link congestion under failures and ensures fast failure recovery.](image-url)
• According to some heuristic, the model replaces a minimum subset of the legacy switches/routers by SDN switches so as to optimize the cost effectiveness.
• The model maintains a real-time view of all network devices, i.e., both legacy and SDN network devices.
• An interoperability scheme controls the traditional network protocols and the SDN protocols to achieve desired traffic engineering (TE) criteria.

These techniques are implemented by the main system components illustrated in Figure 10:
1) The SDN deployment planner runs periodically and decides which legacy switches/routers can be replaced by SDN switches based on the network topology, traffic demand history, and resource constraints.
2) The Global Topology Viewer maintains a global view of the entire hybrid SDN network.
3) The Traffic Engineering Module maintains the TE requirements by controlling forwarding paths.
4) The Failover Module ensures quick recovery after failures and avoids congestion.

Hong et al. [114] have conducted an analysis of the proposed model considering the additional costs incurred by deploying SDN switches and the resulting TE benefits. The results indicate that the proposed model reduces link traffic by 32% as compared to a traditional network. Moreover, the analysis of the tradeoff between the SDN switch costs and the TE performance gains indicate that a hybrid SDN network consisting of 20% SDN switches and 80% legacy switches can achieve optimum performance in terms of network device cost and TE performance gains. A limitation of the David et al. model is that it does not consider testing and debugging other network invariants, such as black holes and loops.

Xu et al. [115] have presented incremental SDN deployment strategies for upgrading existing traditional networks and have introduced throughput-maximizing routing for hybrid SDN networks. A heuristic algorithm is proposed to deploy SDN switches among legacy switches under given budget constraints. Two deployment strategies are introduced, namely replaced deployment and incremental deployment. With replaced deployment, legacy devices are replaced with SDN devices. With incremental deployment, SDN devices are deployed among legacy devices so that the old legacy devices are not wasted. This study argues for the advantages of incremental deployment, which preserves the advantages and benefits of the legacy system. The introduced heuristic incremental deployment algorithm achieves an approximate factor of about $1 - 1/e$ for the budget constraints. After the incremental deployment of SDN devices among legacy devices, a set of routes for each flow is built that follows the network policies. For maximizing the throughput, an approximate multi-commodity $h$-splittable flow routing algorithm is executed. This algorithm is based on depth-first-search and randomized rounding techniques. Performance evaluations indicate that the incremental deployment can increase the throughput by about 40%. Moreover, the introduced routing algorithm can improve the throughput by approximately 31% as compared to the Equal Cost Multi-Path (ECMP) protocol [125].

Kar et al. [116] have examined the maximum coverage problem in hybrid SDN networks so as to address the efficient deployment of SDN switches among legacy switches. Based on the SDN switch deployment, three types of paths are generated: legacy paths, pure SDN paths, and hybrid SDN paths. If a path contains only legacy switches, then it is a legacy path. If a path contains only SDN switches, then it is a pure SDN path. If a path contains both types of switches, then the path is a hybrid path. The pure and hybrid SDN paths may also be classified as SDN covered paths, while legacy paths are also classified as SDN uncovered paths. SDN coverage is further classified into two categories: Path coverage (Pcoverage) and Hop coverage (Hcoverage). Pcoverage is coverage of paths of the entire network with at least one SDN-enabled node in the path. Hcoverage indicates the percentage of SDN-enabled nodes in the path, i.e., Hcoverage is the ratio of SDN-enabled nodes to the total number of nodes on a hybrid path. For example, if a path consists of ten nodes of which two are SDN nodes, then the Hcoverage is 20%.

Kar et al. [116] formulated the budgeted maximum coverage problem using maximum coverage and minimum cost for the hybrid SDN network. Two efficient heuristic algorithms are proposed to address the coverage problem: The maximum number of uncovered path first (M UC PF) algorithm for Pcoverage and a maximum number of minimum hop covered path first (MMHcPF) algorithm for Hcoverage. MATLAB evaluations indicate that M UC PF is good in terms of efficiency and economy to deploy a hybrid route between hosts and network devices. To achieve 100% path coverage, M UC PF required 5–15% less cost than other mechanisms. Similarly, MMHcPF required 5–20% less cost to achieve maximum hop coverage.

Guo et al. [117] have studied the traffic engineering problem in OSPF/SDN networks and proposed heuristics based on an incremental (migration) deployment mechanism. More specifically, Guo et al. have evaluated and examined the placement of SDN devices among legacy devices for efficient traffic engineering in an ISP network. The main focus of the migration technique is to find an optimized sequence of routers to minimize the maximum link utilization in an ISP network. A genetic algorithm is used to find the optimized migration sequence from a set of possible sequences. In addition, several heuristic algorithms, including a greedy algorithm and a static migration algorithm, are considered. Incorporating all these algorithms ensures that the genetic algorithm performs better than any of the considered individual algorithms. The performance evaluation concludes that the genetic algorithm can achieve a maximum link utilization lower than the other individual algorithms when 40% SDN nodes are deployed among legacy network devices.

F. Summary and Lessons Learned

We have summarized the comparison of the mechanisms and protocols as well as the objectives and evaluation tools of the existing hybrid SDN network deployment studies in Table II. Table III summarizes the performance evaluations in these hybrid SDN network deployment studies, characterizing the
A. Network Architecture and Design Focused Studies

<table>
<thead>
<tr>
<th>Study</th>
<th>Mechanism</th>
<th>Protocol</th>
<th>Objectives</th>
<th>Evaluation Tool</th>
<th>VLAN</th>
</tr>
</thead>
<tbody>
<tr>
<td>Levin et al. [15]: Seminal</td>
<td>Waypoint enforcement</td>
<td>OSPF or EIGRP, OpenFlow</td>
<td>Integrate legacy devices with SDN contr. via SDN switches and OF</td>
<td>Mininet</td>
<td>Yes</td>
</tr>
<tr>
<td>Panopticon design</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Poularakis et al. [104]:</td>
<td>Greedy based algorithm</td>
<td>OSPF, OpenFlow</td>
<td>Gradual upgrade of OSPF network to SDN</td>
<td>Real-world topologies of ISP</td>
<td>No</td>
</tr>
<tr>
<td>Migration designs</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Xu et al. [105]: Optical</td>
<td>Joint duplicated deployment and routing (DDR)</td>
<td>OSPF or BGP, OpenFlow</td>
<td>Hybrid optical SDN deployment</td>
<td>Mininet</td>
<td>No</td>
</tr>
<tr>
<td>netw.</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Caria et al. [106]: Optical</td>
<td>Sub-domain connected through SDN nodes</td>
<td>OSPF, OpenFlow</td>
<td>Hybrid SDN partitioning for dynamic optical circuits</td>
<td>SNDlib Library</td>
<td>No</td>
</tr>
<tr>
<td>netw.</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Martinez et al. [107]:</td>
<td>Sub-6GHz and mm wave techn. cooperate with microwave links</td>
<td>BDN, GPRS, OpenFlow</td>
<td>Service-based model for hybrid SDN wireless mesh backpack</td>
<td>ns3</td>
<td>No</td>
</tr>
<tr>
<td>Wireless mesh netw.</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>He et al. [108]: Satellite</td>
<td>SDN/IP Hybrid Space</td>
<td>OSPF or RIP, OpenFlow</td>
<td>New SDN/IP network design: overlay of hybrid space inform. netw.</td>
<td>Mininet, POX, Quagga</td>
<td>No</td>
</tr>
<tr>
<td>overlay</td>
<td>Information Network</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Prototype</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

B. Hardware Shim

<table>
<thead>
<tr>
<th>Study</th>
<th>Protocol</th>
<th>Objectives</th>
<th>Evaluation Tool</th>
<th>VLAN</th>
</tr>
</thead>
<tbody>
<tr>
<td>SDN shim [109]</td>
<td>OpenFlow</td>
<td>Provide SDN-like control for legacy devices by installing shim</td>
<td>Realtime practical experiments</td>
<td>Yes</td>
</tr>
</tbody>
</table>

C. ILP Optimization

<table>
<thead>
<tr>
<th>Study</th>
<th>Objectives</th>
<th>Evaluation Tool</th>
<th>VLAN</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tamal et al. [110]: Optimization trajectory</td>
<td>Migration trajectory from legacy dev. by SDN dev.</td>
<td>Java based simulation, SNDlib</td>
<td>No</td>
</tr>
<tr>
<td># of SDN dev.</td>
<td>OSPF, OpenFlow</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Its a Match! [111]: Min.</td>
<td>Deterministic and greedy approx. alg., ILP</td>
<td>Clean slate and incremental deployments</td>
<td>Python based Tool</td>
</tr>
<tr>
<td># of SDN dev.</td>
<td>OpenFlow</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Caria et al. [112], [113]:</td>
<td>Divide entire network into sub-domains</td>
<td>Partition OSPF netw. with SDN switches</td>
<td>SNDlib library</td>
</tr>
<tr>
<td>Partitioned OSPF netw.</td>
<td>OSPF, OpenFlow</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

D. Optimization Heuristics

<table>
<thead>
<tr>
<th>Study</th>
<th>Objectives</th>
<th>Evaluation Tool</th>
<th>VLAN</th>
</tr>
</thead>
<tbody>
<tr>
<td>Hong et al. [114]</td>
<td>Incremental deploym. of SDN switches replacing legacy switches</td>
<td>Real world ISP and Enterprise Network</td>
<td>No</td>
</tr>
<tr>
<td>Heuristic-based approach</td>
<td>OSPF, BGP, OpenFlow</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Xu et al. [115]</td>
<td>Incremental hybrid SDN deployment</td>
<td>Real-time experiment in Monash Univ. netw.</td>
<td>No</td>
</tr>
<tr>
<td>Heuristic Algorithm</td>
<td>OSPF, ECMP, OpenFlow</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Kar et al. [116]</td>
<td>Maximum coverage problem in hybrid SDN</td>
<td>MATLAB, custom development</td>
<td>No</td>
</tr>
<tr>
<td>Max. # of uncovered path first (MUCPF) for path cov., max. # of min. hop covered path first (MMHCPF) for hop cov.</td>
<td>OSPF, OpenFlow</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Guo et al. [117]</td>
<td>Placement of SDN devices for efficient TE</td>
<td>Custom development</td>
<td>No</td>
</tr>
<tr>
<td>Genetic algorithm</td>
<td>OSPF, OpenFlow</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table II

Summary Comparison of Hybrid SDN Network Deployment Studies.

Traffic overhead of the deployment mechanisms and protocols, the link utilization level that can be achieved with the mechanisms, and whether the number of considered SDN flows was fixed or dynamically varying. Moreover, Table III indicates how the deployment of middle boxes, e.g., firewalls and intrusion detection systems, affects the network performance, whether the mechanisms allow for flexible deployment of additional SDN devices, and whether the deployment mechanisms support inter-segment connectivity, i.e., inter-connecting traditional network segments through SDN-based network segments.

Based on our survey, summary comparisons, we draw the following conclusions. Levin et al. [15] have presented the Panopticon technique for SDN switch deployment. The Panopticon technique, which was originally designed to control prison cells from a central location with limited resources, can be easily adopted and is very simple to implement. However, the Panopticon technique does not consider other networking aspects, such as traffic engineering and load balancing of dynamic traffic demands. Future work should expand Panopticon to incorporate these additional functionalities.

Poularakis et al. [104], Xu et al. [105], and He et al. [108] have presented gradual deployment models for hybrid SDN networks. The Poularakis et al. [104] model is applicable to general network scenarios and proceeds via multiple steps towards full SDN network deployment. This incremental deployment in a series of steps towards full SDN network deployment requires initially only a limited number of SDN devices and then gradually increases the number of SDN devices. While the migration model of Poularakis et al. [104] is formulated for general network scenarios, Xu et al. [105] have specifically considered an optical networking scenario and He et al. [108] have specifically considered a satellite overlay networking scenario.

Caria et al. [106] have presented a sub-domain based model for hybrid SDN network deployment. This architecture can be extended to integrate with IETF ABNO [123] and OpenDaylight [124] controllers. Future research could investigate whether additional levels of partitioning inside the individual sub-domains created by Caria et al. so as to create a hierarchy...
of sub-domains would further increase the flexibility, performance, and scalability of the hybrid SDN network. Martinez et al. [107] have presented a service-based model for hybrid SDN wireless mesh backhaul in which multiple networks smoothly cooperate.

The performance comparisons in Table III indicate that each approach presents performance trade-offs that need to be carefully weighed for particular network scenarios. For example, the Martinez et al. [107] approach has several attractive performance characteristics in that it is resilient to path failures and achieves high performance with middle box deployment while supporting dynamic flow installation. However, the Martinez et al. [107] approach incurs high overhead because it does not consider path stretch. On the other hand, the Levin et al. [15] approach strives to minimize path stretch; therefore, it is not resilient to path failures. In order to deploy more SDN devices, the Levin et al. [15] approach requires additional computation to find efficient paths. Similarly, the He et al. [108] approach is resilient to path failures, has low overhead, and achieves high performance with middle box deployment; however, it cannot flexibly accommodate more SDN switches due to the computation overhead for additional devices.

The SDN shim [109] introduces a hardware module that may increase the budget and is inflexible when a variety of legacy devices exist in the network. The SDN Shim may be well suited for small-scale networks consisting of homogeneous legacy devices. Instead of a hardware (shim) module, actual SDN switch deployments may provide a more flexible and cost-effective solution for introducing SDN capabilities in large-scale legacy networks. In particular, for large-scale network deployments, Tamal et al. [110] have presented an Integer Linear Programming (ILP) based approach to find efficient deployments of SDN devices. The Tamal et al. mechanism provides a good solution for incremental deployments of SDN devices. Hong et al. [114] have also introduced an incremental deployment mechanism that uses a heuristic based solution for the SDN switch deployment. Scalability can be achieved by dividing the traffic demands based on the incremental deployment strategy. The influence of growing rates of flow rule updates for routing paths in these incremental deployments should be studied in future research. Among the ILP optimization approaches, the Tamal et al. [110] approach has several attractive performance features, as Table III indicates. The Tamal et al. [110] approach supports a dynamic number of flows, is resilient to path failures, and achieves high performance with middle boxes deployment; however, it incurs high overhead because of complex custom computations for configurations.

Xu et al. [115] and Kar et al. [116] have designed incremental deployments for hybrid SDN networks under specified cost and the maximum coverage constraints. Xu et al. [115] have also optimized the cost in terms of legacy devices that are considered for replacement. The Xu et al. approach finds solutions that keep the old legacy devices in the network (to continue contributing to network transport services). Generally, there is a trade-off between the improved control and management with SDN devices and the capacity of traditional legacy devices that are removed from the network (or are no longer fully utilized) through the replacement of legacy devices with SDN switches. Future research should examine this trade-off in detail and provide economical solutions that allow for effective upgrades of the network control and management through SDN devices while preserving and efficiently utilizing the installed capacity of legacy devices. The Kar et al. [116] approach has generally the best performance characteristics in the optimization heuristics category. The Kar et al. [116] approach has low overhead, accommodates dynamic flow installation, as well as the deployment of additional SDN devices. However, the Kar et al. approach does not consider link utilization in its modeling, thus it can lead to scenarios where some links may have low load. Also, the Kar et al. approach has low performance with middle boxes deployments because it estimates cost to increase throughput and ignores some other factors.
### III. CONTROLLERs FOR HYBRID SDN NETWORKS

The SDN controller is the central “brain” that controls an SDN network. The control plane encompasses all functions and processes that determine which path to use, including routing protocols, e.g., IPv6 [143], spanning tree protocols, e.g., STP [144], interface state management, e.g., Point to Point Protocol (PPP) [145], Link Aggregation Control Protocol (LACP) [146], connectivity management, e.g., Bidirectional Forwarding Detection (BFD) [147], Connectivity Fault Management (CFM) [148], adjacent device discovery (e.g., hello mechanisms in most routing protocols, such as End System-to-Intermediate System (ES-IS) [149]), Address Resolution Protocol (ARP) [150], as well as topology or reachability information exchange protocols (IP/IPv6 routing protocols, IS-IS in TRILL/SPB [151]). This section surveys the existing controllers that have been proposed for controlling both legacy and SDN devices in a hybrid SDN network. We have organized this survey of the hybrid SDN controllers according to the taxonomy illustrated in Fig. 11.

#### A. Virtualization Based Mechanisms

In network virtualization, hardware and software network resources can be combined to form a single virtual network resource object or a single resource can be “sliced” into multiple virtual resources [93], [94], [152]–[158]. This virtualization can be achieved by introducing virtual local area networks (VLANs) [119], [120] in the network. This section surveys the virtualization based techniques for controlling hybrid SDN networks.

**HybNET** [126] is a network control framework for hybrid SDN networks. HybNET centrally controls and manages legacy and SDN-based switches through virtualization [159] across the entire network. HybNET creates a common interface between legacy and SDN network configurations on the one hand, and the central controller on the other hand. HybNET thus effectively hides the distinctions between legacy and SDN network configurations from the SDN controller. For this purpose, a virtual link may span multiple physical links between SDN and legacy switches. In HybNET, SDN switches carry out the main tasks of network control and management with the help of the controller. Legacy switches function only as forwarding devices. The network virtualization functionality is achieved through VLANs [160] in the traditional network devices and through fine-grained forwarding rules installed by the SDN controller in the SDN switches. HybNET has been implemented in OpenStack [161], which is a popular open source cloud computing platform, by using neutron [161] as the network service of OpenStack for host side network virtualization.

**SYMPHONY** [127] is a framework for integrating the control plane at legacy devices with the SDN control plane, effectively forming a coexistence of the distributed legacy and centralized SDN control planes [36]. In SYMPHONY, the...
Telekinesis thus strives to forward the data packet to the SDN switch to enter the network are forwarded to the nearest SDN switch. Forwarding entries are configured so that data packets that update their forwarding entries. In particular, the legacy switch SDN switches to send special packets to legacy switches to update routing in both legacy and SDN switches. Telekinesis instructs a new flow control, called Legacy Flow Mod, to control the SDN controller as soon as possible so that the controller processes the packet and subsequently installs flow entries in the concerned switches (both legacy and SDN switches). A network view of Telekinesis is shown in Figure 13 where an SDN switch is placed at center of the network topology. All routes must pass through the SDN switch. VLAN techniques are used to instruct legacy switches to implement specific routes. Telekinesis covers only the routing control in hybrid SDN networks and does not consider other network conditions, such as access control list (ACL) violations or network loops. The Telekinesis approach has recently been further refined by the Magneto approach [168], which strives to stabilize the routes to the SDN switches.

Cardigan [129] is a central controller for controlling network traffic in a hybrid SDN network. Cardigan is basically a distributed router based on OpenFlow that is deployed at a public internet exchange [169]. The proposed Cardigan approach consists of three parts. The first part designs and implements an SDN-based distributed routing fabric. The second part supports SDN migration through drop-in replacement of network hardware. The third part identifies incompatibilities in production environments, including implementation barriers to wider deployment. The proposed Cardigan distributed router approach was deployed in a live Internet Exchange [129] and successfully passed production traffic. The Cardigan approach includes virtualization through a router abstraction model, which can reduce the operational complexity, especially for large networks.

Fuentes et al. [130] have proposed an SDN control framework for legacy Data Over Cable Service Interface (DOCSIS) based access networks [170]–[173]. The complexity of the underlying DOCSIS access network is hidden by a Hardware Abstraction Layer (HAL) [174], [175]. Fuentes et al. have built on the HAL of the ALIEN project [176] to design and implement a novel HAL to facilitate the integration of legacy network devices and SDN devices. Fig. 14 shows the integrated DOCSIS SDN architecture. The SDN-capable OpenFlow Aggregation Switch in the lower right of Fig. 14 aids the cable modem termination system (CMTS) to provide...
controller and SDN devices. ClosedFlow [131] performs these actions at SDN switches, and 4. Communication between controller, 3. Modification of flow tables by changing entries like control over legacy switches and routers. In pure SDN the developed ClosedFlow control technique provides SDN-ling legacy switches and routers through a central so-called SDN controller. The legacyFlow mode approach controls legacy network devices through the OpenFlow Infrastructure Controller.

**B. Parsing and Configuration Translation**

This section surveys parsing and configuration translation based techniques for the control of hybrid SDN networks. The parsing and configuration techniques translate the information from legacy network devices into a form that is understood by the SDN controller.

Hand et al. [131] have developed a technique for controlling legacy switches and routers through a central so-called ClosedFlow controller, that is similar to an SDN controller. The developed ClosedFlow control technique provides SDN-like control over legacy switches and routers. In pure SDN networks, four major tasks are performed: 1. Establishment of a control channel between the controller and SDN devices, 2. Knowledge of topology for network-wide view at the controller, 3. Modification of flow tables by changing entries and actions at SDN switches, and 4. Communication between controller and SDN devices. ClosedFlow [131] performs these tasks to control and manage legacy devices in a hybrid SDN network as follows:

1) Controller-switch control channel: a minimum routing instance of OSPF [178] is implemented to create a channel between controller and switch. This includes advertisements for loopback management interfaces of the switch, point-to-point connections between switches, and a VLAN for controller communication.

2) Topology Discovery: The controller discovers the topology through remote logging at each legacy switch. This allows the controller to store the topology state and to learn about link failures.

3) Packet matching and applying actions are achieved through a combination of access control lists (ACL) [179], Route Maps, and interface configurations of legacy devices.

4) Handling Packet-In Event: This event is handled in two ways: (a) remote logging on explicit deny [180], and (b) send the entire packet to the controller.

The proposed approach does not support other network activities, such as load balancing [164]–[167] or loop detections. Also, each type of switch requires a specific corresponding ClosedFlow mechanism.

Exodus [132] controls legacy network devices by obtaining the device configurations, compiling the device configurations into an intermediate format, and, finally, producing the equivalent SDN rules. Exodus facilitates the migration from traditional networking to SDN by translating legacy network configurations, e.g., the Cisco IOS [181], to the corresponding SDN configuration. Exodus obtains the IOS configurations (from possibly different legacy routers) and passes the configurations to the Exodus IOS Parser and Compiler. The Exodus IOS Parser and Compiler then generate the network specification and flowlog libraries that can be used as a prototype for SDN rules. In this way, legacy devices can be made to perform as SDN devices. However, the Exodus study [132] is limited to translating configurations from legacy devices to SDN control information and vice versa; and does not address more complex network control functionalities. A main limitation of the Exodus approach is that a separate specific parsing and compiling code has to be written for each type of specific network device from a specific vendors in order to translate configurations to equivalent SDN rules.

Farias et al. [133] have addressed the problem of two or more SDN networks communicating with each other through intermediate legacy network devices. The legacy network device configurations need to be translated to SDN configurations so that the SDN controller can understand the configurations and vice versa. To solve this problem, Farias et al. [133] have proposed a legacyFlow mode approach. The legacyFlow mode approach controls legacy network devices through the SDN controller. The legacyFlow mode approach translates the OpenFlow rules into vendor-specific configurations of legacy switches. The main limitation of the legacyFlow mode approach is that there are several legacy device manufacturers, for every device series a specific set of rules needs to be translated using the legacyFlow mode.

Figure 14. Illustration of SDN enabled DOCSIS cable network architecture proposed by Fuentes et al. [130]: The details of the DOCSIS cable system are abstracted by a Hardware Abstraction Layer, implemented through the ALIen Hardware INtegration Proxy (ALINP). The entire cable access network is effectively virtualized to one SDN switch for the OpenFlow Infrastructure Controller.
C. Controllers for Cloud and Data Center Networks

Cloud and data center networks are important for supporting a wide range of networked user applications. The SDN paradigm can simplify the management and control of these networks [182]–[185]. This section surveys the control mechanisms that have been specifically developed for hybrid SDN networks in the context of clouds and data centers.

Meridian [134] is an SDN-based architecture for the cloud networking. Meridian is composed of three logical layers: (i) Network model and APIs, (ii) Network orchestration, and (iii) Interfaces to underlying network devices. The network model and API layer provide a declarative and query based API to the cloud controller for the specification of access control policies, prioritizing traffic, and traversing middleboxes. The network orchestration layer links logical network entities with the physical network entities and implements all abstractions by mapping logical operational commands onto the underlying physical network. The interface layer to the underlying network devices layer contains the network layer driver for interacting with different networking technologies, such as SDN switches and legacy switches. The proposed architecture can be leveraged by Open-Stack and IBM smart cloud provisioning controllers.

Cerroni et al. [135] have proposed an SDN-based centralized control architecture for mixed packet-switched and circuit-switched networks cloud and data center networks. The architecture implements SDN in the network layer by inserting an OpenFlow Agent (OA), a Logical Forwarding Element (LFE), and a Physical Forwarding Element (PFE) into each node. The OpenFlow agent implements the interface to the controller. The LFE implements specific hardware independent forwarding. The PFE is implemented for a specific hardware technology and performs hardware-specific forwarding. The controller is extended to handle the underlying proposed network nodes through extensions of the OpenFlow protocol. Experimental result have demonstrated the feasibility of the proposed architecture.

Baik et al. [136] have proposed a hybrid SDN controller based architecture for end-to-end path provisioning in mixed packet-switched and circuit-switched networks. The main components of the proposed architecture are a service manager, a hybrid SDN controller, and a cloud manager [186]. The service manager interacts with users to provide information about the available services. The cloud manager manages the resources available in the data center, e.g., creates virtual machines and sets up paths. The hybrid SDN controller is responsible for the provisioning of the end-to-end path that encompasses path setup in the underlying packet-switched access network and the circuit-switched backbone network. A user accesses the services of the cloud as follows:

1) The user generates the path setup request by accessing the required service.
2) After receiving the path request from the user, the service manager forwards the request to the SDN controller.
3) The service manager forwards the request to the cloud manager for the allocation of the required resources.
4) The cloud manager allocates the required resources (virtual machines) and sets up the path for the border node.
5) The hybrid SDN controller establishes the end-to-end path that encompasses the path setup in the underlying packet-switched access network and the circuit-switched backbone network.
6) The service manager combines the path establishment information from the controller and the cloud manager, and provides a complete service communication path to the user.

The Routing Control Platform (RCP) [137] is an IP routing architecture for establishing a logically centralized control plane isolated from legacy routers. This control plane can be used to evaluate and augment the decision logic of the Border Gateway Protocol (BGP) routing protocol. The BGP routing protocol [187], [188] is commonly used for inter-domain routing between Autonomous System (ASs), which correspond to specific network domains, whereby each AS is under a specific administrative control. BGP is similar to the Routing Information Protocol (RIP) in that BGP learns routes from different neighbors, so-called BGP peers, and then under consideration of the newly learned route decides what route toward a particular destination is the best route. For each known destination, BGP then sends this single best route to its peers. RCP allows an individual AS to easily deploy a new customer service. RCP employs SDN for managing the centralized control plane and thus forms a hybrid SDN network. More specifically, RCP builds on RouteFlow [189] to compute BGP routes with the help of the SDN control plane. The RouteFlow Control Platform (RFCP) [137] acts as an indirection layer for control messages that carry routing and forwarding information from BGP routers. The RFCP translates the routing and forwarding information to OpenFlow rules. A limitation of the RCP study [137] is that it only considers BGP for analysis of the entire network. Other protocols would also need to be considered, and their effect on network performance needs to be evaluated.

Dey et al. [138] have examined the benefits and challenges of service-based cloud infrastructures and presented a hybrid SDN routing architecture for cloud system management. According to studies on cloud platforms [14], [190], the strict separation of the control and data planes does not necessarily deliver the best solution for flexible and scalable networking. Dey et al. [138] have provided mechanisms for the adjustment of data plane functionalities in cloud platforms, including failure management and loop-free routing. In the considered hybrid SDN architecture, most data plane operations are performed by traditional local routers, with partial offloading to the SDN controller. The hybridization of the control plane means that most shortest path calculations are performed at the SDN controller and some portions of these calculations are performed at the traditional routers. Similarly, hybridization of the data plane means that some of the packet forwarding task are coordinated by the SDN controller, while the remaining packet forwarding tasks are performed by the local routers. This type of hybridization may be highly effective as cloud computing platforms become more common.

Large-scale hybrid SDN cloud ISP networks require so-
controllers discover the legacy network devices via SDN de-
manship for data centers. Lin et al. [140] propose to let the
D. Miscellaneous
Mininet emulator as well as on distributed SDN testbeds.
scenarios. Mantoo is a group of management tools to install
such as cloud and data center networks, Mantoo a group
framework and Mininet for large-scale hybrid SDN networks,
for small-scale experiments. To deploy and test the OSHI
and hybrid SDN networks. However, Mininet is only suitable
community to evaluate the performance of SDN mechanisms
of virtual OFCS ports.

the OFCS, while the IP forwarding engine is linked to a set
kernel IP networking functions and Quagga [122], as illus-
(OVS) [161] and an IP forwarding engine based on the Linux
an IP routing daemon. The OFCS consists of an OpenvSwitch
Flow Capable Switch (OFCS), an IP forwarding engine, and
an IP routing daemon. The OFCS is connected to the set of physical network interfaces of the integrated
OFCS is connected to the set of physical network interfaces of the integrated
IP/SDN network; the IP forwarding engine is connected to the virtual internal
OFCS ports.

Figure 15. The Open Source Hybrid IP/SDN (OSHI) node within the Open
Network Operating System (ONOS) [139] consists of an OpenFlow Capable
Switch (OFCS), an IP forwarding engine, and an IP routing daemon. The
OFCS is connected to the set of physical network interfaces of the integrated
IP/SDN network; the IP forwarding engine is connected to the virtual internal
OFCS ports.

lutions for: (a) providing scalability and fault tolerance for the operator, (b) managing the latency incurred by the Open-
Flow protocol for communication with the controller [191]–
[193], and (c) connecting WAN switches [194] with the
SDN controller. In particular, the evaluation of large-scale
cloud networks with thousands of nodes and links requires a
realistic evaluation platform. For this purpose, an Open Source
reference node implementation and Open Source emulation
framework have been established in the context of the Open
Network Operating System (ONOS) [139]. Specifically, the
Open Source Hybrid IP/SDN (OSHI) node contains an Open-
Flow Capable Switch (OFCS), an IP forwarding engine, and
an IP routing daemon. The OFCS consists of an OpenSwitch
(OVS) [161] and an IP forwarding engine based on the Linux
kernel IP networking functions and Quagga [122], as illustrat-
ed in Fig. 15. Physical network interfaces are connected to the
OFCS, while the IP forwarding engine is linked to a set
of virtual OFCS ports.

Mininet [195] has been widely used by the SDN research
community to evaluate the performance of SDN mechanisms
and hybrid SDN networks. However, Mininet is only suitable
for small-scale experiments. To deploy and test the OSHI
framework and Mininet for large-scale hybrid SDN networks,
such as cloud and data center networks, Mantoo a group
of management tools can be used. Mantoo [196]–[198] can
provide a suitable environment for realistic large-scale network
scenarios. Mantoo is a group of management tools to install
and check the OSHI networking framework and services on a
Mininet emulator as well as on distributed SDN testbeds.

D. Miscellaneous
Lin et al. [140] have proposed a QoS-aware routing mecha-

nism for data centers. Lin et al. [140] propose to let the
controller discover the legacy network devices via SDN de-
vice through a spanning tree protocol (STP). For coordina-
tion among the legacy and SDN devices, a Learning Bridge
Protocol (LBP) is used that eliminates the modifications on
legacy devices that are ordinarily needed to coordinate with
SDN devices. The SDN controller finds routes dynamically
based on QoS requirements for the current network topology.
Lin et al. [140] then propose a simulated annealing based
QoS-aware routing (SAQR) algorithm. For a prescribed QoS
requirement, the SAQR algorithm minimizes the delay and loss
rate as well as adjusts the bandwidth requirements for a best-
fitting route. The performance evaluations indicate that SAQR
performs better than MINA, which is an SDN architecture
for the Internet-of-Things [199], in terms of fitness ratio of
delay, loss rate, and bandwidth with significant increases in the
numbers of flows meeting their respective QoS requirements.

Huang et al. [141] have examined the integration of
SDN and traditional network paradigms against the back-
drop that most of today’s SDN control planes cannot handle
legacy devices. To address this shortcoming of SDN control
planes, Huang et al. [141] have developed HybridFlow as
a lightweight control pane for hybrid SDN networks. In
HybridFlow, a hybrid SDN network is abstracted to a logical
SDN network in a lightweight manner, i.e., the abstraction
only updates the network control messages between infrastruc-
tures and applications. By using this HybridFlow abstraction,
traditional control applications and programs can operate the
logical SDN network as an actual SDN network. HybridFlow
has two main responsibilities: Port allocation and maintenance
of the logical SDN network. Port allocation maps the logical
ports in the logical SDN network to the physical ports so
that control application can interact with the physical ports.
The maintenance of the logical SDN network generates rules
and transforms messages. HybridFlow has been implemented
on the POX controller [200], [201]. Experimental results
indicate that HybridFlow works efficiently and introduces only
a marginal overhead.

Sun et al. [142] have introduced a mathematical model
for an SDN controller that implements routing control so as to
optimize the load balancing in a hybrid SDN network. In
the considered hybrid SDN network, legacy and SDN
network devices coexists to form the overall network. Only
SDN devices are controlled by the SDN controller, while
legacy devices use the traditional OSPF shortest path routing
protocol. Sun et al. have proposed a new routing algorithm,
namely Dijkstra-Repeat that finds disjoint multipath routes.
The SDN controller maps multiple flows that are destined
to the same node onto disjoint paths; the flows are then
aggregated upon reaching the destination node. Then, a Lazy
Route Update (LRU) load balancing technique is designed
based on a mathematical model for the SDN controller’s load
balancing optimization. The LRU can improve the load bal-
cing by spreading network traffic across the entire network.
Comparisons of the proposed LRU approach with other load
balancing algorithms indicate that the proposed LRU approach
improves the efficient link utilization.

E. Summary and Lessons Learned
Based on the summary comparison of the hybrid SDN
network control approaches in Table III-D, we draw the
### Table IV

**Summary Comparison of Hybrid SDN Controller Studies.**

<table>
<thead>
<tr>
<th>Study</th>
<th>Mechanism</th>
<th>Protocol</th>
<th>Objectives</th>
<th>Evaluation Tool</th>
<th>VLAN</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>A. Virtualization Based Mechanisms</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>HybNET [126]: VLAN based</td>
<td>Virtualization</td>
<td>OpenFlow</td>
<td>Central control and virtualization of entire network</td>
<td>Custom module on OpenStack</td>
<td>Yes</td>
</tr>
<tr>
<td>SYMPHONY [127]: VLAN based</td>
<td>Legacy Route Server (LRS) for legacy-SDN device commun.</td>
<td>OSPF, OpenFlow</td>
<td>Control and interdomain commun. between legacy and SDN domains</td>
<td>Mininet, miniNExT, POX</td>
<td>Yes</td>
</tr>
<tr>
<td>Shear [16]: VLAN based</td>
<td>Hybrid control and data plane</td>
<td>OpenFlow, STP, OSPF</td>
<td>Arch. for interconnecting SDN and legacy switches</td>
<td>Ryu controller</td>
<td>Yes</td>
</tr>
<tr>
<td>Telekinesis [128]: VLAN based</td>
<td>LegacyFlowMod</td>
<td>OpenFlow</td>
<td>A fine-grained routing control using OpenFlow</td>
<td>Mininet</td>
<td>Yes</td>
</tr>
<tr>
<td>Cardigan et al. [129]: Distributed routing fabric</td>
<td>Extensions to RouteFlow for real-time network connectivity</td>
<td>BGP, OpenFlow</td>
<td>SDN-based distributed routing fabric</td>
<td>Public Internet Exchange</td>
<td>Yes</td>
</tr>
<tr>
<td>Data Over Cable Service Interface (DOCSIS) [130]: Cable access netw.</td>
<td>Hardware Abstractions Layer (HAL)</td>
<td>OpenFlow</td>
<td>Integration of legacy network with OpenFlow control framework</td>
<td>DOCSIS access network and OF aggregation switch</td>
<td>Yes</td>
</tr>
<tr>
<td><strong>B. Parsing and Configuration Translation</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Hand et al. [131]: ClosedFlow controller</td>
<td>Remote logging and VLAN configuration</td>
<td>OSPF, without OpenFlow</td>
<td>OpenFlow-like control over legacy devices</td>
<td>Custom application with real-time network</td>
<td>Yes</td>
</tr>
<tr>
<td>Exodus [132]: Translate leg. dev. config. to SDN</td>
<td>Parsing and translation of configurations</td>
<td>OSPF, OpenFlow</td>
<td>Automatic migration legacy configurations to OpenFlow</td>
<td>Mininet</td>
<td>Yes</td>
</tr>
<tr>
<td>Farias et al. [133]: Commun. between multiple SDN nets</td>
<td>Provide legacy flow data path for legacy devices in SDN network</td>
<td>HTTP-JSON, OSPF</td>
<td>Provides a legacy data path for translating the OpenFlow rules</td>
<td>GENI network</td>
<td>Yes</td>
</tr>
<tr>
<td><strong>C. Controllers for Cloud and Data Center Networks</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Meridian [134]: Interfaces + orchestration</td>
<td>(i) Netw. model, APIs, (ii) Netw. orchestration, (iii) Interfaces to underlying netw. dev.</td>
<td>OpenFlow</td>
<td>Common service model and programming interface for multiple cloud controllers</td>
<td>Floodlight controller platform</td>
<td>Yes</td>
</tr>
<tr>
<td>Cerroni et al. [135]: Logical and phy. path forw. elem.</td>
<td>OpenFlow Agent (OA), Logical Forw. Elem. (LFE) and Physical Forw. Elem. (PFE)</td>
<td>OpenFlow</td>
<td>SDN-based centralized control arch. for mixed packet-sw. and circuit-sw. networks</td>
<td>Virtual test bed</td>
<td>No</td>
</tr>
<tr>
<td>Baik et al. [136]: Hybrid SDN cloud manager</td>
<td>Service Manager, Hybrid SDN Controller and Cloud Manager</td>
<td>OpenFlow</td>
<td>End-to-end path prov. in mixed packet-sw. and circuit-sw. netw.</td>
<td>Cloud network</td>
<td>No</td>
</tr>
<tr>
<td>Routing Control Platform (RCP) [137]: BGP based SDN routing</td>
<td>RouteFlow Control Platform (RFCP) as an indirection layer</td>
<td>BGP, MPLS, OpenFlow</td>
<td>Datastore-centric platform design for advanced routing</td>
<td>RF-Server modules, NOX, JSON, Quagga</td>
<td>No</td>
</tr>
<tr>
<td>Dey et al. [138]: Hybrid control and data planes</td>
<td>Partial offloading of control and packet fwd. from leg. dev. to SDN</td>
<td>BGP, OpenFlow</td>
<td>Hybrid SDN routing arch. for cloud system</td>
<td>Use-case based study</td>
<td>No</td>
</tr>
<tr>
<td>ONOS Open Source Hybrid IP/SDN (OSHI) node [139]</td>
<td>Local Management Entity (LME), OpenFlow Capable Switch (OFCS)</td>
<td>OSPF, OpenFlow</td>
<td>A group of management tools for Hybrid IP/SDN (OSHI) networks</td>
<td>Mininet</td>
<td>Yes</td>
</tr>
<tr>
<td><strong>D. Miscellaneous</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Lin et al. [140]: Learning Bridge Protocol (LBP)</td>
<td>Simulated annealing based QoS-aware routing (SAQR) algorithm</td>
<td>BGP, OpenFlow</td>
<td>QoS in hybrid SDN, QoS-aware routing for data centers</td>
<td>Floodlight controller and Mininet</td>
<td>Yes</td>
</tr>
<tr>
<td>Huang et al. [141]: HybridFlow</td>
<td>Lightweight control plane for hybrid SDN</td>
<td>Traditional and OpenFlow</td>
<td>Integration of SDN and trad. control planes</td>
<td>POX controller and Mininet</td>
<td>No</td>
</tr>
<tr>
<td>Sun et al. [142]: Dijkstra-Repeat disjoint path + lazy route updates</td>
<td>New Dijkstra-Repeat routing algorithm in SDN nodes</td>
<td>OSPF, OpenFlow</td>
<td>SDN controller optim. for hybrid SDN load balancing</td>
<td>Custom development</td>
<td>No</td>
</tr>
</tbody>
</table>

Following conclusions. HybNET [126], Symphony [127], and SHEAR [16] are controller mechanisms that establish connectivity across an entire network based on VLANs. These control mechanisms simplify the network management and control by hiding the underlying network configuration from the users of the legacy and SDN devices. While these VLAN based control approaches provide relatively simply control functionality, they have a number of shortcomings that should be addressed in future research. HybNET [126] does not support automatic detection and configuration of network devices. SHEAR [16] employs a simple spanning tree protocol to prevent loops; however, the simple spanning tree protocol may not work well for dense networks. Telekinesis [128] is also a VLAN based network controller, with a focus on efficient route management in a hybrid SDN network. Telekinesis introduces a new flow routing control, namely Legacy Flow Mod, to control the routing in both legacy and SDN switches. Legacy Flow Mod tends to take a long time to find routes because of demanding processing steps in the route computations. Future research should enhance the route computation in Telekinesis in order...
to minimize computation latencies, while keeping routing efficiency high.

Cardigan [129] is basically a distributed router based on OpenFlow that is deployed at a public Internet exchange. The Cardigan study [129] does not consider the monitoring of the network resource usage, the balancing of traffic loads, nor complex setups of distributed routers in non-mesh environments. Future research should address these open issues. Furthermore, future research should investigate and implement new types of routing policies and address the virtualization of network resources in the Cardigan context.

The SDN control framework for legacy Data Over Cable Service Interface (DOCSIS) based access networks in [130] uses a Hardware Abstraction Layer (HAL) to manage the underlying network complexity. The overhead incurred by the HAL and other helper components in the proposed architecture has not been evaluated. Future research on access networks should examine in detail the quality of service of the broadband access services provided via DOCSIS with virtualization as well as the corresponding overhead. In addition, future work can expand the virtualization approach to other access network types, e.g., digital subscriber line (DSL) based and wireless based access networks.

The performance comparison of virtualization based approaches in Table III-D indicates that Telekinesis [128] is a highly efficient approach with attractive features, including low overhead, dynamic SDN flows, and high performance with middle box deployment. However, Telekinesis has low link utilization because it does not use any efficient link optimization technique. Symphony [127] and Cardigan [129] have also some attractive features, such as low overhead. However, they also have respective weaknesses, e.g., low performance with middle boxes for Cardigan and lack of resilience to path failures for Symphony. SHEAR [16] has been considered for a fixed number of flows, mainly because of limited TCAM capacity and because rules are pre-configured at the time of flow establishment.

Hand et al. [131] have developed a ClosedFlow control approach that translates the major tasks of SDN control to specific tasks that can be executed by specific types of legacy switches. Some problems, such as network loops, violation of network policies, and misconfigurations, may occur during these translation processes. Exodus [132] is a more flexible automatic migration mechanism that is suitable for multiple types of switches. Exodus [132] perform efficient route management through simple configuration translations between network devices and the controller. The path enforcement mechanisms in these hybrid SDN network controllers need to be investigated further in order to minimize latency and improve efficiency.

Farias et al. [133] have introduced the LegacyFlow mode approach for the translation of OpenFlow rules. Generally, parsing and configuration based approaches, such as the LegacyFlow mode approach [133], are more accurate and scale better with increasing network size than virtualization based approaches. This is because, the parsing and configuration based approaches retain most of the functionality of distributed routing protocols and allow new devices to be added easily.

Among the parsing and configuration translation approaches in Table III-D, the Farias et al. [133] approach has attractive features, including low overhead, accommodation of dynamic SDN flows installation, and resilience to path failures; however, it has low link utilization and low performance with middle box deployments due to an excess of rules installed on the devices. The Farias et al. approach can be used for large network scenarios and can flexibly coexist with multiple network segments, albeit its performance is low for inter-connected segments. The Hand et al. [131] approach is implemented over legacy switches to achieve SDN like control so it does not support real SDN features.

<table>
<thead>
<tr>
<th>Study</th>
<th>Traffic Over.</th>
<th>Link Util.</th>
<th>Number of SDN Flows</th>
<th>Path Failure</th>
<th>Middle Box Deployment</th>
<th>Inter-Segment Connectivity</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>A. Virtualization Based Mechanisms</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>HybNET [126]</td>
<td>Low</td>
<td>High</td>
<td>Fixed</td>
<td>Not Resilient</td>
<td>Low Performance</td>
<td>Not Considered</td>
</tr>
<tr>
<td>SYMPHONY [127]</td>
<td>Low</td>
<td>Low</td>
<td>Dynamic</td>
<td>Not Resilient</td>
<td>High Performance</td>
<td>Not Considered</td>
</tr>
<tr>
<td>Shear [16]</td>
<td>High</td>
<td>Low</td>
<td>Fixed</td>
<td>Resilient</td>
<td>High Performance</td>
<td>Considered</td>
</tr>
<tr>
<td>Telekinesis [128]</td>
<td>Low</td>
<td>Low</td>
<td>Dynamic</td>
<td>Resilient</td>
<td>High Performance</td>
<td>Considered</td>
</tr>
<tr>
<td>Cardigan et al. [129]</td>
<td>Low</td>
<td>High</td>
<td>Fixed</td>
<td>Resilient</td>
<td>Low Performance</td>
<td>Not Considered</td>
</tr>
<tr>
<td>DOCSIS [130]: Cable access net.</td>
<td>High</td>
<td>High</td>
<td>Fixed</td>
<td>Not Resilient</td>
<td>Low Performance</td>
<td>Not Considered</td>
</tr>
<tr>
<td><strong>B. Parsing and Configuration Translation</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Hand et al. [131]</td>
<td>High</td>
<td>High</td>
<td>Fixed</td>
<td>Not Resilient</td>
<td>Low Performance</td>
<td>Not Considered</td>
</tr>
<tr>
<td>Exodus [132]</td>
<td>Low</td>
<td>Low</td>
<td>Fixed</td>
<td>Not Resilient</td>
<td>Low Performance</td>
<td>Considered</td>
</tr>
<tr>
<td>Farias et al. [133]</td>
<td>Low</td>
<td>Low</td>
<td>Dynamic</td>
<td>Resilient</td>
<td>Low Performance</td>
<td>Considered</td>
</tr>
<tr>
<td><strong>C. Controllers for Cloud and Data Center Networks</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Meridian [134]</td>
<td>Low</td>
<td>High</td>
<td>Fixed</td>
<td>Resilient</td>
<td>High Performance</td>
<td>Not Considered</td>
</tr>
<tr>
<td>Cerroni et al. [135]</td>
<td>High</td>
<td>Low</td>
<td>Fixed</td>
<td>Not Resilient</td>
<td>Low Performance</td>
<td>Considered</td>
</tr>
<tr>
<td>Baik et al. [136]</td>
<td>Low</td>
<td>Low</td>
<td>Dynamic</td>
<td>Resilient</td>
<td>High Performance</td>
<td>Considered</td>
</tr>
<tr>
<td>Routing Control Platform (RCP) [137]</td>
<td>Low</td>
<td>High</td>
<td>Fixed</td>
<td>Not Resilient</td>
<td>Low Performance</td>
<td>Not Considered</td>
</tr>
<tr>
<td>Dey et al. [138]</td>
<td>High</td>
<td>High</td>
<td>Dynamic</td>
<td>Resilient</td>
<td>High Performance</td>
<td>Considered</td>
</tr>
<tr>
<td>OSHI node [139]</td>
<td>Low</td>
<td>High</td>
<td>Fixed</td>
<td>Not Resilient</td>
<td>Low Performance</td>
<td>Considered</td>
</tr>
<tr>
<td><strong>D. Miscellaneous</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Lin et al. [140]</td>
<td>High</td>
<td>High</td>
<td>Fixed</td>
<td>Not Resilient</td>
<td>High Performance</td>
<td>Considered</td>
</tr>
<tr>
<td>Huang et al. [141]</td>
<td>Low</td>
<td>Low</td>
<td>Fixed</td>
<td>Not Resilient</td>
<td>High Performance</td>
<td>Not Considered</td>
</tr>
<tr>
<td>Sun et al. [142]</td>
<td>Low</td>
<td>Low</td>
<td>Dynamic</td>
<td>Resilient</td>
<td>Low Performance</td>
<td>Considered</td>
</tr>
</tbody>
</table>
Meridian [134] may be scaled to configure performance sensitive cloud workloads, such as Hadoop [202] and content streaming, where scalability is important to accommodate large numbers of tenant network requests. The Meridian study [134] leaves several aspects open for future research, such as automation, security issues, and path enforcement. For clouds and data centers respectively, the approach by Baik et al. [136] and the Routing Control Platform (RCP) approach [137] have been proposed. The Routing Control Platform (RCP) [136] is an IP routing architecture for establishing a logically centralized control plane to evaluate and augment the decision logic of the Border Gateway Protocol (BGP). The RCP study is limited to BGP for analysis of the entire network, other routing protocols can be considered in future research. In particular, advances in the OSPF and IS-IS routing protocols [203] promise advantages for easing the congestion management and for unlocking protocol optimizations, e.g., IP Fast Reroute and protection against link flooding attacks (LFAs) [204]. Dey et al. [138] have proposed service-based hybrid cloud infrastructures where some operations are handled at the data plane and others are handled at the controller. The other studies on cloud and data center network controllers have introduced tools to control the hybrid SDN network, such as Mantoo [196], [198] and the tools by Cerroni et al. [135]. These tools provide efficient control and management tasks in hybrid SDN networks.

Among the controller for cloud and data center networks category, the Baik et al. [136] approach has many attractive features, including low traffic overhead, accommodation of dynamic number of SDN flows, and resilience to path failures; however, the link utilization that is quite low. The Dey et al. [138] approach also has similarly good performance characteristics; however, its traffic overhead is very high because of its path failure detection mechanism.

Lin et al. [140] have proposed a Quality of Service (QoS) routing technique for hybrid SDN networks. More specifically, the Learning Bridge Protocol (LBP) has been proposed for coordination among legacy and SDN devices, eliminating the need for alterations on legacy switches. This mechanism normally considers L2 (link layer) switches, while L3 (network layer) switches and routers may also be considered. Huang et al. [141] have integrated the SDN and traditional network paradigms through their HybridFlow cooperation. In HybridFlow, a network abstraction is used to control the network. In dynamically changing topologies, it is very tedious to update these network abstractions and to manage a network based on the abstractions. Sun et al. [142] introduced the SDN controllers optimization problem for load balancing in hybrid SDN networks as a mathematical optimization model. Sun et al. [142] have proposed a new routing algorithm, namely Dijkstra-Repeat, for SDN nodes to find disjoint multipath routes. However, Dijkstra-Repeat may not be appropriate for complex traffic scenarios, as it is computationally expensive.

**IV. NETWORK MANAGEMENT**

Network management is a vital task concerned with several types of utilities, applications, and protocols to smoothly perform network operations. Network management encompasses the configuration of the operation of the network devices and their performance monitoring. All network interfaces and IP subnets are commonly configured through command line interfaces (CLIs) in conventional networks or the northbound RESTful API in SDN networks. Putzolu et al. [205] and Cor-dray et al. [206] are some examples of network management systems. A variety of approaches have been proposed for network management in pure SDN networks, see e.g., [50], [207]–[209]. This section surveys the existing network management approaches for hybrid SDN networks. The survey follows the classification in Figure 16.

**A. Resource Management**

Resource management is the process of allocation and reallocation of network resources, such as bandwidth and buffers, among the network users and devices [230]–[232]. Resource management is a vital task that has a profound impact on network performance. This section surveys the existing resource management approaches for hybrid SDN networks.

Through hybrid SDN techniques, Santos et al. [210] have attempted to address the problem of resource management in a network that consists of both infrastructure-based and infrastructure-less network components. The controller in a hybrid SDN network has complete knowledge of all the network devices and hosts. Thus, it is easy for the network administrator to control the network bandwidth through SDN switches. Fig. 17 illustrates an example of efficient resource or capacity management using SDN technology. In the scenario illustrated in Fig. 17, Bob and Charlie can communicate with the SDN-based Access Point (AP); however, Alice is too far away from the AP. Alice can communicate through Bob or Charlie, i.e., Bob and Charlie can act as intermediate gateways for Alice, thus enabling Alice to communicate with the Internet at large. If we solve this scenario with a traditional network, then Bob will face the problem of appropriate network bandwidth allocation and Bob’s Internet Service Provider (ISP) does not know that Alice is connected to the network through Bob. Also, the ISP cannot differentiate the traffic of Alice and Bob, and thus no extra bandwidth will be assigned to Bob. Hence, Bob may suffer from network bandwidth shortages. Also, there is a security issue in that Bob can view all the traffic of Alice as it is passing through Bob.

Santos et al. [210] solve this problem with SDN APs and an SDN controller as follows. Whenever Alice joins the network through Bob, the SDN controller learns that Alice is connected through Bob. Then, the SDN controller decides whether Alice can use the network. If yes, then the SDN controller allocates extra bandwidth to Bob to accommodate the traffic load of Alice. The network also applies a security policy for Alice (which can be achieved through a user-defined application running on the SDN controller [233], [234]). Moreover, the SDN controller can apply quality-of-service (QoS) policies by creating a virtual port at the SDN AP and SDN switches according to the users. For instance, Weighted Fair Queuing [235] can be used at each port for QoS provisioning. Thus, every user gets an appropriate bandwidth
The VMF architecture is composed of three main components, namely the Filtering and Common Processing (FCP) Module, the Transformation and Adaptation (TA) module, the Basic Common Monitoring (BCM) module, and the User Defined Monitoring (UM) module. The FCP module is responsible for collection of network events, event mitigation function, packet and flow filtering, time stamping, anomaly traffic detection, host detection, and other related functions. The TA module provides communication between remote administration units of network systems and controllers by supporting the SDN-based OpenFlow protocol as well as the traditional Simple Network Management Protocol (SNMP) [237]. The OpenFlow Statistics collections Proxy (OSP) collects statistics from the SDN switches and provides the statistics to the SDN controller. The Detect and Mitigate Abnormality (DMA) component detects abnormal behaviors in the network. The proposed middlebox architecture provides integrated network management services for the hybrid SDN network and reduces the load on the SDN controller.

Kato et al. [212] have proposed a management mechanism for resource usage consolidation and power consumption minimization in enterprise networks. The management mechanism relies on SDN-controlled network reconfigurations. In the proposed approach, network elements that are not used for a prescribed time duration are turned off. The proposed approach involves the identification of time intervals that include low traffic requirements, the criticality and utilization of network elements, and the switching off of appropriate nodes and links. The proposed architecture is based on a modular implementation of various functions of the network infrastructure in the form of software applications. This modular approach is very flexible. The results show that a 47% decrease in link traffic can be achieved in an evaluation scenario through the resource consolidation, while the power consumption minimization saves up to 45% of the consumed energy.

Sieber et al. [213] have examined the network control and management issues arising when legacy and SDN devices form a hybrid SDN network. To manage these diverse types of devices, Sieber et al. have proposed a unified data plane as well as control plane abstractions in form of a Network Services Abstraction Layer (NSAL) on top of the network.
management and control plane. In addition, Sieber et al. have introduced a unified data model for SDN and legacy devices that configures both network devices in a unified manner to achieve Quality of Service (QoS) for time critical applications, such as VoIP. With the unified data model, operators can easily configure both types of network devices using a single interface. A demonstration of QoS management has been presented in [213], showcasing how multiple networks can benefit from the unifying Network Service Abstraction Layer in combination with per-device QoS abstractions.

In the study [214], Sieber et al. have continued their investigation of device-neutral and vendor-neutral programmability of hybrid SDN networks. The legacy and SDN devices require specific configurations that vary from device to device. The SDN management plane can typically not directly configure all types of devices. Within the context of a Panopticron [15], [17] testbed, Sieber et al. [214] have investigated resource management applications that discover and configure the network for QoS. The SDN controller is configured to perform various functions, such as monitoring of network devices and preventing undesired traffic interruptions. Simulation results indicate that with the examined resource management, Panopticron can achieve good QoS.

B. Auto-Configuration

In a traditional network, network devices and network polices are configured manually. SDN provides network abstraction and auto configuration of network devices and policies. In a hybrid SDN network, the configuration task can be challenging as legacy devices continue to require manual configuration. Also, the configuration of the SDN part and the traditional part of the network need to be coordinated. This section surveys auto-configuration mechanisms to facilitate the configuration of hybrid SDN networks.

Katay et al. [215] have proposed an auto configuration technique for SDN switches in a hybrid SDN network. The proposed technique uses three main components for auto configuration of a new SDN switch 1) Locator, 2) Configurator, and 3) DHCP-SDN (an extended version of the traditional dynamic host configuration protocol (DHCP) [238]). The locator provides the location information of a new switch. The configurator enables reachability between the new switch and available SDN controllers. The DHCP-SDN configures the switch, by providing information about the controllers and seamless service across the SDN part of network and the traditional part of the network. The proposed mechanism has been tested in a hybrid SDN environment and has successfully configured a newly deployed SDN switch. Shortcomings of the proposed auto-configuration system are that it may be leveraged by an intruder to launch an attack against the SDN controller and that it may introduce a loop in the existing network.

A semantic-based technique for configuring legacy network devices in a hybrid SDN network has been proposed in [216]. More specifically, the Web Ontology Language [239] is used for the development of a legacy device (switch/router) ontology for processing extracted information at the command-line interface (CLI) of the legacy network device. The proposed system extracts configuration information from the CLI of the legacy device and is referred to as Ontology-Based Information Extraction System. A semantic learning algorithm [240] is proposed to infer the legacy device configuration from its CLI. The system input consists of the switch/router CLI and the ontology. Then, the proposed system generates the legacy device specific ontology. The device specific ontology is produced based on configuration command instances from the semantics of each CLI space. The experimental results in [216] confirm that the proposed technique effectively extracts the semantics of the configuration using the CLI interface of the device with 90% classification accuracy.

For a hybrid SDN network, Mishra et al. [217] have developed a model to implement SDN-like policies using the OpenFlow protocol with legacy switches. In the proposed network, each subnet is connected to at least one SDN switch which monitors and controls the subnet. The main idea of the proposed framework is the use of unused IP addresses to implement SDN-like policies at legacy switches without upgrading the entire network to SDN. In a traditional network, packet forwarding and policy implementation are based on destination IP addresses. Some of the IP addresses remain unused in the network, so these can be assigned to SDN devices that are deployed in the network to form a hybrid SDN network. Static routes are installed in the legacy switches using paths traversing the SDN devices. In this way, all traffic in both destination and source subnets can be managed using the new IP address.

Amin et al. [218] have presented the automatic Policy violation Detection for Topology Change (auto-PDTC) policy configuration mechanism for topology changes. The network topology changes frequently due to the addition or removal of devices or links in the network. Network policies that are configured on different network device interfaces are often severely affected by network topology changes. Auto-PDTC provides an automatic topology change detection mechanism that validates the policies for the respective interfaces. A graph-based topology change detection mechanism [241] has been adopted to detect topology changes. ACL policies are represented in the form of a 3-tuple and a 6-tuple. When a topology change is detected, then a tree traversing method is used to indicate the affected interfaces where policies are violated. The policies for the indicated interfaces are then configured accordingly. A limitation of the study [218] is that only ACL policies are considered; other network policies need to be considered in future research.

Sieber et al. [219] have examined network device reconfigurations in hybrid SDN networks and proposed an analytical model for the recongurability of partial SDN deployments. The reconfiguration times of legacy and SDN network devices are typically different in that legacy device reconfigurations are up to two orders of magnitude slower. The long configuration times of legacy devices may delay rules installation and packet processing as well as delay the detection of topology changes. Sieber et al. have conducted a queuing theory based analysis to quantify and compare the reconfiguration rates of hybrid SDN networks. The results indicate that a small number of slow-to-reconfigure legacy devices can severely slow down the
reconfiguration of hybrid SDN networks.

C. SDN-BGP Based Hybrid SDN Network Management

This section surveys BGP based approaches for managing hybrid SDN networks. The Border Gateway Protocol (BGP) is the default inter-domain Internet routing protocol. In an SDN network, BGP routing performs a vital role by distributing OpenFlow messages to manage SDN packet flows.

Gámperli et al. [220] have proposed an emulation framework for a hybrid SDN-BGP network with multiple Autonomous Systems (ASs). Gámperli et al. use the proposed emulator to measure the BGP convergence time in a hybrid SDN-BGP network. To support the Hybrid SDN-BGP network emulation, Mininet [195], a tool used for simulating SDN, is integrated with the Quagga [122] BGP software. The framework automatically configures the network. Each AS is emulated as a single router to reduce interference between inter-domain and intra-domain routing. The experimental results show that inter-domain routing centralization can reduce the convergence time with SDN cluster deployment. The proposed framework provides a high-level API for rapid prototyping. However, the framework is not suitable for emulating state consistency and concurrency issues.

The Virtual Routing Engine (VRE) [242] is a system that enables flexible operation management and on-demand routing in hybrid SDN networks. VRE operates above the SDN controller. A topology discovery module extracts the topology information from the traditional devices and SDN switches in the hybrid SDN network. The VRE server enables flexible topology abstraction of the physical network. In the VRE server, each VRE instance is a virtual machine that is equipped with a Unix IP-based engine [243]. The VRE system has been verified by implementing a prototype hybrid SDN network containing OpenFlow and legacy switches.

Vanbever et al. [222] have developed a lightweight hybrid SDN model that can be adopted to obtain SDN benefits in a traditional network. The model eases the configuration management compared to traditional networking to facilitate improved traffic engineering and load balancing. The core part of the model is a flexible and vendor-independent API for SDN [244] that operates on top of the traditional network. The API programs the forwarding entries in legacy switches, similar to the programming of SDN switches. Vanbever et al. [222] consider the Link State Interior Gateway Protocol (IGP) intra-domain routing protocol [245], the BGP inter-domain routing protocol [187], and the MPLS forwarding mechanism [246]. A runtime component of the API is responsible for managing the message forwarding between different network devices. This runtime API translates the forwarding paths into routing protocol messages. This translation function augments the IGP network topology with a virtual network topology consisting of virtual nodes, links, and destinations. Routers thus effectively adopt the virtual topology for packet forwarding.

Kotronis et al. [223] have developed a hybrid BGP-SDN framework to improve the performance of traditional network protocols, such as BGP and the Minimum Route Advertisement Interval (MRAI) [247], by using the SDN concept. More specifically, SDN is used to improve the inter-domain routing properties, to control the routing activities using the central SDN controller, and to form logical autonomous systems (ASs) [221]. Hybrid BGP-SDN employs a central Network Operating System (NOS) to manage multiple ASs through a single SDN controller. The proposed solution, called hybrid BGP-SDN, is made publicly available as an emulation framework.

D. Energy Efficiency

Networking devices consume a lot of energy for their operations. Many protocols have been proposed for ensuring energy efficient operation in traditional networks. In a pure SDN network, the controller has complete topology knowledge, which can readily be exploited for energy efficient operation [248], [249]. This section surveys network management protocols for ensuring energy-efficient operation in hybrid SDN networks.

Wang et al. [224] have addressed the problem that switches often consume more energy than necessary due to pronounced underutilization of links. Link utilizations in large Internet service provider networks are on average less than 40% [15]. Hence, disproportionately large amounts of energy are often used to support relatively small data traffic demands. Therefore, there is a need to find mechanisms that adapt the energy usage in proportion to the data traffic demands. For a hybrid SDN network setting, Wang et al. [224] have proposed to determine subsets of the network that are able to support the traffic demands. Then, the network subset with the minimum energy consumption is kept operational and the other parts of the network are switched off. The proposed mechanism assumes that energy costs of switches and links are known, e.g., from their types and configuration settings. The SDN controller in the proposed mechanism adjusts the power states of the SDN devices (but not the legacy network devices) to satisfy the data traffic demands with minimum energy expenditures. More specifically, the proposed approach switches on a subset of SDN switches that together with the legacy switches satisfies the data traffic demands and switches off the rest of the SDN switches. Wang et al. [224] have used a multicommodity flow (MCF) [250] formulation to represent the data flows and the power states of links and switches. The power states of links and switches are represented by binary variables in the MCF. The MCF formulation is then represented as a mixed-integer programming (MIP) problem. The MIP is NP-hard [250], i.e., has high complexity, requiring long computation times to find the optimal solution. Therefore, Wang et al. have proposed the following low-complexity heuristic approach:

- Using a tree structure topology, connect nodes that want to exchange data traffic
- Creates spanning trees of some subsets of nodes, called small closest sets (SCSs).
- As edges of a spanning tree, select routes with the lowest energy consumption between nodes.

The proposed approach selects paths with minimum weights between nodes by forming a group of spanning trees. In case this group does not satisfy the data traffic demand, then the...
overloaded links are removed by generating other groups of spanning trees as per the existing topology. Thus, several groups of spanning trees are typically used to satisfy the data traffic demands. The main limitation of this approach is that it considers only SDN devices for minimum energy consumption. However, a hybrid SDN network has both legacy and SDN devices. There is a need for a mechanism that considers both SDN and legacy devices to minimize the energy consumption. Second, the proposed approach is computationally expensive since the step of finding the minimum power network subset is an NP-hard problem and the proposed heuristic approach is still quite computationally demanding.

Huin et al. [225] have examined the energy consumption in hybrid SDN networks and proposed the Smooth ENErgy Aware Routing (SENAtoR) algorithm for energy-aware routing. SENAtor selects alternative routes and puts some links and devices into sleep mode. The main SENAtor strategy is to select the routes that utilize a minimum number of network devices. SDN devices that are not utilized are then put into sleep mode, which includes turning off their network interfaces. During sleep mode of SDN devices, their links are also down, so traffic is redirected to alternate paths. It is also desired to enable the network devices immediately when needed. For this purpose, three methods are adopted. In the first method, pre-set tunnels are used as back-up routes in case of link failure and switch-off (sleep) mode. In the second method, the SDN controller suppresses any incoming OSPF traffic to pretend a link disconnection on the interface to deactivate. The third method performs SDN monitoring to rapidly detect unpredicted traffic peaks and link failures. Extensive simulations for different network topologies indicate that SENAtor is a good energy saver. When green services are enabled and traffic spikes occur in legacy devices, SENAtor achieves lower loss rates than an all-OSPF benchmark case. The SDN controller, in particular, helps to find efficient traffic paths. Related path control schemes for energy efficiency have been studied in [251], [252].

Alan et al. [226] have proposed to combine legacy network devices with SDN devices through a so-called “supercharging” strategy. Alan et al. [226] define supercharging to mean the cooperation of legacy and SDN switches through a two-stage forwarding table structure. This forwarding table structure is employed in [226] to reduce routing convergence time and thus to improve energy efficiency. After a network failure, legacy switches typically take a long time to converge because the convergence requires many forwarding table entry updates. Alan et al. [226] have proposed a two-stage forwarding table structure that spans across the two types of devices: (i) Legacy routers hold a primary table, (ii) SDN switches hold a secondary table. In order to speed up convergence, this hierarchical two-stage Forward Information Base (FIB) [253] structure tags entries with the same primary and backup NextHop (NH) in the primary table, and then directs the traffic to the primary or backup NH in the secondary table. If the primary NH fails, then only few entries on the switch need to be updated. Besides the convergence, other aspects of the router operation can be supercharged using the two-stage table structure, e.g., large routing table sizes can be accommodated.

Figure 18. Overview of a forwarding table consisting of TCAM and SRAM in a hybrid SDN network [165]: The router first looks up entries in the TCAM table; if an entry is matched, then the action is immediately applied to the packet (e.g., forward to a port). If no entry is matched in the TCAM table, then the router looks up the SRAM and then takes action on the packet.

The Alan et al. [226] study focuses only on the convergence time for the network; other factors, such as route computation and the network throughput, are not considered.

E. Optimal TCAM Usage

Ternary Content Addressable Memory (TCAM) is a special type of high-speed memory that perform search operations in a single clock cycle [254]–[257]. TCAM memory is used in networking devices, such as switch and routers, to speed up networking tasks, e.g., route look-up and packet classification. This section surveys studies on network management studies focusing on TCAM usage in hybrid SDN networks.

Zhang et al. [165] have introduced hybrid routing to balance the load from multiple traffic classes while incurring low complexity and achieving good scalability. The proposed approach greatly reduces the number of forwarding entries compared with pure explicit routing. The proposed approach thus requires only very small TCAM for implementation. The proposed hybrid routing stores the destination based routing forwarding entries in Static Random-Access Memory (SRAM) [258] and thus substantially reduces the required TCAM. Figure 18 illustrates a forwarding table example. The router looks up entries for sent packets in the TCAM table. If there is a match with a TCAM entry, then the router forwards the packet to the required port. Otherwise, the router continues the search in the SRAM table and performs an action on the packet. The evaluation results in [165] demonstrate that in typical scenarios, the hybrid routing can save around 84.6% TCAM resources as compared to explicit routing.

Othman et al. [227] have proposed a hybrid control model for SDN. Generally, the OpenFlow protocol enables the SDN controller to control the data flows in the network. Moreover, SDN provides great flexibility by separating the decision-making logic in the control plane from the actual forwarding of packets in the data plane. Nevertheless, Othman et al. [227] argue that there are limitations in SDN, e.g., limited scalability and extensive TCAM usage. SDN depends on a central SDN controller to control flows. Othman et al. [259] have proposed a hybrid control model that combines both central and distributed control to provide a relaxed control and a fail-safe [260] mechanism while reducing TCAM usage in hybrid SDN networks.
F. Large Scale Deployment

HP [228] has recommended the hybrid SDN network architecture for large-scale network deployments. In a hybrid SDN network architecture, the controller may delegate some portion of the forwarding decisions to the switches, thus reducing the complexity and amount of traffic from the controller. In a hybrid SDN network, link and end host discovery depend on some specialized rules. These rules are based on applications installed on the controller. For some specific flows, the controller makes data plane decisions, while legacy switches make other decisions for forwarding network traffic. The link and end host discovery at the SDN controller helps to form a complete view of the network topology that is used by controller applications [261]. In addition, hybrid SDN network operation increases performance and scalability by preventing normal traffic (that needs only basic legacy forwarding and no filtering or checking) from consuming space in flow tables that can be used by SDN applications. The evaluation results for a range of scenarios in [228] indicate that the hybrid SDN network has significantly fewer rules installed than a pure SDN network.

Henneke et al. [229] have reviewed the requirements of real industrial networks, e.g., for factory automation [262]–[265], that need to be addressed by the SDN research community. The review postulates six major requirements: (i) In multipurpose heterogeneous network, various applications should be able to share the same infrastructure without interrupting each other. (ii) The different applications running over the future industrial network require different QoS services with their specific needs. (iii) The use of real-time industrial network monitoring is to improve the network performance and to avoid congestion. (iv) To improve industrial network security, one should use private network slicing and configurable security levels. (v) To provide reliable services in real industrial network environments one should use stable control and management platforms and fault tolerant solutions. (vi) The effective operation of the real industrial network can, in principle, be achieved by using a hybrid SDN environment.

G. Summary and Lessons Learned

We summarize the comparison of the surveyed network management studies in Table VI. Resource management and consolidation are vital tasks in a network. Santos et al. [210] have presented basic functioning resource management techniques. However, the techniques by Santos et al. [210] do not address some networking scenarios, which should be examined in future research. The missing scenarios include a fast real-time implementation, the secure and seamless handover for multiple types of networks, the handling of multiple domains with potentially conflicting sets of rules, as well as the identification of diverse device capabilities and the integration with other control planes. SuVMF [211] and Katov et al. [212] have also provided basic resource management functionalities for hybrid SDN networks, with some additional features, namely monitoring functionalities in SuVMF et al. [211] and power consumption minimization in Katov et al. [212]. The performance evaluations in [211], [212] found good performance levels for basic networking scenarios through load balancing, i.e., the re-routing of traffic flows around congested network elements. Future research should extend the performance studies to larger networks to examine the scalability characteristics and should evaluate the reliability [34]. Moreover, future research should address orchestration architectures and protocols of multiple SuVMF instances deployed in various locations. More specifically, future research should consider the operation of multiple instances of SuVMF at different locations and evaluate the resulting reliability, performance, and scalability.

The performance comparison of the resource management approaches in Table VII indicates that the Katov et al. [212] approach has attractive performance characteristics, including low traffic overhead, accommodation of dynamic SDN flows installation, and high performance with middle box deployment. However, the Katov et al. [212] approach is not resilient to path failures because it considers only other traffic metrics.

Mishra et al. [217] have presented an approach for the auto-configuration of SDN-like policies in hybrid SDN networks employing unused IP addresses, while Katiyar et al. [215] have developed the DHCP-SDN auto-configuration module. Sometimes, such SDN-like policies are violated when the network topology changes, as has been examined by Amin et al. [218]. All these three auto-configuration mechanisms by Mishra et al. [217], Katiyar et al. [215], and Amin et al. [218] can be combined in a single application that covers the respective aspects of auto-configuration in a single unified application. While such a single unified application can be efficient and elegant, it may give rise to security threats. Intrusion detection and other aspects of network security, such as ARP spoofing, need to be thoroughly examined for these autoconfiguration mechanisms.

The web ontology-based system in [216] can easily extract the switch information from the log files of legacy switches. Thus, the web ontology-based system can facilitate the automation of entire network systems. However, the study in [216] considered only elementary learning algorithms. Future research should examine the adoption of more efficient learning algorithms, e.g., algorithms based on artificial intelligence, to obtain and process the network topology information. Moreover, enhancements of the learning algorithm can be based on historical instantiations, i.e., the algorithm could perform semantic disambiguation by taking into account previous decisions and semantic-web integration in order to efficiently generate accurate network switch/router topologies.

Among the auto-configuration approaches in Table VII, the Katiyar et al. [215] and Amin et al. [218] approaches perform better than other approaches by having low overhead, accommodation of dynamic SDN flows, and high performance with middle box deployment. However, the Katiyar et al. [215] approach has not considered inter-segment connectivity; while the Amin et al. [218] approach is not resilient to path failures because it is difficult for the SDN controller to find alternative paths from legacy switches. Although the web ontology-based system [216] has high traffic overhead for a fixed number of SDN flows, it is a good configuration mechanism for SDN networks that employ ontology based network management.
### Table VI

**SUMMARY COMPARISON OF NETWORK MANAGEMENT STUDIES**

<table>
<thead>
<tr>
<th>Study</th>
<th>Mechanism</th>
<th>Protocol</th>
<th>Objectives</th>
<th>Evaluation Tool</th>
<th>VLAN</th>
</tr>
</thead>
<tbody>
<tr>
<td>Santos et al. [210]: SDN-based WiFi AP</td>
<td>ID-Based Cryptography (IBC) RADIUS/EAP</td>
<td>SOK, OpenFlow</td>
<td>Resource management in infrastr.-based and infrastr.-less netw.</td>
<td>Java based Application, Floodlight Controller</td>
<td>No</td>
</tr>
<tr>
<td>SDN enterprises the unified virtual monitoring function (SuVMF) [211]</td>
<td>NPV, OF Statistic Collect. Proxy (OSP), Detect and Mitigate Anomaly (DMA)</td>
<td>OpenFlow</td>
<td>Monitor traffic and resource management</td>
<td>Hardware and software module</td>
<td>No</td>
</tr>
<tr>
<td>Katov et al. [212]: Res. usage consolidation, enterprise netw.</td>
<td>Traffic matrix and netw., element est., TE techn.</td>
<td>OpenFlow, STP, OSPF</td>
<td>Resource usage consolidation, reduced power consump.</td>
<td>OMNet++ 4.3.1</td>
<td>No</td>
</tr>
<tr>
<td>Sieber et al. [213]: QoS</td>
<td>Real-time QoS</td>
<td>OpenFlow, NMS, VOIP, QoS for VOIP</td>
<td>QoS for VOIP</td>
<td>StableNet</td>
<td>Yes</td>
</tr>
<tr>
<td>Sieber et al. [214]: Hyb. SDN QoS mgt.</td>
<td>Programmable mgt. for hybrid SDN netw.</td>
<td>OpenFlow, OSPF</td>
<td>Device-neutral programmability</td>
<td></td>
<td></td>
</tr>
<tr>
<td>A. Resource Management</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Katiyar et al. [215]: Locator, Configurator, DHCP-SDN</td>
<td>Locator, Configurator, DHCP-SDN module</td>
<td>OSPF, OpenFlow</td>
<td>Auto-config. of new switches, seamless serv.</td>
<td>Floodlight controller, Custom development</td>
<td>Yes</td>
</tr>
<tr>
<td>Web Ontology Language [216]: Ontology based info. extract. system</td>
<td>Semantic learning algorithm</td>
<td>OSPF, OpenFlow</td>
<td>Bridge configuration gap in hybrid SDN through ontology language</td>
<td>Juniper, Quagga, Custom development</td>
<td>No</td>
</tr>
<tr>
<td>Misra et al. [217]: Assign unused IP addr. to SDN dev.</td>
<td>Custom Algorithm and Model</td>
<td>OpenFlow</td>
<td>SDN-like policies using OF over legacy switches</td>
<td>Mininet</td>
<td>Yes</td>
</tr>
<tr>
<td>Amin et al. [218]: auto-PTDC topology change detection</td>
<td>Auto-PTDC</td>
<td>OpenFlow, OSPF</td>
<td>Auto config. of network policies in hybrid SDN</td>
<td>Mininet</td>
<td>Yes</td>
</tr>
<tr>
<td>Sieber et al. [219]: Optimize reconfig. rate</td>
<td>Queue analy. of reconfig. rate</td>
<td>OpenFlow, BGP</td>
<td>Re-config. of leg. and SDN dev. in hyb. SDN</td>
<td>Real-world topologies</td>
<td>Yes</td>
</tr>
<tr>
<td>B. Auto-configuration</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>G¨amperli et al. [220]: Emulation framework, convergence time eval.</td>
<td>Hybrid SDN-BGP emulation framework</td>
<td>BGP, OpenFlow</td>
<td>Measure routing convergence time in hybrid SDN-BGP netw.</td>
<td>Mininet, Quagga, BGP software</td>
<td>Yes</td>
</tr>
<tr>
<td>Virtual Routing Engine (VRE) [221]</td>
<td>VRE based on virtual machines</td>
<td>OSPF, IGP, OpenFlow</td>
<td>Flexible operation mgmt., on-demand routing</td>
<td>Test Bed with OpenFlow 1.3</td>
<td>No</td>
</tr>
<tr>
<td>Vanbever et al. [222]: Config. mgt. through routing prot. msg.</td>
<td>API to control legacy and SDN switches</td>
<td>OSPF, IGP, BGP</td>
<td>Config. managmt., load balancing, traffic eng.</td>
<td>Production network</td>
<td>No</td>
</tr>
<tr>
<td>Kotronis et al. [223]: Hybrid BGP-SDN framew.</td>
<td>Inter-domain centralization</td>
<td>BGP, OpenFlow</td>
<td>Improve convergence, policy conflict resol., inter-domain troubles.</td>
<td>Mininet, Quagga</td>
<td>No</td>
</tr>
<tr>
<td>C. SDN-BGP Based Network</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Wang et al. [224]: Switch off subset of SDN dev.</td>
<td>Efficient link util. through min. power netw. subsets</td>
<td>Sleep mode for unused links</td>
<td>Heuristic Algorithm, OSPF</td>
<td>Mininet</td>
<td>No</td>
</tr>
<tr>
<td>Huin et al. [225]: SENAtoR</td>
<td>Adapt routing to traffic load to spare some hardware</td>
<td>SENAtoR- Smooth ENergy Aware Routing algorithm is proposed</td>
<td>OSPF, OpenFlow</td>
<td>Mininet plus SENAtoR Algorithm</td>
<td>No</td>
</tr>
<tr>
<td>Alein et al. [226]: Hier. two-stage fwd. table</td>
<td>Energy eff., reduced routing converg. time</td>
<td>Two-stage method in hybrid SDN</td>
<td>FPGA, OSPF, BGP</td>
<td>NX-OS v6.2, FPGA-based generator</td>
<td>Yes</td>
</tr>
<tr>
<td>D. Energy Efficiency</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Zhang et al. [165]: TCAM-GRAM fwd. table</td>
<td>Hybrid configuration routing</td>
<td>OpenFlow</td>
<td>Load bal. for multiple traffic classes, TCAM optim.</td>
<td>ROCKETFUEL</td>
<td>No</td>
</tr>
<tr>
<td>Othman et al. [227]: Hybrid central-distr. control</td>
<td>Custom algorithm and model</td>
<td>OpenFlow</td>
<td>Relaxed control and fail-safe mechanism, TCAM optimization</td>
<td>OMNet ++</td>
<td>No</td>
</tr>
<tr>
<td>E. Optimal TCAM Usage</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>HP [228]</td>
<td>Research and development for large scale deployment</td>
<td>Traditional and OpenFlow Protocols</td>
<td>Delegation of some forwarding decisions to SDN controller</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Hennke et al. [229]</td>
<td>Reviews SDN basics and correlates existing work to requirements</td>
<td>OpenFlow and traditional network protocols</td>
<td>Security, QoS, monitoring related problems and solutions</td>
<td></td>
<td></td>
</tr>
<tr>
<td>F. Large Scale Deployment</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

G¨amperli et al. [220], Kotronis et al. [223], and Vanbever et al. [222] have examined BGP performance improvements and presented hybrid BGP-SDN network management solutions for hybrid SDN networks. To the best of our knowledge it is unclear whether the BGP routing protocol is the best starting point for developing hybrid network management solutions for hybrid SDN networks. We believe that it would be worthwhile to re-examine the design of such hybrid SDN...
### Table VII
**SUMMARY OF PERFORMANCE COMPARISONS OF NETWORK MANAGEMENT STUDIES**

<table>
<thead>
<tr>
<th>Study</th>
<th>Traffic Overhead</th>
<th>Link Util.</th>
<th>Number of SDN Flows</th>
<th>Path Failure</th>
<th>Middle Box Deployment</th>
<th>Inter-Segment Connectivity</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>A. Resource Management</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Santos et al. [210]</td>
<td>Low</td>
<td>High</td>
<td>Fixed</td>
<td>Not Resilient</td>
<td>Low Performance</td>
<td>Not Considered</td>
</tr>
<tr>
<td>SuVMF [211]</td>
<td>High</td>
<td>Low</td>
<td>Fixed</td>
<td>Not Resilient</td>
<td>Low Performance</td>
<td>Considered</td>
</tr>
<tr>
<td>Katov et al. [212]</td>
<td>Low</td>
<td>High</td>
<td>Dynamic</td>
<td>Not Resilient</td>
<td>High Performance</td>
<td>Considered</td>
</tr>
<tr>
<td>Sieber et al. [213]</td>
<td>High</td>
<td>High</td>
<td>Fixed</td>
<td>Not Resilient</td>
<td>Low Performance</td>
<td>Considered</td>
</tr>
<tr>
<td>Sieber et al. [214]</td>
<td>Low</td>
<td>High</td>
<td>Dynamic</td>
<td>Resilient</td>
<td>High Performance</td>
<td>Considered</td>
</tr>
<tr>
<td><strong>B. Auto-configuration</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Katiyar et al. [215]</td>
<td>Low</td>
<td>Low</td>
<td>Dynamic</td>
<td>Resilient</td>
<td>High Performance</td>
<td>Not Considered</td>
</tr>
<tr>
<td>Web Ontology Language [216]</td>
<td>High</td>
<td>Fixed</td>
<td>Resilient</td>
<td>High Performance</td>
<td>Not Considered</td>
<td></td>
</tr>
<tr>
<td>Mishra et al. [217]</td>
<td>High</td>
<td>Low</td>
<td>Dynamic</td>
<td>Resilient</td>
<td>Low Performance</td>
<td>Considered</td>
</tr>
<tr>
<td>Amin et al. [218]</td>
<td>Low</td>
<td>High</td>
<td>Dynamic</td>
<td>Not Resilient</td>
<td>High Performance</td>
<td>Considered</td>
</tr>
<tr>
<td>Sieber et al. [219]</td>
<td>Low</td>
<td>High</td>
<td>Dynamic</td>
<td>Resilient</td>
<td>Low Performance</td>
<td>Considered</td>
</tr>
<tr>
<td><strong>C. SDN-BGP Based Network</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Gämperli et al. [220]</td>
<td>Low</td>
<td>High</td>
<td>Fixed</td>
<td>Resilient</td>
<td>High Performance</td>
<td>Considered</td>
</tr>
<tr>
<td>Virtual Routing Engine (VRE) [221]</td>
<td>High</td>
<td>Low</td>
<td>Fixed</td>
<td>Not Resilient</td>
<td>Low Performance</td>
<td>Not Considered</td>
</tr>
<tr>
<td>Vanbever et al. [222]</td>
<td>High</td>
<td>Low</td>
<td>Fixed</td>
<td>Not Resilient</td>
<td>Low Performance</td>
<td>Considered</td>
</tr>
<tr>
<td>Kotronis et al. [223]</td>
<td>Low</td>
<td>Low</td>
<td>Dynamic</td>
<td>Resilient</td>
<td>Low Performance</td>
<td>Considered</td>
</tr>
<tr>
<td><strong>D. Energy Efficiency</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Wang et al. [224]</td>
<td>High</td>
<td>Low</td>
<td>Fixed</td>
<td>Resilient</td>
<td>High Performance</td>
<td>Considered</td>
</tr>
<tr>
<td>Huin et al. [225]</td>
<td>Low</td>
<td>Low</td>
<td>Fixed</td>
<td>Resilient</td>
<td>Low Performance</td>
<td>Not Considered</td>
</tr>
<tr>
<td>Alein et al. [226]</td>
<td>Low</td>
<td>High</td>
<td>Dynamic</td>
<td>Not Resilient</td>
<td>Low Performance</td>
<td>Considered</td>
</tr>
<tr>
<td><strong>E. Optimal TCAM Usage</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Zhang et al. [165]</td>
<td>High</td>
<td>High</td>
<td>Dynamic</td>
<td>Resilient</td>
<td>Low Performance</td>
<td>Considered</td>
</tr>
<tr>
<td>Othman et al. [227]</td>
<td>High</td>
<td>Low</td>
<td>Fixed</td>
<td>Not Resilient</td>
<td>High Performance</td>
<td>Considered</td>
</tr>
<tr>
<td><strong>F. Large Scale Deployment</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>HP [228]</td>
<td>High</td>
<td>Low</td>
<td>Fixed</td>
<td>Not Resilient</td>
<td>Low Performance</td>
<td>Considered</td>
</tr>
<tr>
<td>Henneke et al. [229]</td>
<td>High</td>
<td>High</td>
<td>Dynamic</td>
<td>Not Resilient</td>
<td>Low Performance</td>
<td>Considered</td>
</tr>
</tbody>
</table>

Network management solutions by considering some other link state routing protocols, such as IGP or OSPF, as a starting point. Possibly, by building on these other routing protocols or by combining mechanisms from multiple routing protocols, the convergence for network management actions can be improved. Importantly, the hybrid BGP-SDN solutions proposed so far, and any new solutions should be thoroughly tested and verified for large networks encompassing multiple autonomous systems (ASs).

For the SDN-BGP based network approaches, Table VII indicates that the Gämperli et al. [220] and Kotronis et al. [223] approaches have good performance characteristics, including low traffic overhead, resilience to path failures, and inter-segment connectivity. However, the Gämperli et al. [220] approach is mainly designed for a fixed number of SDN flows and requires complex reconfigurations for additional SDN flows and devices. On the positive side, the Kotronis et al. [223] approach reduces BGP convergence times and churn rates for dynamically expanding SDN deployments. The Vanbever et al. [222] approach has several drawbacks, such as high traffic overhead, low link utilization, and low performance with middle box deployment due to custom-built modules that are difficult to integrate in the wider network context.

Reducing the energy consumption of communications networks has become an important aspect of network management in recent years. Wang et al. [224] and Huin et al. [225] have developed energy savings mechanisms that put idle links and devices into a sleep mode. The key challenge for these energy savings mechanisms is to maintain high levels of network quality of service while some network links and devices may be unavailable due to their sleep mode. The proposed approaches route network traffic such that the active (non-sleeping) links and devices are highly utilized while the sleeping links and devices are bypassed. A key limitation of the Wang et al. [224] and Huin et al. [225] approaches is that they put only the SDN devices and associated links to sleep, while the legacy devices are kept fully active. Future research should extend these approaches by putting both SDN devices and legacy devices to sleep as needed to minimized energy consumption while ensuring high network quality of service.

Table VII indicates that the energy efficiency approaches have overall mixed performance characteristics. The Wang et al. [224] approach is resilient to path failures and achieves high performance with middle boxes, but has high traffic overhead. On the other hand, the Huin et al. [225] approach has low traffic overhead and resilience to path failures, but low performance with middle boxes. The Huin et al. [225] approach reduces the energy consumption of ISP networks by 5–35% depending on the penetration of SDN hardware. Overall, the performance comparison in Table VII reveals that the existing approaches focus on individual performance aspects, but neglect the other performance characteristics. There is a need for a comprehensive mechanism that considers all major performance characteristics.

Zhang et al. [165] and Othman et al. [227] have reduced the TCAM usage for flow table entries by delegating some flow rules to SRAM and splitting the flow control between a central and distributed control instances. Future research should examine the TCAM savings potential due to intelligent algorithms that facilitate efficient TCAM usage, e.g., algorithms that predict future network flows. Such prediction algorithms could be exploited to relieve the controller from...
processing several types of transactions, such as node join and link removal.

Large-scale hybrid SDN network deployments have been examined in [228], [229]. HP [228] has discussed the problems related to the large-scale deployments of hybrid SDN networks. In pure SDN networks, the controller explicitly makes all forwarding decisions. In hybrid SDN networks, the controller delegates some forwarding decisions to distributed protocols. For large-scale deployments, distributed controllers for hybrid SDN networks should be developed. Moreover, other vital aspects, including security, interoperability, and flexibility, need to be addressed in the context of large-scale deployment. Henneke et al. [229] have reviewed the requirements of real large-scale networks that need to be addressed by the SDN research community. Henneke noted that real-time control applications and their impact on existing and future time-critical industrial communication protocols are key research challenges.

V. TRAFFIC ENGINEERING

Traffic engineering is a mechanism for optimizing network performance by observing the behavior of data transmitted over the communication links. In traffic engineering, a wide range of optimization techniques is applied to achieve the maximum network performance, e.g., the maximum network throughput. MPLS [266] and GMPLS [267] are widely used traffic engineering mechanisms in traditional networks. Several traffic engineering mechanisms, such as HONE [268], Lime [269], and similar approaches, such as [270]–[273], have been proposed for pure SDN networks. This section surveys the traffic engineering mechanisms that have been proposed for hybrid SDN networks according to the classification illustrated in Fig. 19.

A. Flexible Traffic Management

This section surveys approaches that are focused on providing a high degree of flexibility in the traffic management in hybrid SDN networks. We first survey the hybrid SDN approaches for flexible traffic management that are based on a combination of MPLS and SDN, followed by other approaches.

1) MPLS-SDN Based Flexible Traffic Management:

Casado et al. [274] have conducted an extensive study on the flexibility aspect of traffic management in hybrid SDN networks. Casado et al. have first reviewed the Active Networking [292], Asynchronous Transfer Mode (ATM) [293], and MPLS [246] technologies for flexible network traffic engineering. Casado then focus on MPLS due to its wide usage for traffic engineering and its support for virtual private networks (VPNs) [294]. Casado et al. propose to re-design MPLS for SDN deployments as follows. Casado et al. [274] argue that SDN deployments lack certain capabilities, such as lookup by headers, sufficient flexibility, and switching between IPv4 and IPv6 [295]. Based on these limitations, Casado et al propose a new network model, called fabric, to provide a simple, vendor-neutral, future-proof hardware with a flexible control plane. The network model adopts some MPLS features for the construction of fabric, as well as some SDN features, such as separate control and forwarding planes.

Tu et al. [275] have introduced a tunnel-splicing approach in a hybrid SDN network consisting of MPLS and SDN routers by proposing the following two key mechanisms: (i) An abstraction mechanism maps the underlying network devices into uniform nodes to hide the details of the various devices. (ii) A second mechanism shifts the manipulation of the SDN flow tables and the MPLS label switch tables to an independent tunneling module. The Path Translator module in the SDN controller examines the routes, checks the label space of each router, and generates the respective forwarding rules for the routers along the paths. The tunneling process is illustrated in Figure 21. Suppose that the serial numbers of the routers along a general path from ingress to egress are \( [1, 2, \ldots, n] \). Using Penultimate Hop Popping (PHP) [296], the Path Translator determines the forwarding rules for each route from router \( n - 1 \) backwards to router 1. The tunnel splicing approach has been implemented using several Linux based routers accompanied by Quagga. The routers support the MPLS and OpenFlow forwarding functions. Several experiments have been performed to validate the feasibility and flexibility of the system. The simulation results indicate that the proposed method mitigates congestion and provides high bandwidth paths for network traffic. The tunnel splicing approach also provides fast failure recovery. The convergence time for failure recovery is minimized by the independent operations of the MPLS and OpenFlow protocols.

Sinha et al. [276] have developed a hybrid SDN network that jointly operates based on a mix of SDN and MPLS. MPLS has some similarities with SDN, such as separation of the control and data planes and flow abstraction. ISPs have been using MPLS for network management for many years due to the efficient MPLS traffic engineering mechanism enabled by the separated control and data planes and the flow abstraction. The main idea of the hybrid network approach by Sinha et al. [276] is to divide the network traffic based on equivalence classes of forwarding elements. The controller divides the traffic based on several classes, such as “OFPInstructionGotoTable”, “OFPActionPushMpls”, “OFPActionSetField”, and “OFPActionOutput”, to create a new flow table and output data for the respective port. The standard MPLS data plane is used to coordinate with the SDN control plane to form a hybrid SDN model. Edge routers are SDN devices, while the network core consists of legacy devices running MPLS, as illustrated in Figure 21. This mixing of network devices and control planes provides a interesting option for the smooth transition from MPLS networking to SDN based networking. Label space reduction strategies for such hybrid MPLS-SDN networks have been studied in [297].

2) Other Approaches: Sharma et al. [277] have argued that in traditional networks, network management tools are static and inflexible. In SDN, network management is quite easy and complexity is reduced due to automating complex functions. Although SDN provides fine-grained control over end-to-end network flows and traffic engineering, Sharma et al. [277] argue that there is still a need for additional levels of flexibility in network management and control tasks. To
address this need, Sharma et al. [277] have developed an integrated network management and control system (I-NMS) framework. The I-NMS framework includes many legacy network management functions, which are helpful in deploying hybrid SDN networks. In particular, network operators are often not willing to fully deploy SDN control in their networks. The I-NMS framework allows the flexible deployment of many legacy network management functions in the parts of the hybrid SDN network that still rely on legacy network devices. In addition, with I-NMS, the SDN controller in a hybrid SDN can handle a selected set of flows for fine-grained traffic engineering, while other network traffic flows are handled by the legacy network protocols and management functions in I-NMS.

Nakahodo et al. [278] have developed Smart-OSPF for a hybrid SDN network. In Smart-OSPF, congestion is decreased by distributing network traffic at edge routers only, while the intermediate nodes operate with conventional OSPF routing. The Smart-OSPF objective is to reduce network congestion by using the edge node to apply a good transport policy, i.e., through traffic distribution at edge routers. Nakahodo et al. [278] argue that it is difficult to implement the traffic distribution function that is needed for Smart-OSPF in legacy routers because they are closed boxes from different vendors. On the other hand, an SDN programmable controller is well suited for the implementation of the Smart-OSPF traffic distribution function in a hybrid SDN network.

Zhang et al. [279], [298] have developed a framework to flexibly manage traffic flows in hybrid SDN networks using the traditional network management system (NMS) [299]. In general, the SDN controller interacts with the underlying SDN routers and contains a data acquisition module. The data acquisition module holds information about the underlying SDN network topology and other related information. The developed framework has a unified interface module that is designed such that it can be used by any NMS to obtain information about the underlying SDN network using the simple network management protocol (SNMP) [237]. The Software
Figure 20. Illustration of tunnel-splicing process [275]: Using Penultimate Hop Popping, the Path Translator decides forwarding rules for each router from the penultimate router \( n - 1 \) backwards to the first router 1 on an end-to-end path. Depending on the router type, the tunnel splicing process assigns labels and installs MPLS forwarding rules or installs OpenFlow flow entries.

Defined Network Management Protocol (SDNMP) [279] holds real-time data (information) of the underlying SDN network, such as topology information and flow table status. The SDNMP has two components: (i) the data storage and processing component, and (ii) the view function. The data storage and processing component is responsible for storing data (i.e., information about the underlying SDN network). The view function displays the required information about the underlying SDN network topology, flow table status, and information about different services. The proposed SDNMP framework thus unifies the communication between the traditional NMS and SDN by storing OpenFlow (SDN) events in the MIB [300] and by using an SNMP based interface.

Polverini et al. [280] have studied the traffic matrix (TM) in Internet Service Provider (ISP) networks and specifically the flow spread parameter to mitigate estimation errors [301], [302]. Two basic questions related to flows have been addressed: First, how many additional flows need to be measured for traffic matrix estimation? Second, which explicit flows have a strong influence on the traffic matrix estimation error? Controlling the mixture of legacy and SDN switches in hybrid SDN networks typically incurs higher operational costs than controlling only one type of switch. Moreover, SDN switches have limited sizes of the expensive TCAM memory, which needs to be used efficiently. To address these problems, Polverini et al. [280] develop a simple deterministic Flow Spread Base Algorithm (FSBA) based on a single flow spread parameter that is associated with each individual flow. The difference between the upper and lower bound related to the network traffic intensity is called the flow spread. The flow spread parameter can be computed with the help of the available data, such as link loads and the routing information, without any additional information of the TM, such as the actual flow intensities. Simulation results demonstrate that the FSBA improves the quality of the TM approximation with only a few measurements.

B. Efficient Routing and Forwarding

Feng et al. [281] have developed a so-called OpenRouteFlow model to modify routing on legacy devices. A hybrid SDN network lacks an integrated network view and control protocols that operate the entire network. Due to the transmission bitrate limitations of the control channel of legacy devices, complete network status information from the entire network can typically not be transmitted to the SDN controller in a timely manner. To address these issues, OpenRouteFlow keeps the distributed (conventional) routing operational and opens (unlocks) the routing and flow information of conventional routers so that it can be collected by the SDN controller and used to compile a simplified view of the entire network. In this way, network visualization and control are decoupled by combining conventional and SDN routing/forwarding information. This mechanism makes the network control and operations simple and efficient. The OpenRouteFlow architecture has two layers as shown in Figure 22. One layer contains the OpenRouters in the data plane, which exposes...
The RLOC is the IP address of the LISP router for the host, i.e., the RLOC specifies the attachment point of a host to the network. EID-to-RLOC mapping is conducted by a distributed architecture. LISP can be used in hybrid SDN networks to achieve scalability, interoperability, and inter-domain deployment. LISP is based on incremental deployment to control current IP networks. SDN can be adopted by upgrading some routers to function as LISP tunnel routers. In this way, LISP supports the flexible deployment of hybrid SDN networks.

Guo et al. [283] have developed a hybrid SDN traffic engineering model that accommodates multiple traffic matrices (TMs). Traffic engineering with multiple TMs is an NP-hard problem that requires efficient solutions. To optimize the routing, Guo et al. adopt an offline weight setting for legacy devices with an online splitting ratio optimization for multiple TMs. To gain the information from historic TMs, a data mining algorithm is used to store historical data about all TMs and calculates a weight coefficient for every representative TM. Representative TMs are linearly combined with OSPF weights to obtain an expected TM. Simulation results demonstrate that the proposed approach achieves efficient traffic engineering.

Ren et al. [284] have examined the traffic engineering and flow management in hybrid SDN networks and proposed a Flow Routing and Splitting (FRS) algorithm for efficient route management. In [284], the routing problem is formulated as a Mixed Integer Programming (MIP) to optimize the maximum link utilization as well as traffic splitting along each SDN switch. FSR manages the entire routing so that each flow must traverse an SDN switch to improve flow administration and traffic engineering. Simulation results demonstrate that this mechanism performs better than earlier mechanisms, such as [21], [290].

Davoli et al. [285] have examined the traffic engineering problem in the context of an IP carrier network and proposed an SDN-based segment routing traffic engineering model. The SDN controller allocates network traffic to the links according to the link capacity. For flow assignment, a heuristic-based approach is used that minimizes overall network passing time. This heuristic has two steps: First, Constrained Shortest Path First (CSPF) allocation provides the initial flow allocations. Second, a heuristic re-allocation re-assigns all admitted flows one by one to minimize the overall network passing time. The second step is executed multiples times until no further improvements are achieved. The experimental evaluations of this system indicate that the proposed SDN-based segment routing achieves effective traffic engineering in IP carrier networks.

C. Forwarding Inconsistencies

A forwarding inconsistency is the forwarding of data traffic through unpredicted routes. Forwarding inconsistencies can create severe problems for network traffic management. For instance, forwarding inconsistencies can allow data traffic to bypass a firewall or can create forwarding loops and traffic black holes. The following approaches have been proposed to address forwarding inconsistencies in hybrid SDN networks.

The sweet-little-lies study [286] notes that distributed routing protocols in traditional networks, such as OSPF [178] and IS-IS [303], are scalable and robust, but not flexible. By using a centralized controller, SDN provides fine-grained routing control, i.e., a very high degree of flexibility, at the cost of control overhead, which subsequently leads to scalability problems. The sweet-little-lies study [286] proposes an approach, called fibbing [304], [305], which combines the advantages of SDN route control and traditional routing protocols. Fibbing offers central control of the distributed computation of forwarding paths while striving for both flexibility and scalability. Fibbing introduces fake nodes, fake links, and fake routes to achieve load balancing. Through this fibbing, the SDN controller can persuade the legacy routers into computing desired paths by presenting the routers with carefully crafted network topologies. Fibbing does not require the installation of low-level rules in the legacy forwarding devices. Instead, the legacy routers simply run the legacy routing protocols based on their link state knowledge (which includes the fake nodes and links). Fibbing is not restricted to destination-based forwarding over loop-free paths. Fibbing can similarly support other network management functions, e.g., fibbing can steer traffic to resolve forwarding inconsistencies. A limitation of this approach is the computational effort for finding the fake nodes and links. If the network topology changes, then the fake nodes and links need to be recalculated.

Vissicchio et al. [21] explain that updating a hybrid SDN network can create forwarding inconsistencies due to the simultaneous existence of different control planes. Legacy devices use link state routing, such as OSPF [178], or distance vector routing, such as RIP [306], at the control plane to build the forwarding tables. SDN devices get commands from the SDN controller to populate their forwarding tables. Vissicchio et al. [21] propose a Generic Path Inconsistency Avoider (GPIA) technique for avoiding forwarding inconsistencies through network updates. In the considered hybrid SDN network, nodes runs both the SDN protocol and a traditional routing protocol to compute their forwarding tables. A change from an initial network configuration to a final network configuration is defined as a network update. More specifically, the
network update is a set of operations on a node. An operation can modify the mechanism used for computing forwarding tables. Moreover, an operation can add or delete entries in forwarding tables at legacy or SDN devices. The proposed GPIA approach assumes that (i) both the initial and the final network configurations have consistent forwarding, and (ii) the traditional network protocol is forwarding consistent in absence of the SDN protocol and vice versa. The proposed approach updates a single node at a time and examines whether this update causes forwarding inconsistencies by simulating the change, i.e., by computing the resulting intermediate forwarding paths, and comparing the intermediate configuration with the initial and final network configurations. Future work needs to extend the GPIA approach to consider more than one node. Also, the scalability and flexibility characteristics of GPIA need to be thoroughly studied.

Wang et al. [287] have studied the problems and inconsistencies of incremental SDN deployments and have proposed a generalized solution to handle the heterogeneity of hybrid SDN networks. More specifically, Wang et al. have developed a heuristic algorithm to select the switches that should be upgraded to SDN devices to gain more control over the entire network, while avoiding forwarding inconsistencies. A new basic traffic engineering method is adopted for traffic distribution and resolving the traffic inconsistencies. Both distributed and centralized routing models are combined to provide strong traffic engineering capabilities. Performance evaluations for three network topologies indicates that the proposed SDN switch placement mechanism is helpful for achieving efficient traffic engineering.

Wang et al. [288] have further studied the traffic engineering problem in hybrid SDN networks with a graph-based traffic distribution mechanism. Hybrid SDN networks feature different traffic forwarding mechanisms, namely forwarding based on SDN control and forwarding according to distributed routing by the traditional network devices. The mixing of these protocols may generate inconsistencies that deteriorate performance. To improve the performance of such hybrid SDN networks, the flexibility of a few deployed SDN switches can be used to avoid forwarding inconsistencies. Wang et al. take the different forwarding properties of the network devices into account and model the traffic engineering problem based on forwarding graphs that accommodate traffic distributions. Forwarding graphs are constructed to achieve high throughput while avoiding forwarding inconsistencies. The traffic distribution mechanism is optimized through linear programming. Simulation results indicate that forwarding graph-based traffic engineering achieves significant improvements over existing techniques.

D. Efficient Link Utilization

The network links carry the network traffic. Typically, some links are under-utilized, while others are congested. Ideally, all links should be equally utilized for achieving good performance. This section surveys mechanisms for efficient link utilization in hybrid SDN networks.

He et al. [289] have proposed a fast algorithm for near-optimal traffic engineering in a hybrid SDN network. Two types of traffic flow through the considered hybrid SDN network: SDN traffic controlled by the SDN controller and uncontrolled traffic (generated by legacy devices) that also needs to be controlled by the SDN controller. He et al. [289] have addressed the maximization of the link utilization while making traffic engineering highly flexible and elegant. A novel routing protocol for traffic engineering is designed to accommodate the SDN traffic through traditional network devices. The routing protocol is a hop-by-hop protocol that combines the capacities of the traditional network and the manageability of SDN. The designed routing protocol is compatible with SDN as well as traditional networks. Network operations in the hybrid SDN network are classified into two modes: a barrier mode and a hybrid mode. A conventional technique for the placement of SDN devices among legacy devices is introduced for the barrier mode. In the barrier mode, SDN devices work in an overlay network fashion and a prescribed portion of the capacity of each link is reserved for the overlay network operation. In the hybrid mode, SDN traffic and traditional traffic equally share the network resources. A fully polynomial-time approximation scheme (FPTAS) is proposed to optimize the entire traffic in the considered hybrid SDN network modes. This FPTAS is evaluated with theoretical and numerical analysis as well as simulations. The simulation results demonstrate that FPTAS achieves near-optimal traffic engineering in the examined hybrid SDN network modes.

Guo et al. [290] have proposed the so-called SDN/OSPF Traffic Engineering (SOTE) traffic engineering algorithm for hybrid SDN networks. OSPF [178] based routing tends to cause network congestion on the shortest paths in traditional networks. Guo et al. propose to exploit the centralized SDN control to mitigate this OSPF congestion problem. The proposed method optimizes the OSPF weight setting so as to lower the maximum link utilization. The SDN controller splits flows arriving to SDN switches. The legacy nodes run OSPF for their normal operation. The SOTE algorithm achieves acceptable performance when only 30% of the switches are SDN-enabled. A limitation of the SOTE study in [290] is that only the congestion aspect is considered. Other network aspects, such as network loops and black holes, should be considered in future research.

A Fully Polynomial Time Approximation Scheme (FPTAS) for maximizing the network traffic flow [291] has been proposed to enhance the traffic engineering in hybrid SDN networks. In hybrid SDN networks with only a small to moderate number of SDN switches, most of the data traffic passes through legacy switches, while only a limited portion of the traffic passes through SDN switches. To control the overall traffic in a hybrid SDN network, the SDN controller has typically only limited capabilities to control legacy switches. The main FPTAS objective is to attain the maximum flow that can be achieved by tuning forwarding behaviors of the SDN switches. Specifically, FPTAS [291] aims to maximize the network traffic flows by solving the Max-Flow problem. As there can be an exponential number of links in the network, the problem cannot be solved through linear programming. Thus, FPTAS derives a polynomial time formulation for the Max-Flow problem.
For efficient traffic engineering with the proposed FP-TAS, the SDN controller obtains traffic information using two approaches: first, by inquiring about historical data on SDN switches and amounts of traffic between legacy and SDN devices, and second, by collecting link load statistics with SNMP or similar mechanisms. Based on this traffic information, the SDN controller generates forwarding rules for SDN switches. These forwarding rules regulate the traffic distribution in the entire network. In this way, FPTAS finetunes the forwarding behaviors of the SDN devices and thus enhances the capabilities of the traditional network. A key limitation of the FPTAS study [291] is that it only considers the OSPF routing protocol for the legacy network.

E. Summary and Lessons Learned

Table VIII summarizes the surveyed studies on traffic engineering in hybrid SDN networks. Casado et al. [274], Tu et al. [275], and Sinha et al. [276] have examined MPLS-SDN based traffic engineering mechanisms. Moreover, Casado et al. [274] have considered switching between IPv4 and IPv6 as well as the layering of SDN approaches; these additional features are particularly important for large-scale networks. To connect multiple networks in large scale network structures, tunneling approaches could be based on the hybrid MPLS-SDN system proposed by Sinha et al. [276]. Future research should examine the Sinha et al. [276] model and related tunneling approaches for large-scale networks with a wide range of traffic flows.

Sharma et al. [277] have presented the i-NMCS integrated network management system for dynamic hybrid SDN networks. The i-NMCS can support a wide variety of end user devices and can incorporate device and user related information. Future research needs to evaluate the i-NMCS for heterogeneous network environments.

Nakahodo et al. [278] have developed a smart-OSPF traffic distribution function at SDN edge switches to mitigate congestion in the legacy core network. While Smart-OSPF can reduce latency by mitigating congestion, Smart-OSPF may also contribute to network latency increases due to the added computational processing on edge switches. The Smart-OSPF study [278] has considered relatively small networks. However, for large networks, the added latency problem may become critical. Future research should extend Smart-OSPF by considering some SDN devices in the core of the network, which could be utilized to improve the traffic engineering. Also, the trade-off between network latency reduction due to the added traffic engineering and the required computation times should be thoroughly evaluated for a wide range of network sizes. Potentially, novel customized algorithms are required for solving the traffic distribution function in the edge SDN switches in a computationally efficient manner for large networks.

Zhang et al. [279] have provided a software defined network management protocol (SDNMP) for hybrid SDN networks by building on the traditional network management system (NMS) and simple network management protocol (SNMP). While SDNMP has basic functionalities for maintaining the data about underlying SDN and traditional networks, the flexibility and scalability properties of SDNMP need to be thoroughly examined in future research. Moreover, the SDNMP techniques need to be extended to manage a wide range of heterogeneous devices and networks.

Polverini et al. [280] have examined the traffic matrix (TM) in Internet Service Provider (ISP) networks and have investigated a flow spread parameter to reduce the TM estimation error. The resulting Flow Spread Based Algorithm (FSBA) introduced by Polverini et al. [280] requires new measurements and additional entries to be stored in the TCAM of SDN switches. This overhead may overwhelm the switch memory. Thus, future research should examine estimation algorithms that avoid the extra load on the switch TCAM.

The performance comparison of the flexible traffic management approaches in Table IX reveals that the Polverini et al. [280] approach achieves good performance in all major characteristics, except that it is not resilient to path failures as it does not consider alternative paths. Among the other approaches, Sinha et al. [276] and Zhang et al. [279] have attractive performance characteristics, including low traffic overhead, resilience to path failures, and high performance with middle box deployment. Zhang et al. [279] also achieve near-optimal load balancing as compared to traditional routing, while Sinha et al. [276] optimize the bandwidth for the network. The Casado et al. [274] approach installs SDN flows dynamically and has high link utilization capability but is not resilient to path failures.

Feng et al. [281], LISP [282], Guo et al. [283], Ren et al. [284], and Davoli et al. [285] have examined the efficient routing and forwarding traffic engineering problems in hybrid SDN networks. The Locator/ID Separation Protocol (LISP) [282] is a southbound SDN protocol that splits the IP address identity from its location. LISP requires extra headers to encapsulate traffic as well as mapping updates and mapping resolutions for traffic engineering. The approach by Guo et al. [283] considers multiple traffic matrices (TMs) to increase the efficiency of TE operations. Future research needs to examine how the consideration of multiple traffic matrices can be efficiently scaled for large networks.

Ren et al. [284] have addressed the traffic engineering and flow management in hybrid SDN through the Flow Routing and Splitting (FRS) algorithm for efficient route management. The traffic splitting functionality in the FRS algorithm may lead to longer paths compared to routing without flow splitting. Thus, FRS may degrade the routing efficiency. Future research needs to carefully quantify this routing inefficiency and develop solutions that achieve a prescribed trade-off between traffic splitting (i.e., congestion mitigation) and the hop-lengths of routing paths.

Table IX indicates that for the efficient routing and forwarding approaches, the Locator/ID Separation Protocol (LISP) [282] and Guo et al. [283] approaches have good performance in all major characteristics. The LISP approach flexibly facilitates incremental SDN deployment while achieving high link utilization and scalability in the number of SDN flows as well as inter-operability and inter-segment connectivities. The Guo et al. approach considers multiple
traffic matrices to enhance the routing function; thus achieving efficient link utilization with low latencies. The Guo et al. [283] approach is also resilient to path failures due to its fast re-route calculation method. The Feng et al. [281] and Ren et al. [284] approaches have also positive performance aspects, including low latency and resilience to path failures. In addition, the Feng et al. [281] approach maps SDN flows into traditional network. The fibbing approach is generally suitable for both data center and traditional network.

In [286], fibbing is used for achieving efficient link utilization by introducing fake routes in the network. The fibbing study [286] only considers a simple OSPF network and fibbing requires relatively high computational effort for finding the fake node and links. If the network topology changes, then fibbing requires relatively high computational effort for finding the fake routes in the network. The fibbing approach is generally suitable for both data center and traditional network.

The sweet little lies approach [286] and the approach by Vissicchio et al. [21] are both generally good mechanisms for resolving route inconsistencies. Among these two approaches, the Vissicchio et al. [21] approach is more recent and efficient as it uses the high-performance Generic Path Inconsistency Avoider (GPIA) algorithm; thus, the Vissicchio et al. [21] approach is generally suitable for both data center and traditional ISP networks.

Table VIII

<table>
<thead>
<tr>
<th>Study</th>
<th>Objectives</th>
<th>Mechanism</th>
<th>Technique/Protocol</th>
<th>Evaluation Tool</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tu et al. [275]: MPLS-SDN tunnel splicing</td>
<td>Tunnel splicing mechanism to mitigate congestion</td>
<td>OpenFlow</td>
<td>Splicing btw. MPLS and SDN routers</td>
<td>Quagga, MPLS, OpenFlow 1.3</td>
</tr>
<tr>
<td>Sinha et al. [276]: Hybrid SDN edge-MPLS core</td>
<td>Hybrid SDN model for SDN and MPLS</td>
<td>OSPF, MPLS, OpenFlow</td>
<td>Partition traffic using forwarding equivalence classes at ingress router</td>
<td>Ryu controller, Mininet</td>
</tr>
<tr>
<td>Sharma et al. [277]: i-NMCS</td>
<td>Integrated Network Management and Control System (i-NMCS)</td>
<td>OpenFlow</td>
<td>Integration of dynamic control for flows using OpenFlow</td>
<td>Custom development</td>
</tr>
<tr>
<td>Nakahodo et al. [278]: Smart-OSPF (SDN edge, OSPF core)</td>
<td>Smart-OSPF in hybrid SDN netw.</td>
<td>OpenFlow, OSPF</td>
<td>Reduce netw. congestion through traffic distr. at SDN edge node</td>
<td>OVS, Smart-OSPF</td>
</tr>
<tr>
<td>Zhang et al. [279]: SDNMP</td>
<td>Transparently storing OpenFlow events into MIB</td>
<td>NMS, OpenFlow</td>
<td>Manage SDN using trad. netw. managmt. system (NMS)</td>
<td>SDN testbed over traditional network</td>
</tr>
<tr>
<td>Polverini et al. [280]: Flow Spread Base Alg. (FSBA)</td>
<td>Study traffic matrix (TM) and flow spread param. to mitigate estim. error</td>
<td>BGP, OpenFlow</td>
<td>Flow Spread Base Algorithm (FSBA)</td>
<td>SDN controller and custom development</td>
</tr>
</tbody>
</table>

B. Efficient Routing and Forwarding

<table>
<thead>
<tr>
<th>Study</th>
<th>Objectives</th>
<th>Mechanism</th>
<th>Technique/Protocol</th>
<th>Evaluation Tool</th>
</tr>
</thead>
<tbody>
<tr>
<td>Feng et al. [281]: OpenRouteFlow model</td>
<td>Efficient routing mechanism</td>
<td>OpenRouteFlow and OpenRouteFlow Controller</td>
<td>Link state, IGP, OpenFlow</td>
<td>Java, Floodlight controller, Custom development</td>
</tr>
<tr>
<td>LISP [282]: Locator/ID Sep. Prot.</td>
<td>Split IP address identity from its location</td>
<td>Southbound SDN protocol</td>
<td>OSPF, BGP, LISP</td>
<td>LISP on VM</td>
</tr>
<tr>
<td>Guo et al. [283]: Historic TM weighing</td>
<td>Routing optimizer framework</td>
<td>TE for multiple traffic matrices</td>
<td>OSPF, OpenFlow</td>
<td>Abilene, CERNET, GEANT, Custom Development</td>
</tr>
<tr>
<td>Ren et al. [284]: Flow Routing and Splitting (FRS)</td>
<td>Reduce congestion</td>
<td>TE and flow management in hybrid SDN netw.</td>
<td>OSPF, OpenFlow</td>
<td>FRS and custom Development</td>
</tr>
<tr>
<td>Davoli et al. [285]: Segm. routing in IP carrier netw.</td>
<td>TE in IP carrier network and SDN-based segment routing TE model</td>
<td>Heuristic to minimize overall network passing time</td>
<td>OSPF, OpenFlow</td>
<td>SDN-TE-SR module, custom development</td>
</tr>
</tbody>
</table>

C. Forwarding Inconsistencies

<table>
<thead>
<tr>
<th>Study</th>
<th>Objectives</th>
<th>Mechanism</th>
<th>Technique/Protocol</th>
<th>Evaluation Tool</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sweet little lies [286]: Fibbing</td>
<td>Efficient link utilization</td>
<td>Multipath routing using fake nodes/links</td>
<td>Fibbing, Link state, IS-IS, IGP</td>
<td>Rocket fuel topologies, Custom development</td>
</tr>
<tr>
<td>Vissicchio et al. [21]: Generic Path Inconsistency Avoider (GPIA)</td>
<td>Resolve forwarding inconsistencies</td>
<td>GPIA algorithm</td>
<td>OSPF, GPIA</td>
<td>Rocket fuel topologies, GPIA Algorithm</td>
</tr>
<tr>
<td>Wang et al. [287]: Incremental hybrid SDN deployment.</td>
<td>Generalized solution to accommodate heterogeneity of hybrid netw.</td>
<td>Heuristic algorithm</td>
<td>OSPF, OpenFlow</td>
<td>Telstra, Ebone, Custom Development</td>
</tr>
<tr>
<td>Wang et al. [288]: Forwarding graph based traffic distr.</td>
<td>TE for hybrid SDN netw.</td>
<td>Graph based traffic distribution</td>
<td>OSPF, OpenFlow</td>
<td>Telstra, Ebone, Custom Development</td>
</tr>
</tbody>
</table>

D. Efficient Link Utilization

<table>
<thead>
<tr>
<th>Study</th>
<th>Objectives</th>
<th>Mechanism</th>
<th>Technique/Protocol</th>
<th>Evaluation Tool</th>
</tr>
</thead>
<tbody>
<tr>
<td>He et al. [289]: Barrier and hybrid mode TE</td>
<td>Fast algorithm for near-optimal TE</td>
<td>Efficient traffic distribution and link utilization</td>
<td>OSPF, IS-IS, OpenFlow</td>
<td>Custom development in C++</td>
</tr>
<tr>
<td>Guo et al. [290]: SDN/OSPF TE (SOTE)</td>
<td>Weight setting for flows and TE for OSPF/SDN</td>
<td>SOTE algorithm is used</td>
<td>SOTE and OSPF</td>
<td>Rocket fuel topologies, custom development</td>
</tr>
<tr>
<td>FPTAS [291]: Max-Flow TE</td>
<td>Traffic control for hybrid SDN, energy optimization</td>
<td>Max-flow with fully polyn. time approx. scheme (FPTAS)</td>
<td>OSPF, Max-Flow, and Max-flow dual</td>
<td>Mathematical formulation</td>
</tr>
</tbody>
</table>
### Table IX
SUMMARY OF PERFORMANCE COMPARISONS OF TRAFFIC ENGINEERING STUDIES ON HYBRID SDN NETWORKS.

<table>
<thead>
<tr>
<th>Study</th>
<th>Latency</th>
<th>Link Util.</th>
<th>Number of SDN Flows</th>
<th>Path Failure</th>
<th>Middle Box Deployment</th>
<th>Inter-Segment Connectivity</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>A. Flexible Traffic Management</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Casado et al. [274]</td>
<td>Low</td>
<td>High</td>
<td>Dynamic</td>
<td>Not Resilient</td>
<td>Low Performance</td>
<td>Considered</td>
</tr>
<tr>
<td>Tu et al. [275]</td>
<td>High</td>
<td>High</td>
<td>Fixed</td>
<td>Resilient</td>
<td>Low Performance</td>
<td>Considered</td>
</tr>
<tr>
<td>Sinha et al. [276]</td>
<td>Low</td>
<td>High</td>
<td>Fixed</td>
<td>Resilient</td>
<td>High Performance</td>
<td>Considered</td>
</tr>
<tr>
<td>Sharma et al. [277]</td>
<td>Low</td>
<td>High</td>
<td>Resilient</td>
<td>High Performance</td>
<td>Not Considered</td>
<td></td>
</tr>
<tr>
<td>Nakahodo et al. [278]</td>
<td>High</td>
<td>Low</td>
<td>Dynamic</td>
<td>Not Resilient</td>
<td>Low Performance</td>
<td>Not Considered</td>
</tr>
<tr>
<td>Zhang et al. [279]</td>
<td>Low</td>
<td>High</td>
<td>Fixed</td>
<td>Resilient</td>
<td>High Performance</td>
<td>Considered</td>
</tr>
<tr>
<td>Polverini et al. [280]</td>
<td>Low</td>
<td>High</td>
<td>Dynamic</td>
<td>Not Resilient</td>
<td>High Performance</td>
<td>Considered</td>
</tr>
<tr>
<td><strong>B. Efficient Routing andForwarding</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Fung et al. [281]</td>
<td>Low</td>
<td>Low</td>
<td>Dynamic</td>
<td>Resilient</td>
<td>Low Performance</td>
<td>Considered</td>
</tr>
<tr>
<td>LISP [282]</td>
<td>High</td>
<td>High</td>
<td>Dynamic</td>
<td>Resilient</td>
<td>High Performance</td>
<td>Considered</td>
</tr>
<tr>
<td>Guo et al. [283]</td>
<td>Low</td>
<td>High</td>
<td>Dynamic</td>
<td>Resilient</td>
<td>High Performance</td>
<td>Considered</td>
</tr>
<tr>
<td>Ren et al. [284]</td>
<td>Low</td>
<td>High</td>
<td>Fixed</td>
<td>Resilient</td>
<td>Low Performance</td>
<td>Not Considered</td>
</tr>
<tr>
<td>Davoli et al. [285]</td>
<td>High</td>
<td>High</td>
<td>Fixed</td>
<td>Not Resilient</td>
<td>High Performance</td>
<td>Considered</td>
</tr>
<tr>
<td><strong>C. Forwarding Inconsistencies</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Sweet little lies [286]</td>
<td>High</td>
<td>High</td>
<td>Dynamic</td>
<td>Not Resilient</td>
<td>High Performance</td>
<td>Considered</td>
</tr>
<tr>
<td>Vissicchio et al. [221]</td>
<td>Low</td>
<td>High</td>
<td>Dynamic</td>
<td>Not Resilient</td>
<td>High Performance</td>
<td>Considered</td>
</tr>
<tr>
<td>Wang et al. [287]</td>
<td>High</td>
<td>High</td>
<td>Dynamic</td>
<td>Resilient</td>
<td>High Performance</td>
<td>Not Considered</td>
</tr>
<tr>
<td>Wang et al. [288]</td>
<td>Low</td>
<td>High</td>
<td>Dynamic</td>
<td>Resilient</td>
<td>High Performance</td>
<td>Considered</td>
</tr>
<tr>
<td><strong>D. Efficient Link Utilization</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>He et al. [289]</td>
<td>High</td>
<td>High</td>
<td>Dynamic</td>
<td>Not Resilient</td>
<td>Low Performance</td>
<td>Considered</td>
</tr>
<tr>
<td>Guo et al. [290]</td>
<td>Low</td>
<td>Low</td>
<td>Dynamic</td>
<td>Resilient</td>
<td>Low Performance</td>
<td>Considered</td>
</tr>
<tr>
<td>FPTAS [291]</td>
<td>Low</td>
<td>High</td>
<td>Dynamic</td>
<td>Resilient</td>
<td>High Performance</td>
<td>Considered</td>
</tr>
</tbody>
</table>

The fake nodes and links need to be recalculated. Future research should thoroughly evaluate the fibbing approach for large heterogeneous networks where multiple packet-switched network and circuit-switched networks are connected.

Wang et al. [287], [288] have examined the problems and forwarding inconsistencies of incremental SDN deployments. Wang et al. [288] have proposed a graph-based traffic distribution mechanism. This mechanism requires quite high computational effort to compute the graph distribution for the entire network.

The Wang et al. [287], [288] approaches perform well in the forwarding inconsistencies category, see Table IX. Both Wang et al. approaches share the positive performance characteristics of accommodating dynamic numbers of SDN flows, resilience to path failures, and high performance with middle box deployment. Moreover, the Wang et al. [288] approach improves the network throughput and provides good load balancing, thus achieving low latency and efficient link utilization. The sweet little lies approach [286] has high latencies due to fake nodes and links that are added to the network.

He et al. [289] have proposed a fast algorithm for near-optimal traffic engineering by using a hop-by-hop protocol that combines the capacities of the traditional network and the manageability of SDN. Theoretical and numerical results verify the correctness of the approach, but the simulation results are based on a custom simulator that does not fully characterize the actual implementation. Future research needs to thoroughly evaluate this approach with standard simulation and emulation tools so as to provide a detailed performance evaluation.

Guo et al. [290] exploit the centralized SDN control to mitigate the OSPF congestion problem by setting OSPF weights to optimize the link utilization. A limitation of the Guo et al. [290] study is that only the congestion aspect is considered. Other network aspects, such as network loops and black holes, should be considered in future research. FPTAS [291] aims to attain the maximum flow through the SDN switches that can be achieved by using Dijkstra’s algorithm and by fine-tuning the forwarding behaviors of the SDN switches. The FPTAS study [291] considers only the OSPF protocol and the Dijkstra algorithm for path computation, which lacks efficiency in highly dense networks. Thus, other routing protocols than OSPF and Dijkstra should be considered in future research.

Table IX indicates that for the efficient link utilization category, the He et al. [289] approach achieves high link utilization with dynamic numbers of SDN flows because it eliminates congestion by using a fast novel traffic engineering algorithm. However, the destination-based routing algorithm in He et al. [289] can introduce high latencies. Also, the He et al. approach performs poorly with middle boxes due to the additional required processing and is not resilient to path failures as it does not consider alternative routes. The Guo et al. [290] approach has similarly low performance with middle boxes, but is resilient to path failures because of its efficient re-routing method.

### VI. NETWORK TESTING/VERIFICATION AND SECURITY

This section surveys studies related to network security as well as testing and verification for hybrid SDN networks. Network security deals with a wide range of attacks, such as ARP spoofing [238], denial-of-service (DoS), distributed denial of service (DDoS), and man-in-the-middle attacks [73]. Only a very limited number of studies have addressed security aspects of hybrid SDN networks. Thus, the topic area of security as well as testing and verification for hybrid SDN networks has vast potential for future research and development. Figure 23 gives an overview of this section.
A. Testing and Verification

Tsai et al. [307] have examined the role of network monitoring and measurements and presented a monitoring system for hybrid SDN networks. SDN models have been implemented in several testbeds used for network verification and implementations. One such testbed is OF@TEIN, a federated testbed in Asia [310] that encompasses several sites. Tsai et al. [307] have specifically proposed SDN based interdomain routing over the OF@TEIN testbed so as to facilitate network monitoring and measurements. In the OF@TEIN testbed, SDN nodes can use public IP networks for information exchanges with each other. The OF and BGP protocols are used in the OF@TEIN testbed as a control architecture. The OF@TEIN testbed may encompass multiple sites that are interconnected with a Software Defined Routing Exchange.

To verify the system operations and management, a monitoring and observation mechanism is deployed over the OF@TEIN testbed. In particular, in the data plane of the testbed network, the forwarding policies for experimental traffic are divided into three parts, namely forwarding policies for LAN traffic, Intra (testbed site)-traffic, and Inter (testbed site)-traffic. Monitoring and measurement require several monitoring indexes in the control plane, such as BGP session, advertised IP prefixes, and routing information base (RIB), as well as in the forwarding information base (FIB). Moreover, if the next-hop information can be recorded by probe packets, it is possible to realize transparent hops in the intermediate network.

B. Security

Chu et al. [308] have examined link failures in hybrid SDN networks. In a hybrid SDN network, SDN switches must properly communicate with the routing protocols implemented in the legacy switches. Many protocols have been developed for overcoming the problems of link failures in traditional networks [311]–[314]. These protocols typically present a trade-off between overhead/reduced throughput vs. availability. Also, these protocols may not ensure 100% path availability. For instance, in the Equal Cost Multi-Path (ECMP) protocol [125], every path is divided into equal-cost segments from source to destination. So, whenever a link is down, ECMP directs the traffic to another path. However, if the alternative paths are not actually available, then 100% failure recovery is not possible. Similarly, in the Multiple Routing Configuration (MRC) protocol [313], [315], all routing configurations are kept in the router. If one link fails, then the router sends all the traffic on another path, bypassing the failed path. A common limitation of these approaches is that they use a fixed recovery path for any particular link failure and cannot respond by comprehensively considering the entire network traffic. This may lead to congestion in the post-recovery network.

The SDN paradigm separates the control plane from data plane and executes the control in a central controller. Whenever SDN detects a link failure, then the SDN controller sends updated routes to the SDN switches based on the traffic load and the traffic type. Thus, load balancing can be achieved through SDN according to the type of traffic flows, as defined by the controller. As the SDN controller has an overall network view, it can easily avoid congested links by providing a set of alternative paths. Multiple paths between each ingress-egress router pair can be maintained and the network condition on each path can be monitored. The least congested path can then be chosen to recover from failures.

Ubaid et al. [309] have proposed a new approach to automatically detect attack conditions and mitigate attacks in hybrid SDN networks. SDN networks need to adopt security mechanisms to protect users from different types of attacks, including new types of attacks such as Link Flooding Attacks (LFAs) and other DDoS attacks. The new approach proposed by Ubaid et al. [309] prevents LFA, ARP Spoofing, and DDoS attacks in hybrid SDN by adding a separate module (server) in the network, where ARP packets are received. Topology information of the entire network is collected at the proposed server. In particular, the network topology information from legacy switches, SDN switches, and from DHCP servers is collected at the proposed server. Flow rules are installed on the SDN switches and the legacy switches are configured to forward ARP packets to the server. ARP packets are analyzed for possible attacks. A graph-based traversal method is adopted.
to detect the location of the attacker. Experimental results indicate that the threats from LFA, ARP Spoofing, and DDoS attacks can be effectively resolved by the proposed approach. Furthermore, the proposed approach supports multiple controllers in the network and can also be used in pure SDN networks.

C. Summary and Lessons Learned

We have summarized the approaches for addressing testing and verification as well as network security in Table X. Tsai et al. [307] have examined the role of network monitoring and measurements and proposed a monitoring system for hybrid SDN networks. As this is the only study focused on hybrid SDN network testing and verification to date, there is ample room for future research on hybrid SDN network testing and verification.

Network security is an important pillar of network management and control. Several security mechanisms have been developed so far for pure SDN networks; however, there is a scarcity of security research for hybrid SDN networks. Chu et al. [308] have developed an approach based on the Equal Cost Multi-Path (ECMP) protocol [125] to recover from a link failure in a hybrid SDN network by finding alternative links for data delivery. Ubaid et al. [309] have proposed a new approach to automatically detect attack conditions and mitigate attacks in hybrid SDN networks. This approach adopts a custom heuristic to compute the attacker’s IP address which requires much effort and time. Future research should explore graph-based techniques to quickly detect the attacker’s position in the network and to perform related defensive measures. Moreover, future research needs to extend these approaches to effectively recover from multiple link failures and to avert other network attacks, such as link flooding and Man-in-the-Middle attacks.

Table XI reveals that among the security studies, both Ubaid et al. [309] and Chu et al. [308] incur high overhead due to the computing of possible attack conditions. The Ubaid et al. [309] approach is resilient to path failures as it considers backup paths to re-route traffic.

VII. FUTURE RESEARCH DIRECTIONS

In this section we identify and outline the main open research problems on hybrid SDN networks that should be addressed in future research.

A. Automated Network Management

It is necessary to verify and debug whether a network performs as per its specifications. For pure SDN networks, several approaches have been proposed for verifying and debugging the network configuration and operation, such as Veriflow [316], HSA [317], and ndb [318]. Veriflow [316] performs real-time network verification. Veriflow sits between the SDN application and the network devices to intercept and check every rule sent from the controller to the network as illustrated in Fig. 24. Good rules are sent to network, while bad rules generate an alarm. HSA [317] performs static analysis of production networks, isolates the network into slices, and checks for possible networking problems, such as black holes and routing loops. Similarly, ndb [318] captures and reconstructs the sequence of events leading to an instance of erratic behavior. ndb uses a postcard model where the switch sends a “postcard” containing the switch id, a version number, and an output port number. These values are then sent to the collector for analysis.

Hybrid SDN networks pose new problems for automated network management. For hybrid SDN networks, there have not yet been any tools proposed for verifying and debugging the network configuration and operations. A key problem is how to collect network state information from legacy devices that use distributed protocols. Suppose we implement a debugging and testing approach similar to Veriflow on the control plane in a hybrid SDN network. If a packet passes only through legacy devices, then this packet will only be handled by traditional protocols and will not visit the controller. How could such a packet be “captured” for controller processing?

B. Management of Network Updates

Network maintenance and upgrades are vital tasks that require care and attention. Upgrades or maintenance of existing device hardware and software requires temporarily the shifting of network traffic to other devices. Nevertheless, the overall network traffic service level should be maintained and congestion should be avoided. A range of mechanisms, e.g., [319]–[322], have been proposed for achieving these objectives in pure SDN networks. Foerster et al. [323] have presented a survey on consistent network updates for SDN networks. The Foerster et al. survey [323] covers several approaches and protocols for updating computer networks in a fast and consistent manner. The survey discusses different network properties and features that are affected during update operations. The survey also classifies different algorithmic techniques that are needed to ensure consistent updates. For hybrid SDN networks, Vissicchio et al. [126] have proposed a mechanism for updating and confirming consistent forwarding in a single node of a hybrid SDN network. Significant network update challenges remain to be addressed in future research:

- How can multiple nodes be consistently updated?
How can load balancing be incorporated into update mechanisms to avoid link congestion?

How can different policies be verified during an update session?

C. Policy Language

Several network policy languages have been developed for pure SDN networks. Mostly, these policy languages belong to the Frenetic [324] family of languages, such as Pyretic [325], Merlin [326], Kinetic [327], and NetKat [328]. These languages translate the network level objectives into corresponding rules to be implemented on SDN devices [329]. For example, Pyretic [325] provides network programmers and operators with a high-level abstraction for building modular control applications. More specifically, Pyretic offers (i) composition operators that simplify the integration of multiple tasks and packet-processing policies in controllers, and (ii) network objects that abstract away the details of the physical network and enable programmers to build applications with a high-level view of the network topology. Hybrid SDN networks pose unique challenges for policy languages, e.g.,

- How can a policy language instruct legacy devices in a hybrid SDN network?
- How can legacy devices be instructed to interact with the SDN controller?
- How are the separate rules for SDN devices and legacy devices generated by a policy language and how can the consistency of these rules be ensured?

To the best of our knowledge, there is presently no policy language for hybrid SDN networks. Thus, policy languages for hybrid SDN networks have enormous potential for future research.

D. Energy Efficiency

Green computing and networking has become a highly important research area in recent years [85], [330]–[332]. Many schemes, e.g., Giroire et al. [333], Bolla et al. [334], and Wang et al. [335], have been proposed to achieve energy efficiency in SDN networks. In a pure SDN network, the controller has the overall view of the complete network and can perform efficient route searches, even if some devices or links are in sleep mode. In hybrid SDN networks, legacy devices may be running energy efficient routing protocols in a distributed manner. The controller in a hybrid SDN network can control the energy efficiency protocol only for the SDN switches [224]. Hybrid SDN networks pose therefore unique challenges for controlling the overall energy efficiency of the hybrid SDN network. For instance, how can we compute the minimum number of devices to put to sleep among the distributed legacy devices and the centrally controlled SDN devices in a hybrid SDN network? Wang et al. [224] have presented an initial energy efficient mechanism for effective link utilization in hybrid SDN, see Section IV-D. However, comprehensive solutions and extensive evaluations are needed for minimizing the energy consumption in hybrid SDN networks and are an important direction for future work.

E. Security

Security is a highly important aspect of any computer network. A wide range of security mechanisms have been proposed for traditional networks and for pure SDN networks. For example, FRESCO [336] and SDNsec [337] are popular security mechanisms for pure SDN networks. Surveys and comparisons of SDN security approaches have been presented in [43], [71], [73], [74]. In an SDN network, all applications, including security applications, such as firewall, DDoS detection, and scan detection, run at the controller. In FRESCO [336], for instance, the security kernel is the main component that runs in the controller and checks for possible security threats. Hybrid SDN networks pose challenging new security problems, for instance:

- Only SDN devices communicate with the controller in a hybrid SDN network, while legacy devices use distributed protocols. So how can legacy devices be included in security applications running at the SDN controller?
- Is it possible that traditional network security protocols can cooperate with the SDN controller and the security applications running in the SDN controller?
- How can the SDN controller detect attacks at legacy devices running distributed traditional protocols?
F. Network Virtualization

Flowvisor [338], Hyperflow [339], and the Optical Flowvisor [340] are prominent network virtualization approaches for pure SDN networks [93]. Virtualization helps to reduce network costs, while improving network performance and efficiency [341]. Flowvisor [338] transforms one given physical network into multiple virtual network slices, whereby each user can independently operate its own network slice. Flowvisor enforces transparency and isolation between the network slices by inspecting, rewriting, and policing OpenFlow messages as they pass through the virtual network slice. Hybrid SDN networks open up several future research directions on network virtualization, including:

- How can a hybrid SDN network consisting of both legacy devices and SDN devices be partitioned into multiple virtual network slices?
- What percentage of SDN devices should be in a slice to achieve effective and efficient network slicing?
- How can these multiple slices be coordinated as they are utilizing the same underlying physical network?

G. Wireless Networks

SDN has not only been used in wired network but also in wireless networks [97], [98], [342]–[346]. Wireless networks have unique features, such as complex signal propagation characteristics, an error-prone shared medium, interference, as well as the hidden and exposed terminal problems. Wireless networks are characterized by frequent network topology changes as well as limited bandwidth. Wireless networks are highly important for a wide range of emerging technologies that involve extensive software controlled computations near wireless clients, e.g., in mobile edge computing [78], [347]–[350] and the Internet of Things [351]–[353].

One promising avenue for employing SDN in wireless networks is to combine SDN with Software Defined Radios (SDRs). SDRs [354], [355] offer enhanced programming of the wireless data plane via a network controller. In particular, SDRs allow the wireless MAC and PHY layer to be defined by the user, allowing for flexible setting of the radio parameters within the MAC and PHY layers. Macedo et al. [99] have argued that software defined radios, network virtualization, and software defined networking should compliment each other in the development of highly flexible wireless hybrid SDN networks.

A few studies have begun to explore frameworks for wireless pure SDN networks. For instance, Odin [356] is a software-defined framework for a enterprise WLAN. Odin provides programmability and deploys different services and features as network application. Odin has a lightweight Virtual Access Point (VLAP) for client connectivity management. This VLAP is separated from the physical AP and different clients that are connected to the physical AP are considered to have a logical connection to the VLAP. The VLAP design not only facilitates the handling of the re-association states but also manages node mobility, load balancing, and network interference. Aeroflux [357] introduces local and global controllers to handle the local and global events for an Odin [357] network. The local controller only handles the events that do not need global coordination. Thor [358] is also an extension of Odin [356] that introduces energy efficient mobility management for WLANs. Similarly, OpenRadio [359], Soft-RAN [360], CloudRAN [361] are some additional examples of SDN-based wireless networks.

Several key challenges for hybrid wireless SDN networks remain unresolved:

- The distributed control plane of legacy wireless devices in a hybrid wireless SDN network requires new mechanisms for the range of functionalities outlined in the preceding Sections VII-A–VII-F for wired networks.
- Network management of hybrid SDN networks with a wireless network component is complex and requires new simple management mechanisms.
- How can paths be determined in multi-hop wireless hybrid SDN networks?
- How can in-band and out-of-band communication be possible due to wireless channel interference?
- Wireless networks have a highly dynamic structure; network slicing is a big challenge due to the channel interference.

Additional hybrid SDN network research arise from the interactions of wireless access networks with the corresponding wired backhaul networks. For instance, how can hybrid SDN techniques aid in coordinating transmissions over the heterogeneous network segments in hybrid wireless-fiber networks? These hybrid wireless-fiber networks, which are also referred to as fiber-wireless (FiWi) networks [362], [363], consist of a wireless network segment and an optical network segment, which is often based on a passive optical network [364]–[368], possibly in conjunction with optical metropolitan area networks [369]–[371]. Each of these network segments has its specific protocol mechanisms that account for the particular physical layer characteristics of the wireless and optical network segments. With the ongoing proliferation of SDN control through some portions of these heterogeneous networks, it will be important to develop effective protocol mechanisms for operating these partially SDN controlled heterogeneous networks.

H. Distributed SDN Controllers

To cope with scalability and flexibility requirements, distributed controllers, such as Onix [372], ONOS [139], Orion [373], Disco [374], and ElastiCon [375], have been proposed for pure SDN networks. A distributed controller should provide scalability, reliability, and simplicity. The distributed controller should still provide a consistent view of the entire network [376] and fast synchronization of network events. Future research directions for distributed SDN controllers for hybrid SDN networks include:

- How can topology information from legacy devices, which are not directly connected to the SDN controller, be efficiently collected?
- How can legacy devices communicate with multiple distributed controller instances?
• How can scalability be achieved with distributed controllers in hybrid SDN networks?

I. Network Measurements

Network measurement and monitoring provide valuable insights for streamlining the network operations and for optimizing network performance [377]. Different types of network measurement and monitoring tools, such as OpenSketch [378], Payless [379], planck [380], and Opennetmon [381], are available for pure SDN networks. For instance, OpenSketch [378] performs network measurement by using sketch based streaming algorithms. These streaming algorithms process data streams that present the input as a sequence of items which can be examined with only a few passes. OpenSketch [378] provides generic and efficient network measurement by separating measurement control and data plan functions. It performs hashing and classification, followed by counting. For hybrid SDN networks, SuVMF [211], see Section IV-A, performs some network monitoring functionalities in the context of its resource management. However, to the best of our knowledge, there is no comprehensive mechanism for measuring and monitoring network activities in a hybrid SDN network.

J. Simulation Tools

Several simulation and emulation tools are available for evaluating pure SDN networks. Mininet [195], EstiNet [382], and GNS3 [383] are prominent tools used for pure SDN network simulations. The hybrid SDN research community currently depends on these same SDN tools, which have generally only limited capabilities for simulating legacy devices. For instance, Mininet can simulate legacy devices to some degree, but does not provide the complete set of functionalities for simulating legacy devices. The following challenges need to be addressed for simulation tools for hybrid SDN networks:

• Presently, VLANs cannot be constructed between legacy and SDN devices.

• Network policy implementation is not a simple task in hybrid SDN networks; it is challenging to adapt pure SDN network simulations with Mininet to correctly reflect policies in hybrid SDN networks.

• How can an SDN controller communicate with legacy devices?

To address these challenges, a dedicated simulation tool needs to be developed for hybrid SDN networks.

VIII. CONCLUSION

The Software Defined Networking (SDN) paradigm has moved network management and control to a centralized controller. While SDN promises a wide range of benefits, organizations are often reluctant to replace their entire traditional network by an SDN network due to a variety of reasons, including cost constraints. Incrementally deploying a few SDN devices among the legacy devices in a traditional network, creates a so-called hybrid SDN network. A hybrid SDN network requires only modest investments into SDN devices and may provide control functionalities approaching those of a pure SDN network. Essentially the hybrid SDN network approach still utilizes the installed traditional network infrastructure, while providing SDN-like control and management.

In this survey, we have comprehensively surveyed the state-of-the-art of hybrid SDN networks. We have categorized the hybrid SDN network survey into five main categories: hybrid SDN network deployment strategies, controllers for hybrid SDN networks, network management techniques for hybrid SDN networks, traffic engineering mechanisms for hybrid SDN networks, as well as testing/verification and security mechanisms for hybrid SDN networks. Throughout, we have summarized the problem scope and contributions of existing hybrid SDN network studies.

We have also identified the gaps in the existing literature and outlined the future research directions that need to be pursued to close the gaps in the existing literature. The main future research directions are the development of automated protocols for network management and network updates as well as the definition of effective policy languages for hybrid SDN networks. Moreover, energy efficiency and security as well as network virtualization and wireless networks pose important challenges for future research. In addition, it is important to develop efficient distributed SDN control techniques for hybrid SDN networks and to develop effective measurement techniques and simulation tools for hybrid SDN networks.

REFERENCES


cdf1-4fd7-b63e-6eea4b1f7a7b


cdf1-4fd7-b63e-6eea4b1f7a7b

C. Trois, M. D. D. Fabro, L. C. E. de Bona, and M. Martinello, “A
E. Haleplidis, J. H. Salim, J. M. Halpern, S. Hares, K. Pentikousis,
M. Karakus and A. Durresi, “Quality of service (QoS) in software de-

C. Rotsos, D. King, A. Farshad, J. Bird, L. Fawcett, N. Georgalas,
S. Khan, A. Gani, A. W. A. Wahab, M. Guizani, and M. K. Khan,
T. Dargahi, A. Caponi, M. Ambrosin, G. Bianchi, and M. Conti, “A
M. Coughlin, “A survey of SDN security research,” University of
P. Fonseca and E. Mota, “A survey on fault management in software-
F. Pakzad, M. Portmann, W. L. Tan, and J. Indulska, “Efficient topology
discovery in openflow-based software defined networks,” Computer
D. Li, S. Wang, K. Zhu, and S. Xia, “A survey of network update in
A. S. Thyagaturu, A. Mercian, M. P. McGarry, M. Reisslein, and
W. Kellerer, “Software defined optical networks (SDONs): A compre-
hensive survey,” IEEE Communications Surveys & Tutorials, vol. 18,
A. S. Rangisetti and B. R. Tamma, “Software defined wireless net-
works: A survey of issues and solutions,” Wireless Personal Com-
F. Chhamik, S. Zandehroudbar, B. R. Tamma, M. Lee, J. H. Lee, and
M. B. Mukherjee, “Software-defined optical networks (SDONs): a survey,”
S. Bera, S. Misra, and A. V. Vasilakos, “Software-defined networking for
internet of things: A survey,” IEEE Internet of Things Journal, vol. 6,
A. Lee, X. Wang, H. Nguyen, and I. Ra, “A hybrid software defined
networking architecture for next-generation IoTs,” KSII Trans. on
E. Molina and E. Jacob, “Software-defined networking in cyber-
physical systems: A survey,” Computers & Electrical Engineering,
survey on software-defined networking in vehicular ad hoc networks:
Challenges, applications and use cases,” Sustainable Cities and Society,
vol. 53, no. 6, pp. 26–34, Jun. 2015.
I. T. Haque and N. Abu-Ghazaleh, “Software-defined wireless network-
ing: A survey and taxonomy,” IEEE Commun. Surv. & Tut., vol. 18,
K. M. Modieigineyane, B. B. Letswamotse, R. Malekian, and A. M-
Abu-Mahfouz, “Software defined wireless sensor networks application
opportunities for efficient network management: A survey,” Computer
T. Huang, F. R. Yu, C. Zhang, J. Liu, J. Zhang, and Y. Liu, “A survey on
large-scale software defined networking (SDN) testbeds: Approaches and challenges,” IEEE Communications Surveys & Tutorials, vol. 19,
A. Blenk, A. Basta, M. Reisslein, and W. Kellerer, “Survey on network
virtualization hypervisors for software defined networking,” IEEE
Communications Surveys & Tutorials, vol. 18, no. 1, pp. 655–685,
First Qu. 2016.
R. Mijumbi, J. Serrat, J.-L. Gorricho, N. Bouten, F. De Turck, and
R. Boutaba, “Network function virtualization: State-of-the-art and re-
search challenges,” IEEE Communications Surveys & Tutorials, vol. 18,
Y. Li and M. Chen, “Software-defined network function virtualization:
J. van de Belt, H. Ahmadi, and L. E. Doyle, “Surveying wireless link


Martin Reisslein (S’96-M’98-SM’03-F’14) is a Professor in the School of Electrical, Computer, and Energy Engineering at Arizona State University (ASU), Tempe. He received the Ph.D. in systems engineering from the University of Pennsylvania in 1998. He currently serves as Associate Editor for the IEEE Transactions on Mobile Computing, the IEEE Transactions on Education, IEEE Access, and Computer Networks. He is Associate Editor-in-Chief for the IEEE Communications Surveys & Tutorials, Co-Editor-in-Chief for Optical Switching and Networking, and chairs the steering committee of the IEEE Transactions on Multimedia.

Nadir Shah received the B.Sc. and M.Sc. degrees from Peshawar University, Peshawar, Pakistan, in 2002 and 2005, respectively, the M.S. degree from International Islamic University, Islamabad, Pakistan, in 2007, all in computer science, and the Ph.D. degree from Sino-German Joint Software Institute, Beihang University, Beijing, China. He was a Lecturer with the Department of Computer Science, COMSATS Institute of Information Technology, Abbottabad, Pakistan, from 2007 to 2008. He is currently an Associate Professor with the COMSATS Institute of Information Technology. He has authored several research papers in international journals/conferences, such as the ACM Computing Surveys and the IEEE Communication Letters. His current research interests include computer networks, distributed systems, and network security. He is serving in the editorial board of IEEE Softwarizations, AHWSN and MCJS. He has been serving as a Reviewer for several journals/conferences, including ICC, INFOCOM, WCNC, Computer Networks (Elsevier), IEEE Communications Letters, IEEE Communication Magazine, IEEE Transactions on Industrial Informatics, and The Computer Journal.