ElasticWISP: Energy-Proportional WISP Backhaul Networks

by

Duncan Ewan Cameron

A thesis
submitted to the Victoria University of Wellington
in fulfilment of the
requirements for the degree of
Master of Engineering
in Network Engineering.

Victoria University of Wellington
2020

Abstract

The provision of rural broadband infrastructure is a challenge for network operators across the globe, irrespective of their size. Wireless Internet Service Providers (WISPs) have shown that the small-scale deployment of wireless broadband infrastructure is a viable alternative to relying on cellular network providers for remote coverage. However, WISPs must often resort to using off-grid renewable energy sources such as solar energy for powering network sites, often resulting in undesirable, low-performance backhaul radios being used between sites out of concern for excessive energy consumption.

The challenges of managing performant wireless backhaul networks in respect to energy constraints at remote, off-grid sites informs the need for *energy-proportional* design. Backhaul radios typically used by WISPs are not *energy-proportional*, meaning they use a consistent amount of energy, irrespective of wireless link utilisation. Using data from a real WISP network, diurnal traffic patterns show that WISP networks could benefit from *energy-proportional* design, without having to sacrifice performance.

To encourage the development of high-performance, *energy-proportional* WISP backhaul networks, *ElasticWISP*, an optimisation architecture that reduces network-wide backhaul energy consumption while satisfying the user-demand for traffic, is introduced. *ElasticWISP* dynamically controls the configuration of backhaul radios based on bandwidth demands and the network-wide energy consumption of these radios. Through simulations driven by real WISP topology and data traffic, results show that *ElasticWISP* can offer energy savings of approximately 65% when WISP operators follow

the proposed backhaul design methodology.

Finally, a lightweight Multiprotocol Label Switching (MPLS)-based traffic engineering scheme, based on Segment Routing, is proposed. The implementation, named Segment Routing over MPLS (SR-MPLS), keeps traffic engineering path-state within each packet, meaning per-flow state is only held at *SR-MPLS* ingress routers. The lightweight approach of *SR-MPLS* also eliminates the otherwise necessary network-wide label flooding of traditional Segment Routing, making it ideal for bandwidth-sensitive wireless backhaul networks. Evaluation of *SR-MPLS* shows that it can perform as well as – and sometimes better than – competitor schemes.

To my parents, Phil and Susanne Cameron.

Acknowledgements

Firstly, I would like to thank my admirable supervisor, Dr. Alvin Valera. You have taught me so much in the short two years that I have known you. I would not be where I am today without your well-considered advice.

A special thank you to Prof. Winston K.G. Seah for believing in me, and enabling this research to be accomplished. Your weekly support has improved my research abilities immensely.

To my peers in the Wireless Networks (WiNe) Research Group – thank you for making the last 12 months so gratifying.

Finally, I would like to acknowledge The Information Society Innovation Fund (ISIF Asia), and InternetNZ/Ipurangi Aotearoa.

ISIF Asia's generosity to the Wellington Faculty of Engineering has funded a range of Internet research relevant to the Asia-Pacific region, including my own.

InternetNZ's support of improving Internet access throughout all of New Zealand will enable me to attend the Institute of Electrical and Electronics Engineers (IEEE)/International Federation for Information Processing (IFIP) Network Operations and Management Symposium (NOMS 2020) later this year.

To ISIF Asia and InternetNZ: I am forever grateful. One day I hope to give back as you have given to me.

Contents

List of Figures

List of Tables

Acronyms

5G Fifth Generation.

AC Alternating Current.

ADC Analogue to Digital Converter.

Adj-SID Adjacency Segment.

AGM Absorbent Glass Mat.

Ah Ampere hour.

API Application Programming Interface.

AQM Active Queue Management.

ARP Address Resolution Protocol.

BFD Bidirectional Forwarding Detection.

CLI Command Line Interface.

CORE Common Open Research Emulator.

CPU Central Processing Unit.

CR-LDP Constraint-based Routing Label Distribution Protocol.

CRC Cyclic Redundancy Check.

CSA-IE Channel Switch Announcement Information Element.

CSPF Constrained Shortest Path First.

DC Direct Current.

DFS Dynamic Frequency Selection.

DPDK Data Plane Development Kit.

DVFS Dynamic Voltage and Frequency Scaling.

EAR Energy-Aware Routing.

eBGP External Border Gateway Protocol.

ECMP Equal-Cost Multi-Path.

EHF Extremely High Frequency.

FIB Forwarding Information Base.

FIFO First in, First out.

FQ-CoDel Flow Queue Controlled Delay.

GbE Gigabit Ethernet.

GDT Gas Discharge Tube.

GPIO General Purpose Input/Output.

GUI Graphical User Interface.

ICIC Inter-Cell Interference Coordination.

IEEE Institute of Electrical and Electronics Engineers.

IETF Internet Engineering Task Force.

IFIP International Federation for Information Processing.

IGP Interior Gateway Protocol.

IP Internet Protocol.

IPFIX IP Flow Information Export.

IPv4 Internet Protocol version 4.

IPv6 Internet Protocol version 6.

IS-IS Intermediate System to Intermediate System.

ISIF Asia The Information Society Innovation Fund.

ISP Internet Service Provider.

ITU International Telecommunication Union.

KVM Kernel Virtual Machine.

kWh Kilowatt hour.

L2VPN Layer 2 Virtual Private Network.

L3VPN Layer 3 Virtual Private Network.

LAG Link Aggregation Group.

LDP Label Distribution Protocol.

LER Label Edge Router.

LFIB Label Forwarding Information Base.

LRP Label Resolution Protocol.

LSA Link State Advertisement.

LSP Label Switched Path.

LSR Label Switching Router.

MAC Media Access Control.

MAN Metropolitan Area Network.

MCF Multi-Commodity Flow.

mmWave Millimeter Wave.

MPLS Multiprotocol Label Switching.

MPPT Maximum Power Point Tracking.

MTU Maximum Transmission Unit.

NIC Network Interface Controller.

NOMS 2020 Network Operations and Management Symposium.

OPEX Operational Expenditure.

OSPF Open Shortest Path First.

PCB Printed Circuit Board.

PHP Penultimate Hop Popping.

PLA Physical Link Aggregation.

PMSR Poor Man's Segment Routing.

PoC Proof of Concept.

PoE Power over Ethernet.

PoP Point of Presence.

PPS Packets Per Second.

QoE Quality of Experience.

QoS Quality of Service.

RSS Root Sum Squared.

RSVP Resource Reservation Protocol.

RTT Round Trip Time.

SBC Single-Board Computer.

SDN Software-Defined Networking.

SID Segment Identifier.

SMA Simple Moving Average.

SNMP Simple Network Management Protocol.

SNR Signal-to-Noise Ratio.

SR-MPLS Segment Routing over MPLS.

SRv6 Segment Routing over IPv6.

SSID Service Set Identifier.

TCP Transmission Control Protocol.

TTL Time To Live.

UDP User Datagram Protocol.

VLAN Virtual Local Area Network.

VoIP Voice over IP.

VPLS Virtual Private LAN Service.

VPRN Virtual Private Routed Network.

Wh Watt hour.

WiNe Wireless Networks.

WISP Wireless Internet Service Provider.

Chapter 1

Introduction

WISP networks play an important role in connecting remote and rural areas to the Internet [1–6]. In New Zealand alone, there are over 30 regional WISPs, connecting more than 70,000 households and businesses to the Internet [7]. Amongst various operational challenges, researchers have identified that many WISP operators build their networks out of necessity in order to provide faster connectivity to deprived communities, even if they lack requisite knowledge to design and maintain wireless networks [2, 3, 8]. While efforts have been made to lower the barrier to entry for such WISP operators to successfully operate networks [1–3, 9], little has been accomplished to overcome one of the greatest operational challenges: off-grid network energy consumption [2, 9, 10].

To gain a better understanding of why energy consumption is important to WISPs, their target markets must be considered. By nature, underserved remote and rural communities often lack ubiquitous deployment of other infrastructure, e.g. power networks. Where a WISP builds Internet infrastructure in an unconnected frontier, there is often an assumption that grid-power will not be available, or will at a minimum be unreliable [3]. To power network equipment such as radios and routers, an appropriate

energy harvesting system must be designed, taking into consideration the business case and cost of the entire system. As WISP markets are typically remote and rural with low subscriber density, budgets are often limited.

WISPs are then presented with a dilemma: build low-cost, low-performance infrastructure to offer basic Internet connectivity, or build high-cost, high-performance infrastructure to offer services competitive to those found in cities. The latter may not be sustainable, although not necessarily due to prohibitively expensive radios, but rather the prohibitively expensive energy harvesting systems that are responsible for providing power to remote sites[1]. Therefore it is in a WISP's best interest to conserve as much energy as possible in order to maintain a sustainable balance between network performance and capital expenditure.

1.1 About WISPs

A WISP network typically consists of a combination of point-to-multipoint customer access radios, and point-to-point backhaul radios. The former is used for distribution of Internet access to end-users, and the latter is used for backhaul, or connecting edge routers to the network core. When deciding what backhaul radios to purchase, WISPs may consider licensed microwave radios to prevent unwanted interference, and consequently, unpredictable service degradation. However, these higher-performance radios, should they be licensed or unlicensed, typically consume far greater energy than their lower-performance counterparts.

Prior research has shown that WISPs can adapt network site configurations to use less energy [8], such as using less radios for customer access purposes, often at the sacrifice of performance. However, many customer access radios available to WISPs have a relatively low energy footprint. In evident contrast, one backhaul radio could use as much energy as five customer

[1] A remote site is also commonly referred to as a Point of Presence (PoP).

access radios. Dedicated backhaul radios are also typically much less flexible in deployment, unless a network operator sacrifices throughput for lower energy consumption. Given this, reducing backhaul energy consumption without disrupting network performance is of particular interest.

Companies such as Ubiquiti Networks [11] have pioneered inexpensive, low-power radios that have enabled low-cost development of Internet access infrastructure in remote and rural communities [8]. However, the majority of these radios operate in unlicensed frequency bands in all but few countries, typically subjecting them to low power budgets and interference from other unlicensed devices operating within the same frequency range. These factors result in unpredictable network performance, impairing even well-established WISP operators [8].

Where unlicensed backhaul radios can be replaced with licensed microwave and other high-performance radios, WISPs must account for their substantially higher power consumption [1]. The considerable energy consumption of high-performance radios makes them unsuitable for many WISPs with a limited budget. Another drawback of such radios is that they are not *energy-proportional*, meaning that they consume a nearly constant amount of energy irrespective of the throughput over them [1, 12].

1.2 Objectives

The key objectives of this thesis are to:

- Investigate WISP backhaul networks and document opportunities for energy reduction.
- Design an optimisation framework to reduce energy consumption in WISP backhaul networks.
- Design a traffic engineering architecture that is suitable for WISP

backhaul networks.

- Validate that the optimisation framework and traffic engineering architecture is of acceptable performance for deployment in real WISP networks.

1.3 Tasks

The key tasks of this thesis are to:

- Survey related work in the area of WISP backhaul networks.
- Survey the state-of-the-art in "green" microwave backhaul networks.
- Implement an energy optimisation framework as required.
- Implement a traffic engineering architecture based on current state-of-the-art research.

1.4 Contributions

The contributions in this thesis are threefold. The first contribution is the WISP backhaul optimisation framework, *ElasticWISP*. The second contribution is the lightweight – or poor man's – MPLS-based Segment Routing architecture, referred to *SR-MPLS* throughout this thesis. The Segment Routing implementation leverages a source-routing paradigm, and enables WISPs to perform fine-grained traffic engineering. By design, the *ElasticWISP* optimiser leverages this traffic engineering capability to enforce optimal-flow paths in a given WISP backhaul topology. The third and final contribution is the power-control architecture that enables *ElasticWISP* to be used as a part of real WISP networks. As a part of this contribution, a series of remotely managed Gigabit Ethernet (GbE)-capable Power over Ethernet (PoE) injectors were designed and manufactured. Additionally,

note that parts of the *ElasticWISP* related content in this thesis has been accepted into the IEEE/IFIP NOMS 2020 for publication.

1.5 Thesis Organisation

Following this introduction, Chapter 2 details a survey of related work, followed by the motivation for *ElasticWISP* in Chapter 3. A brief, high-level overview of the *ElasticWISP* and *SR-MPLS* architectures is then described in Chapter 4. Chapter 4 also describes the formal model of *ElasticWISP*, which is based on a standard Multi-Commodity Flow (MCF) formulation enhanced with energy minimisation and radio configuration constraints. Chapter 5 concisely describes the reference implementation of *ElasticWISP*, as accepted for publication in the IEEE/IFIP NOMS 2020. Thereafter, Chapter 6 discusses the design of *SR-MPLS*.

An outline of the design and implementation of the hardware used to enable practical application of *ElasticWISP* is then given in Chapter 7. Next, using a real WISP network topology, a detailed evaluation of *ElasticWISP* and its energy minimisation potential is presented in Chapter 8. An evaluation of *SR-MPLS* is also provided, benchmarking it against similar existing schemes. Finally, Chapter 9 makes conclusions about *ElasticWISP* and *SR-MPLS*. Planned future improvements, and areas for future exploration, are also outlined in this chapter.

1.6 Additional Information

Please note that when voltage is referred to in this thesis – e.g., 24V powered radios – we assume Direct Current (DC), unless otherwise stated.

Chapter 2

Survey of Related Work

Several related efforts have addressed lowering the cost of expensive off-grid energy systems used by WISPs [9,10]. While these efforts offer important contributions, they do not address the energy-proportionality issues faced by WISPs that desire to build higher-performance networks[1]. This chapter examines related work and identifies gaps in the literature where further opportunities for research exist.

2.1 Sustainability

The globally recognised viability of WISPs, to an extent, relies on them being perceived as sustainable. For a WISP to be sustainable and compete with other commercial network operators, they must be able to provide a competitive service. In many circumstances WISPs are constrained to placing their equipment, such as backhaul radios, in locations that are sub-optimal. Where a WISP places equipment in a remote area where off-grid energy must be harvested, it is critical that the total network energy consumption is kept at a minimum. Ideally, not at the expense of network

[1] In the context of throughput and reliability.

Quality of Service (QoS) and negligent customer over-subscription ratios.

Previous research shows that energy consumption is one of the biggest challenges faced by WISPs that rely on renewable energy sources [8]. Pragmatic research by the University of California, Berkeley and the Google-sponsored Further Reach network discusses the challenges involved with operating solar energy systems and power-hungry backhaul radios [1]. Overall, the challenges involved with using renewable energy harvesting for WISP networks are well understood [2,3,9,10,13–15]. Despite the existence of a well-grounded understanding, it must be recognised that little research has been carried out to find an immediate means of addressing these energy-related network operational difficulties.

2.1.1 Microwave Backhaul Networks

Closely related to WISP backhaul networks are microwave backhaul networks. In some cases, especially where energy harvesting is not constrained, WISPs will use microwave backhaul radios for high-throughput links. It has been found that microwave backhaul networks can account for approximately 50% of overall network energy consumption [16,17], showing the importance of designing energy efficient network architectures. In an effort focused on Millimeter Wave (mmWave) backhaul networks – specifically for supporting omnipresent deployment of 5G infrastructure – energy savings of up to 65% were achieved through turning off both backhaul links and small cells [18]. Evidently, wireless backhaul networks are an ideal candidate for energy efficiency improvements.

Upon further investigation into how cellular carriers handle microwave backhaul energy conservation, it was determined that a common technique simply involves turning off redundant backhaul radios when they are being underutilised. Hence, traffic during periods of low-demand is consolidated over a single link [19,20]. It is further identified that due to the dynamic nature of working with microwave backhaul, e.g., considering frequency

interface, multipath interference, and rain fade, Physical Link Aggregation (PLA) is often necessary to address throughput capacity concerns [20]. Practically, the use of PLA binds microwave backhaul links together, forming a Link Aggregation Group (LAG), enabling higher aggregate throughput to be distributed over the backhaul pairs.

Interestingly, it is also found that most microwave backhaul networks form a tree or chain network topology, and less frequently, a ring topology [20]. While true today, there is a strong consensus that future backhaul networks, such as those that will support Fifth Generation (5G) cellular services, will require a new design paradigm to be realised [18, 20–23]. Particularly, it is believed that future backhaul networks will have dense geographical deployment, and leverage higher mmWave[2] frequencies to ease current-day throughput capacity concerns. While the EHF nature of such backhaul networks enables impressive throughput capacity, it also constrains them to very short physical distance operation, presenting exciting new research challenges.

The use of mmWave backhaul networks within industry has already started to take place. Facebook, in a collaborative effort with Qualcomm, Radwin, Intel, and Nokia, has launched a project named Terragraph [24]. The Terragraph project uses mmWave radios to distribute high-speed – in excess of 1 GbE – Internet access in underconnected urban and suburban areas. Unlike traditional microwave backhaul links that can span tens of kilometres, the Terragraph links have a maximum operating range of 250m, making their use case for WISPs operating in sparsely populated rural areas limited.

In addition, the use of mmWave backhaul radios in WISP networks may only exacerbate energy-related challenges. While urban and suburban environments may be permissive of densely connected mmWave backhaul

[2]International Telecommunication Union (ITU) designated as Extremely High Frequency (EHF).

radios, due to readily available power infrastructure, rural areas are far less tolerant. Powering closely situated mmWave backhaul radios with renewable energy resources may be possible in some circumstances, but with low subscriber density, likely lacks long-term financial sustainability.

Global Internet traffic is predicted to surge to almost double what it is today,[3] by the end of 2022 [25]. Additionally, busy hour[4] network traffic is increasing at an even faster rate. The rapid increase in demand for throughput capacity is troubling for WISPs. Cellular networks are evolving to become significantly faster than they are today, however, WISP networks are in a position where they could be left years behind their urban and suburban counterparts. For Internet-connected rural populations across the world, the lack of fast Internet access risks introducing a second digital divide. Improving energy efficiency of backhaul networks, for any wireless-centric service provider, WISP or cellular, should therefore be of paramount importance to prevent rural communities falling behind.

Given the inevitable increase in throughput capacity required by future backhaul networks, 5G or otherwise, any research contribution made should be general enough to have application on a variety of network topologies. While many existing microwave and WISP topologies might not be meshed, future mmWave backhaul networks might, meaning topology-aware heuristics might not be a suitable approach to initially pursue. In addition, as mmWave technology has application for both WISPs and cellular operators, the literature suggests that any meaningful energy-reduction scheme should be able to operate irrespective of the underlying backhaul technology in use.

[3] As of December 2019.

[4] The busiest 60-minute period of network traffic in a given day.

2.1.2 Energy Optimisation

Numerous general efforts to reduce network wide energy consumption have been made. GreenTE [26] utilises both Open Shortest Path First (OSPF) routing and MPLS, and places line-cards into a sleep-state, reducing backhaul energy consumption by 27%–42%. The GreenTE approach also actively avoids triggering OSPF Link State Advertisements (LSAs) when links are sleeping. While useful for avoiding full network reconvergence, this approach requires either OSPF implementation modification or amendments to the OSPF protocol specification. ElasticTree [27] is another example, reducing energy consumption in data centre networks by up to 60%. Unlike GreenTE, ElasticTree utilises OpenFlow [28] for network traffic forwarding, allowing for explicit enforcement of optimal flow paths. Other conceptual approaches to Energy-Aware Routing (EAR) have achieved similar performance, with energy reduction of approximately 50% [12].

While optimisation approaches that turn off redundant links are generally successful in reducing backhaul-network-wide energy consumption, other approaches to designing "green backhaul" within cellular networks include Inter-Cell Interference Coordination (ICIC) resource allocation schemes. Such ICIC schemes have been shown to improve energy efficiency by up to 50%, however, at the expense of reduced spectral efficiency [29–31]. One of the key problems with applying the likes of ICIC approaches – or any other spectral management solution – to WISP backhaul networks is the inability to integrate any potential design with closed-source vendor software and hardware implementations.

2.2 Traffic Engineering

In the process of researching the challenges involved with operating WISP backhaul networks, it became apparent that traffic engineering is also a challenge faced by many WISPs. One recent example of this was presented

at FaucetCon 2019 [32]. Using the Faucet Software-Defined Networking (SDN) controller [33], a WISP consultancy firm showcased fine-grained traffic engineering, complete with real-time integration with some backhaul radios used by WISPs [34]. The purpose of this demonstration was to show that traffic flows can be balanced across multiple backhaul links, and with relative ease.

Like many SDN controllers, Faucet is designed for controlling OpenFlow-capable SDN switches. However, it should be questioned why WISPs need to use a protocol such as OpenFlow to achieve their traffic engineering needs to begin with. The answer is due to how WISPs build their networks. Previous work showed that OSPF is the prevalent Interior Gateway Protocol (IGP) in WISP networks. Unfortunately for WISPs, OSPF is also a best-effort protocol, and will always use the lowest-cost path for routing traffic. Link-cost metrics can be adjusted to influence flow-paths, however, in many circumstances this is not suitable for performing traffic engineering.

Despite the limitations of OSPF, WISPs have still found ways to use it for traffic engineering. One such approach is the OSPF "Leapfrog" [35], where Virtual Local Area Networks (VLANs) are used to create additional sub-networks. The VLAN tagged path is logically shorter than the alternative non-VLAN tagged path, and an OSPF adjacency is established between the VLAN endpoints. However, inventive approaches to achieve traffic engineering over networks running OSPF do not stop here. Several solutions have used a combination of exterior External Border Gateway Protocol (eBGP) and OSPF to achieve traffic engineering functionality [36–38].

Due to the challenges of implementing traffic engineering with OSPF, MPLS has become a topic of interest for many WISPs in recent years [8]. It should also be noted that despite the recent interest, the concept of using MPLS for traffic engineering has been well established since the early 2000s [39]. However, traditional MPLS traffic engineering mechanisms require more than just an MPLS capable router or switch. Generally, an MPLS traffic

engineering scheme will require:

- Extensions to the IGP of choice, in order to distribute information about link attributes and the network topology.
- A means to perform path computation, such as the Constrained Shortest Path First (CSPF) algorithm.
- Signalling protocols to enable the creation of MPLS Label Switched Paths (LSPs), such as the Resource Reservation Protocol (RSVP), or the Label Distribution Protocol (LDP).
- The necessary RSVP traffic engineering extensions (RSVP-TE) – if using RSVP. Note, RSVP-TE is the intended replacement – as determined by the Internet Engineering Task Force (IETF) – for the now deprecated Constraint-based Routing Label Distribution Protocol (CR-LDP) [40].

It is reasonable to expect that experienced network engineers can build and maintain MPLS networks, including traffic engineering mechanisms as necessary. However, we must also consider the complexity of such networks, as we desire to improve the sustainability of WISPs where possible. While not a technical problem, the shoestring operational nature of many WISPs means that technical competencies may not be high, as discussed in Chapter 1. Irrespective of social challenges, the complexity of traffic engineering mechanisms presents an opportunity for protocol simplification.

2.2.1 Recent Innovations

Given the complexity of implementing traffic engineering – even in a purpose-built MPLS environment – it must be questioned how further simplifications are possible. State-of-the-art research and development from Cisco and the IETF has proven a simple solution is possible: Segment Routing [41]. Unlike traditional MPLS traffic engineering, Segment Routing

does not require a signalling protocol such as RSVP-TE. Instead, an IGP of choice such as OSPF or Intermediate System to Intermediate System (IS-IS) is extended to distribute segments used across the network. Routes are then defined on a per-packet basis, typically either by an IGP or using an optimiser. One of the benefits of Segment Routing is the path-computation flexibility, as even an application could decide on the best path to route its own traffic.

Once a routing decision has been made, an ordered set of segments is encoded into each packet. Segment Routing capable devices then forward the packets based only on the list of segments within the packet. As the segment information is distributed network wide, this ordered set of segments does not need to explicitly form a route. As a result, the encoded stack of segments can be arbitrary, meaning an IGP can route the traffic based on the shortest path – or another constraint based algorithm depending on the network operator's needs – until the next segment – or alternatively, destination – has been reached.

Failure mitigation in Segment Routing networks is handled automatically by the IGP. In addition to IGPs such as OSPF and IS-IS being used to distribute segment information, they are also used for the computation of backup paths. Should failure occur over any path in the network, the IGP is able to automatically compute new paths without waiting for input from an external controller. As a result, Segment Routing has the flexibility to work in a distributed manner without losing the ability to integrate with external optimisation schemes.

Segment Routing has also gained traction in the research community [42], with an open-source IPv6 implementation now part of the Linux kernel [43]. The implementation – known as Segment Routing over IPv6 (SRv6) – supports many of the Segment Routing functionalities, and use cases, as defined in RFC 8402 [41]. However, the implementation of SRv6 in the Linux kernel, at the time of writing, does not support IPv4 in IPv6 encapsulation,

making it unsuitable for meeting the traffic engineering requirements found in WISP backhaul networks.

Next, open-source implementations of MPLS-based Segment Routing were sought out. One approach, dubbed Poor Man's Segment Routing (PMSR), was identified [44]. This approach is minimal, and leverages an MPLS forwarding plane. However, like the Linux kernel implementation of SRv6, PMSR is also unsuitable for WISP traffic engineering, as its minimal approach constrains it to using only a single link between nodes in the network.

2.3 Summary

Despite an abundance of research towards green microwave backhaul networks, a strong opportunity to advance the energy efficiency of backhaul networks used by WISPs is present. In addition, the ability to improve the energy efficiency of WISP backhaul networks is tightly coupled with the ability to overcome traffic engineering challenges faced by WISPs, and by extension, Internet Service Providers (ISPs) of varying sizes. We have seen that while Segment Routing offers a simplification over traditional MPLS traffic engineering schemes, there is still an opportunity to design and implement a suitable MPLS-based approach, as a gap exists within the literature.

Specifically, WISP backhaul networks may benefit most from a PMSR approach to MPLS-based traffic engineering, ideally with very limited network-wide state, and no IGP extensions necessary. Avoiding additional network state and flooding of network-wide MPLS labels is of particular interest due to the unpredictable nature of wireless backhaul links. Next, Chapter 3 builds off the opportunities and challenges identified in this chapter, and motivates the need for *energy-proportional* design in WISP backhaul networks.

Chapter 3

WISP Backhaul Networks

In this chapter, *energy-proportionality* is motivated by showing that traffic patterns in WISP backhaul networks provide opportunities for reduction in energy consumption. It was discovered that for many hours during a given day, high-throughput backhaul radio pairs could be substituted for lower-power alternatives, while still satisfying traffic demand from subscribers. Network access to a local WISP, Venture Networks [45], was leveraged to determine where opportunities to improve WISP backhaul networks exist[1].

3.1 Backhaul Energy Measurement

To understand the potential benefits of energy-proportional operation in WISP backhaul networks, the energy consumption of several backhaul radios targeted at the WISP market was studied. Table 3.1 provides the energy consumption of three radios as reported in datasheets, from self-performed tests using an INA219 [46] energy monitor, and measurements

[1] Venture Networks is located within close proximity of Victoria University of Wellington, making practical data collection possible.

Table 3.1: WISP Backhaul Radio Energy Consumption

Radio	***Max power draw* (W)**[a]	***Low load* (W)**[b]	***Heavy load* (W)**[b]	***Aggregate throughput* (Mbps)**[c]
PowerBeam 5AC G2	8.5	≈ 4	≈ 6	450+
Rocket 5AC Prism G2	9.5	≈ 6	≈ 8	500+
airFiber 5	50	≈ 40	≈ 40	1200+

[a]From radio datasheets.

[b]From self-performed measurements. Results may vary.

[c]Throughput depends on external factors, e.g., variations in the Signal-to-Noise Ratio (SNR) caused by rain fade, interference, and path obstruction. Maximum physical data rate. Values from radio datasheets.

using a Netonix [47] PoE switch that reports the energy draw per port.

Fig. 3.1 shows the daily traffic bandwidth, averaged in 20-second intervals, between two sites of a WISP backhaul network (Able and Heights). The daily traffic peaks at around 400 Mbps of TX traffic. Both sites use full-duplex airFiber 5 radios to support the traffic peaks. From Table 3.1, measurements show that the airFiber 5 uses significantly more energy than the lower-power Rocket Prism, or PowerBeam radios. The most notable aspect of this data is that regardless of the traffic, the energy consumption of the airFiber 5 would remain consistently high at approximately 40 W.

Although there are numerous intervals during the day when the traffic is relatively low – even passing less than 20 Mbps of traffic – the radios still operate (very close to) their maximum power consumption. There is a clear need to enable *energy-proportional* operation of the radios, i.e., the dynamic selection of radios such that energy consumption is minimised

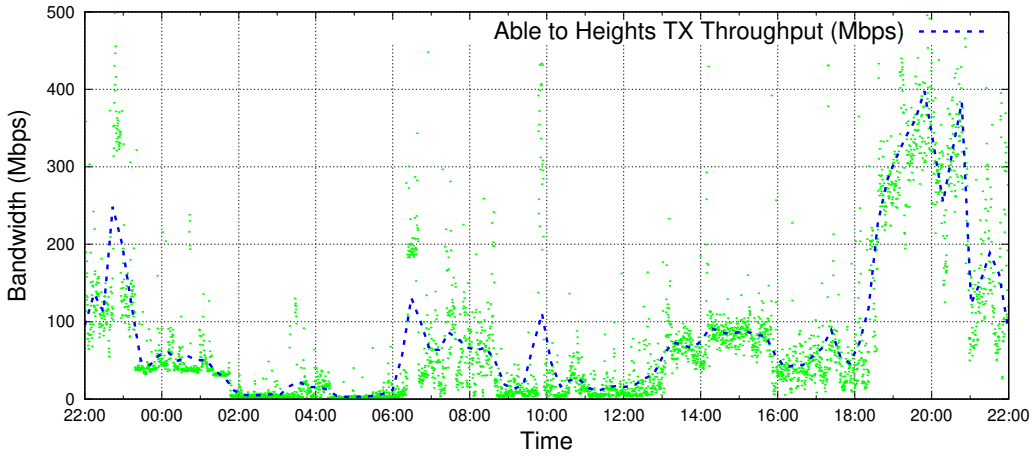

Figure 3.1: Able to Heights TX Traffic (With Bézier Curve).

while maintaining data rate that is sufficient enough to meet traffic requirements. This concept requires each site to be (additionally) equipped with low-throughput, but low-power, radios.

In the case of Fig. 3.1, the high-power airFiber 5 radio would need to be turned on for approximately six hours, depending on link-capacity safety margins set by the network operator. For regular periods of low throughput, even the inexpensive PowerBeam 5AC G2 radio meets the traffic requirements.

3.2 Backhaul Energy Consumption

To draw comparisons about how energy is used by WISP backhaul sites, we consider the energy harvesting and storage requirements of a basic relay site with two backhaul radios operating 24 hours a day. For Tables 3.2, 3.3, and 3.4, we make a rudimentary assumption that the battery bank will be capable of powering the site for two days with severely impaired energy harvesting capability. As the battery banks are lead-acid, they should not be discharged below 50% of their overall capacity.

Figure 3.2: Heights Road Battery Bank.

An example of the battery bank used at the Heights Road site is shown in Fig. 3.2. This battery bank contains several Absorbent Glass Mat (AGM) lead-acid batteries, totalling 600 Ampere hours (Ah) of storage capacity. The battery bank is charged by a 1200 W solar array using a commercially available Maximum Power Point Tracking (MPPT) solar charge controller.

Firstly, in Table 3.2, we consider a relay site that uses two high-power, high-throughput backhaul radios. Secondly, in Table 3.3, we consider a relay site where two low-power, low-throughput backhaul radios are used. Finally, in Table 3.4, we consider a relay site that uses a combination of the two radio types described; high and low power.

We make an optimistic assumption that each high and low power radio pair will operate for 12 hours each per day. Table 3.4 shows that even if both radios operated for 12 hours each per day, the energy reduction is approximately 40% of the highest-consumption configuration shown in Table 3.2.

Table 3.2: WISP Relay Site (2 x airFiber5)

Component	***Minimum value***
Daily consumption (Wh)	2400
Battery bank size (Ah)	800
Solar array capacity (W)	750

Table 3.3: WISP Relay Site (2 x Radio PowerBeam5AC G2)

Component	***Minimum value***
Daily consumption (Wh)	408
Battery bank size (Ah)	140
Solar array capacity (W)	150

Table 3.4: WISP Relay Site (2 x airFiber5 and Powerbeam5AC G2)

Component	***Minimum value***[a]
Daily consumption (Wh)	1404
Battery bank size (Ah)	470
Solar array capacity (W)	450

[a]We assume each radio will operate for 12 hours per day.

3.3 Benefits of Energy-Proportional Backhaul

The energy and traffic throughput measurements show that it is possible to enable WISP operators to benefit from energy-proportional operation in two ways. Firstly, being able to turn off idle resources when they are not needed will make off-grid energy constrained sites more sustainable. For example, where redundant backhauls exist, we can disable and power down one backhaul radio pair to conserve energy.

Secondly, WISPs can be encouraged to design networks that have better energy-proportionality. Having energy-proportional network design is a

key benefit, as WISPs can use high-power, high-throughput radios only as they are needed. In practice, better energy-proportionality requires the deployment of backhauls that include both high and low power radio pairs. Based on earlier measurements, it is shown that designing backhaul links in this way is achievable and will reduce energy consumption should link utilisation not be persistently high.

3.3.1 Sustainability

The dependence that some WISPs – such as Venture Networks – have on off-grid energy systems is a strong motivator for energy-proportional backhaul network design. However, even when using grid-supplied energy, it must be questioned if sustainability – in a far greater scope than is covered in this thesis – should be considered. For example, it is difficult to gauge how much energy is wasted globally due to disproportionate energy consumption of wireless backhaul networks. However, it should be in any network operator's best interest to improve their network sustainability. Ideally, improved sustainability will also decrease network Operational Expenditure (OPEX).

3.4 Summary

Given the difficulty for WISPs to implement traffic engineering schemes using existing routing protocols, there is an evident gap for improvement. Additionally, the "range anxiety" experienced by WISPs is reason itself to aim for improved network energy efficiency. The literature covered in Chapter 2 shows that many research efforts towards "green" backhaul networks are focused towards much larger service provider networks, and are widely not applicable to WISP networks. Efforts to design "green" microwave backhaul networks hold closer relevancy, but are still not tailored towards a WISP on a shoestring budget. To build upon the energy

efficiency and traffic engineering gaps identified, the *ElasticWISP* solution architecture is introduced next, in Chapter 4.

Chapter 4

Solution Architecture

The architecture of *ElasticWISP* consists of three distinct components:

- A network-aware optimiser.
- A flow-path enforcement mechanism.
- A backhaul radio power control mechanism.

This chapter details the high-level design of each component. Later, Chapters 5–7 detail the design and implementation of each major component within the overall *ElasticWISP* architecture.

4.1 Optimisation and Control

The *ElasticWISP* optimiser is required to carry out several functions:

- Collect and process traffic flows to determine the active bandwidth between the network gateway and remote edge routers.
- Automatically update the network topology used for optimisation when link-state changes occur.

- Turn off redundant backhaul radio pairs when the network traffic over them can be routed over lower (power) cost paths.

- Turn on backhaul radio pairs as necessary to manage increases in network throughput where routing over lower (power) cost paths exceeds available network capacity.

- Compute the lowest (power) cost paths for network traffic flows.

- Compute the network-wide power state of backhaul radio pairs.

- Send Forwarding Information Base (FIB) updates to edge routers to enforce optimal flow-paths.

- Send power-control updates to remote backhaul radio pairs to enforce the optimal power state.

- Log network-wide power and link-state changes.

Fig. 4.1 illustrates a high-level, IGP independent overview of the *ElasticWISP* architecture.

Next, in Section 4.2, the multi-commodity flow model of *ElasticWISP* is described. The formal model of *ElasticWISP* is designed to be flexible enough that it can be applied to a variety of different backhaul network types, potentially such as those used in 5G networks. It is also anticipated that the formal model could be used as a benchmark for any future heuristic approaches that are developed.

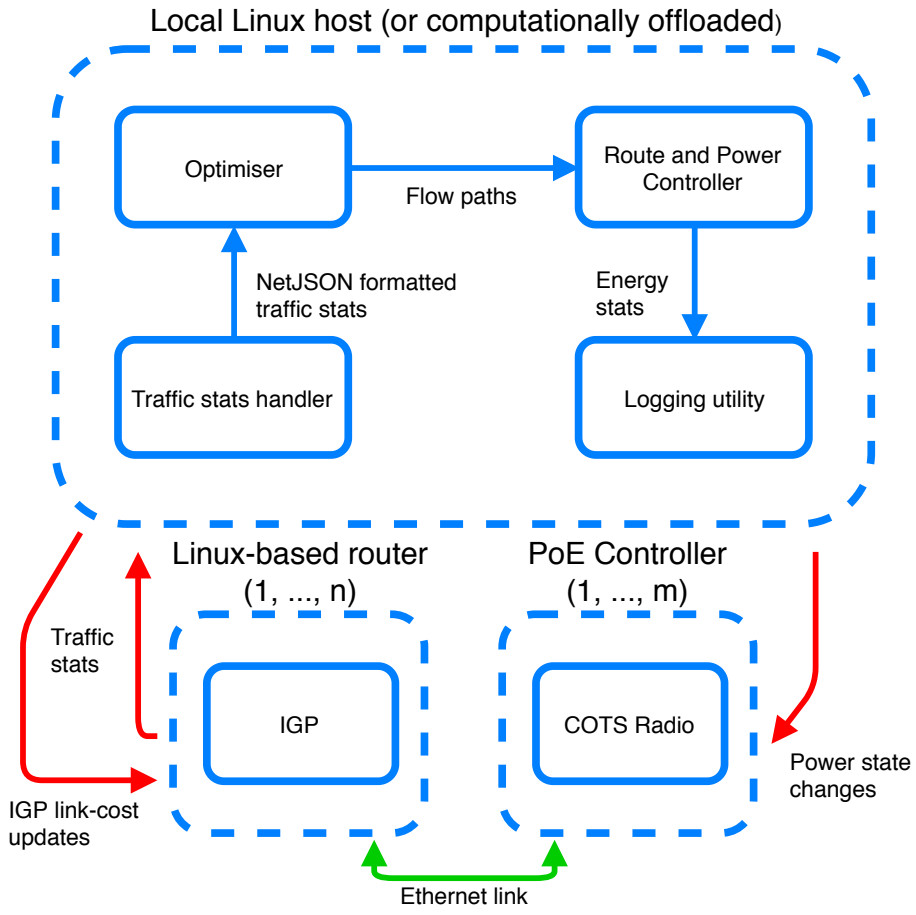

Figure 4.1: *ElasticWISP* System Overview.

4.2 *ElasticWISP* Model Formulation

Given a flow network $G = (V, E)$ with edges $(u, v) \in E$ and with capacity $c(u, v)$. There are k different commodities $K_1, K_2, ..., K_k$, where $K_i = (s_i, t_i, d_i)$. For commodity K_i, s_i is the source, t_i is the sink, and d_i is the demand. We establish that the flow of commodity i along the edge (u, v) is $f_i(u, v)$. Standard multi-commodity flow constraints are then added [48, 49]:

Edge capacity: Ensure that all flows over each link do not exceed the available capacity, that is, $\forall(u, v) \in V$,

$$\sum_{i=1}^{k} f_i(u, v) \leq c(u, v). \tag{4.1}$$

Flow conservation: Intermediary nodes cannot create or destroy commodities. That is, $\forall i \in [1, k]$,

$$\sum_{w \in V} f_i(u, w) = 0, \text{ when } u \neq s_i \text{ and } u \neq t_i. \tag{4.2}$$

Demand satisfaction: Every source and sink must send or receive an amount of flow equal to its demand. We have $\forall i \in [1, k]$,

$$\sum_{w \in V} f_i(s_i, w) = \sum_{w \in V} f_i(w, t_i) = d_i. \tag{4.3}$$

4.2.1 Energy Minimisation Constraints

We now add energy-saving constraints to the multi-commodity flow formulation. We use the notation listed in Table 4.1 to describe the energy minimisation constraints [27].

Table 4.1: Energy Minimisation Notation

Notation	***Definition***
$a(u, v)$	Power cost of the link (u, v)
$X_{u,v}$	Binary decision variable for the power state of link (u, v)
$r_i(u, v)$	Binary decision variable for indicating if commodity i uses link (u, v)

Deactivated links. Ensure that flows are restricted to only links (and radios) that are powered on. For all links (u, v) used by a given commodity i, $f_i(u, v) = 0$ when $X_{u,v} = 0$. The flow variable f will always be positive in our formulation, and thus we can write the linear constraint as

$$\sum_{i=1}^{k} f_i(u, v) \leq c(u, v) \times X_{u,v}, \tag{4.4}$$

$\forall i \in [1, k], \forall (u, v) \in E$.

Bidirectional power. On a radio pair, both radios must be turned on, irrespective of which direction network traffic is flowing, hence, we have

$$X_{u,v} = X_{v,u}, \quad \forall (u, v) \in E. \tag{4.5}$$

Flow splitting. Finally, most IGPs support single-path routing by default[1], we also add a constraint to prevent flow splitting. Additionally, we acknowledge that flow splitting is generally undesirable due to the effects of Transmission Control Protocol (TCP) packet reordering [50]. We constrain the flow of commodity i over link (u, v) to either the full demand or zero. Hence, $\forall i, \forall (u, v) \in E$,

$$f_i(u, v) = d_i \times r_i(u, v). \tag{4.6}$$

Objective function. Now that the appropriate constraints have been added, the objective function is defined, which is to minimise the overall network energy consumption:

$$\textbf{minimise} \sum_{(u,v) \in E} X_{u,v} \times a(u, v). \tag{4.7}$$

Finally, find a flow assignment that satisfies the given constraints.

[1]Equal-Cost Multi-Path (ECMP) is possible in some circumstances. The design of *SR-MPLS* is not ECMP capable at the time of writing.

4.3 Flow-Path Enforcement

ElasticWISP cannot use traditional IGPs such as OSPF and IS-IS for *explicit* flow-path enforcement. While links can be adjusted to prevent them being used before a radio pair is taken down, IGPs will take the shortest path, which in our case, may not always be the "best path", or path that the optimiser desires to be used. Although IGP flow-paths can be manipulated further than what has been shown so far, the process of doing so is generally undesirable. For *ElasticWISP*, explicit flow-path enforcement is beneficial, as we can ensure traffic flows over paths as determined by the optimiser. There are multiple criteria to achieve flow-path enforcement for use with the *ElasticWISP* optimiser:

- Accept FIB updates from the optimiser.
- Perform FIB updates without degrading network performance and user Quality of Experience (QoE).
- Program explicit flow-paths for a given traffic source and destination.
- Not be dependent on a central controller in the case of controller failure or unavailability.
- Not be dependent on the optimiser for normal use in the case of optimiser failure or unavailability.
- Be able to reconverge in the case of link-failure that is unknown to the optimiser.

4.4 Backhaul Power Control

For *ElasticWISP* to run on real WISP networks, it is necessary to expose an Application Programming Interface (API) to control the power state of backhaul radios. In the case of most backhaul radios, this means controlling DC PoE, with voltages that range between 24V–48V. However, remote

power control through APIs, or other mechanisms – other than through the switch Graphical User Interface (GUI) or Command Line Interface (CLI) – is not possible on most commercially available PoE switches. The PoE control device should be able to:

- Turn power on/off to a given radio.
- Return the on/off state of a given radio.
- Report the real-time power consumption of a given radio.
- Offer rudimentary protection against lightning and transient overvoltage.
- Expose a control API over L3 Internet Protocol (IP) connectivity.
- Provide authentication to mitigate unauthorised access.

4.5 Summary

This chapter has described the three pillars upon which the wider *ElasticWISP* scheme is built – an optimisation and control framework, explicit flow-path enforcement, and integration with specifically designed power control hardware. When used together, it is envisioned that WISPs will be able to reduce energy consumption within their backhaul networks, engineer traffic paths based on the optimisation model designed, and deploy the scheme in a real network using the application-specific hardware developed. Ultimately, these three pillars assert *ElasticWISP* as an architecture designed specifically for WISPs operating in remote and rural communities. Chapter 5 will next show how *ElasticWISP* can be applied to WISP backhaul networks that do not run speciality traffic engineering schemes.

Chapter 5

ElasticWISP Reference Implementation

This chapter presents a concise overview about how *ElasticWISP* can integrate with common IGPs, as published in the IEEE/IFIP NOMS 2020. At the core of *ElasticWISP* is a network flow model formulated as a multi-commodity flow problem that has been enhanced with energy minimisation constraints, as described in Chapter 4. To make *ElasticWISP* useful on real WISP backhaul networks, integration with commonly used dynamic routing protocols (IGPs) such as OSPF is necessary[1].

5.1 Routing Protocol Integration and Control

The *best-effort* approach to routing protocol integration is approached pragmatically. The OSPF routing daemon Quagga supports gracefully rerouting traffic when an OSPF link-cost is increased. Other OSPF and non-OSPF routing protocols generally function in this way, however Quagga was chosen due to its stability and mature codebase. Fig. 5.1 depicts the reli-

[1] If a traffic engineering scheme such as *SR-MPLS* is not in use.

able in-band control mechanism that was designed for adjusting the OSPF link-costs of routers running the Quagga OSPF daemon. ZeroMQ [51] and MessagePack [52] based zerorpc [53] was leveraged for this control mechanism.

When the *ElasticWISP* optimiser[2] outputs a subset of links, the zerorpc-based route and power controller determines which radio pairs need their link-cost and power-state changing. If radios need to be booted prior to link-costs being adjusted, the controller will do this first. Power-state changes can then happen through the use of a compatible PoE switch – should they exist in the future – or through a custom PoE control unit[3]. The controller then connects to zerorpc instances running on each router in the network, and adjusts the link-costs per radio as required. Following the completion of these state changes, traffic is gracefully rerouted.

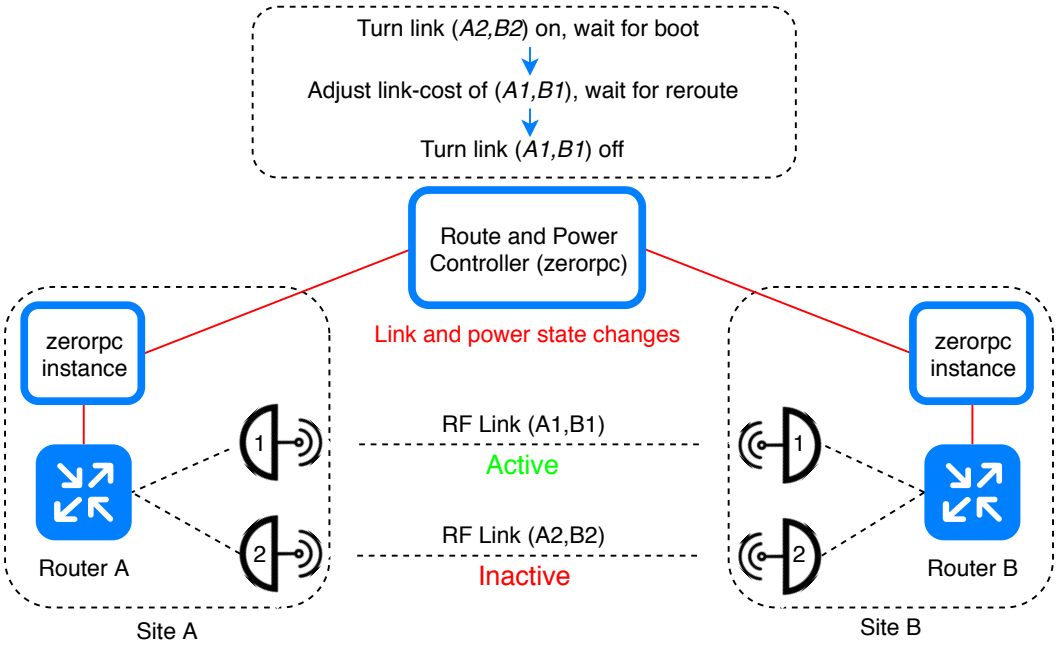

Figure 5.1: Link-Cost and Power Control Mechanism.

[2]Gurobi [54] was used for optimisation during development.
[3]See Appendix C.

5.1.1 Practical Considerations

The best-effort implementation of *ElasticWISP* is focused toward practicality and minimal effort integration with existing WISP backhaul networks. To ensure implementation viability, there can not be substantial packet loss, introduction of latency, or other QoS degradation metrics when a backhaul changeover occurs. Backhaul changeover refers to either of two key events happening:

1. The reroute of traffic between different radio pairs on the same node.
2. The reroute of traffic through a different set of nodes.

In both cases, the originating OSPF router will flood area-wide LSAs notifying other routers of the respective change. In this implementation, adjusting router interface link-costs is leveraged to gracefully reroute traffic without degrading QoS. However, this approach is not without caveats. *ElasticWISP* staggers altering link-costs to avoid any unexpected consequences of simultaneous adjustments, and the excessive LSA flooding that would occur as a result.

Consequently, a staggering delay is incurred when processing topology changes and switching the power state of radio pairs on any given node. Even on a large WISP topology with 16+ OSPF routers in operation, the time overhead per radio adjusted is under three seconds[4]. Of course, the time taken for backhaul radios to boot and reassociate to their remote peers is much longer, and must be taken into careful consideration when configuring a backhaul network to work with *ElasticWISP*[5].

Unlike link-cost adjustments, network-wide power-on state changes can

[4]Measured using the Common Open Research Emulator (CORE). Reconvergence time will vary in a physical network, but is not a concern if using *SR-MPLS*, as traffic can be rerouted on demand.

[5]Radio boot and reassociation time depends on the radio make and model; typically ranging between 30–120 seconds.

be carried out concurrently. As a result, the time taken to turn on links can be reduced to the total concurrent radio boot time. However, should they wish, a network operator can still stagger turning on links. Generally this would only be required if a large number of radios are being turned on at the same time, and if excessive LSAs become problematic. Power-off state changes can be handled in the same way, but should generally be staggered, to avoid too many LSAs from occurring simultaneously as links are taken down[6]. The backhaul changeover process is covered in greater detail in Chapter 6.12.

5.2 Summary

This chapter has briefly described the reference implementation of *ElasticWISP*. Leveraging OSPF as the IGP of choice, zerorpc for control, and Gurobi for optimisation, *ElasticWISP* is able to reroute traffic – in a best-effort manner – gracefully, without degrading network QoS. We can conclude that the best-effort approach used by *ElasticWISP* used in this chapter is appropriate for use where other traffic engineering solutions are not available. Following this summary, Chapter 6 presents the design of *SR-MPLS*, which enables WISPs to implement traffic engineering to achieve complete integration with the wider *ElasticWISP* scheme.

[6]The variation in radio boot times allows for LSAs to be flooded at different time intervals. Powering radios off happens instantaneously, resulting in LSAs being flooded at approximately the same time interval.

Chapter 6

MPLS Segment Routing Design

To enforce the optimal flow-paths being used, a lightweight, novel approach to MPLS Segment Routing (designated as *SR-MPLS*) was designed. *SR-MPLS* enforces explicit flow-paths by encoding the optimal set of backhaul links to traverse within each packet. These links are known as Adjacency Segments (Adj-SIDs), and are pushed to the packet as an MPLS label stack. Finally, note that design decisions in this chapter were made assuming OSPF is in use, as previous work showed that it is the routing protocol of choice for most WISPs [8].

SR-MPLS has specifically been designed with constrained WISP backhaul networks in mind, and operates with minimal network state. Traditional Segment Routing boasts of requiring next to no additional network state on top of what is already provided by the underlying IGP. *SR-MPLS* goes further. In an *SR-MPLS* domain, global network state (outside of what is provided by the IGP) is only present should it be explicitly enabled by a network operator, for the purpose of *loose source routing*. The following subsections describe Segment Routing paradigms, and the design rationale of *SR-MPLS*.

6.1 IGP Integration

Two approaches to IGP integration were designed:

- Label distribution with opaque LSAs, as per RFC 8402 [41].
- Unmodified IGP integration with on-demand hop-by-hop shortest-path routing. This was designed as a "poor man's" approach to Segment Routing, and is the focus of this chapter.

Traditionally, Segment Routing relies on an IGP such as OSPF or IS-IS to distribute labels. OSPF implementations such as Quagga, in their current form, do not support the extensions necessary to distribute labels used to enable Segment Routing. These extensions, such as those for OSPF, are defined in RFC 8402 [41] and RFC 8665 [55]. The second approach (*SR-MPLS*) requires no extension to IGPs, and leverages the native IGP link-state database to perform next-hop forwarding where explicit flow-paths do not exist.

In a typical Segment Routing domain, path computation can be performed by a central entity, such as the *ElasticWISP* optimiser, or in a distributed node-by-node manner, using the link-state information available from the IGP. In the case of the distributed approach, this requires that each Segment Routing capable node in the network has a global view of all other Segment Routers, as described in RFC 8402. In this regard, the process of packet forwarding is very similar to what would traditionally happen with an IGP. However, unlike traditional IP forwarding, a list of segments (MPLS headers) must be inserted into each respective packet.

Between intermediary nodes, forwarding is performed based on the MPLS label headers that have been encoded into the packet. Each MPLS header has a bottom of stack flag. If this flag is set to 0, then the last MPLS header in the packet has been reached. In this case, the MPLS header will be stripped, and the inner IP packet will be forwarded normally.

6.2 About MPLS

An MPLS header is a total of 32 bits, made up of:

- A 20-bit label[1].
- A 3-bit class of service field[2].
- A 1-bit bottom of stack flag.
- An 8-bit Time To Live (TTL) field.

Fig. 6.1 shows a visual representation of an MPLS header.

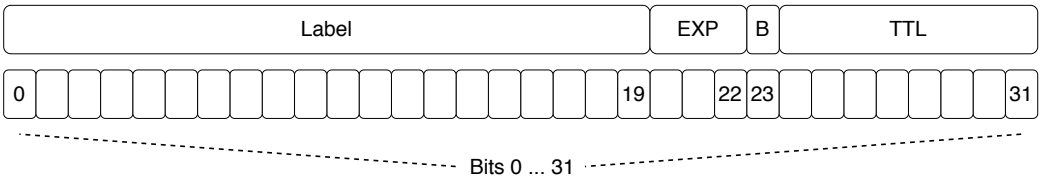

Figure 6.1: MPLS Header.

Rather than using IP next-hop information for forwarding, MPLS uses labels. MPLS originally became widespread in service provider networks due to the perceived inefficiencies of IP routing, in addition to the shortcomings of IP-based traffic engineering [56]. Despite the recent boom in SDN research, MPLS has remained an important part of service provider networks, especially so for WISPs [8].

One of the most important MPLS concepts is "label stacking". In brief, an MPLS label stack permits a variety of forwarding operations to take place based on the value – and network-wide significance – of the label headers within the stack. In the case of Segment Routing, labels in the stack can be used with both strict and loose source routing paradigms. For clarity, Fig. 6.2 depicts how an MPLS label stack fits within an Ethernet frame.

[1]Ranging from 0 – 1048575.
[2]Also known as the experimental (EXP) field.

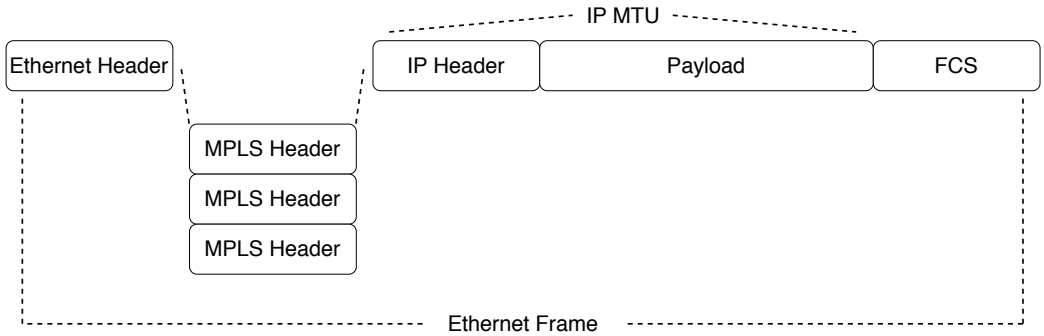

Figure 6.2: MPLS Packet.

6.3 *SR-MPLS* Terminology

A brief overview of MPLS and Segment Routing terminology will assist in reading about the design of *SR-MPLS*. Firstly, MPLS has three key label operations:

- Push – The action taken when an MPLS label is inserted into a packet.
- Pop – The action taken when an MPLS label is removed from a packet.
- Swap – The action taken when an MPLS label is removed from a packet, and replaced with a new MPLS label.

Secondly, MPLS-based Segment Routing uses several keywords to identify network Segment Identifiers (SIDs) [44]:

- Adjacency SIDs – Only known locally between adjacent *SR-MPLS* routers.
- Node SIDs – Global, used to identify the \32 prefix of an *SR-MPLS* router. Cannot be used without a means of distribution, such as with IGP extensions.
- Prefix SID – Global, associated with an IP prefix. Cannot be used without a means of distribution, such as with IGP extensions.

The design of *SR-MPLS* does not utilise prefix SIDs. Instead, *SR-MPLS* uses adjacency SIDs where possible, to keep network state at a minimum, while node SIDs are used only if required by the network operator.

Additionally, general MPLS terminology can be applied to *SR-MPLS*:

- LSP – a unidirectional tunnel between a pair of *SR-MPLS* routers across a larger *SR-MPLS* domain.
- Label Edge Router (LER) – makes the initial path selection, and encodes the explicit adjacency SID label stack. Encapsulates IP packets that will traverse the LSP.
- Label Switching Router (LSR) – what is alternatively called a "transit node". An LSR only performs label-switching in the middle of an LSP.
- Egress node – the final *SR-MPLS* router in a LSP, which "pops" the final MPLS header.

6.4 IPv4 and IPv6 MPLS Encapsulation

Additionally, while Internet Protocol version 6 (IPv6) Segment Routing (SRv6) exists, *SR-MPLS* can be used for either Internet Protocol version 4 (IPv4) or IPv6 transportation (of L3 packets). The design of *SR-MPLS* is focused toward IPv4, as SRv6 is a well established, promising area of research. Of course, designing a dual-stack IPv4 and IPv6 scheme presents some challenges. For one, easily determining what EtherType[3] the inner packet requires is straightforward if you are dealing with a single IP protocol, i.e., IPv4 **or** IPv6. However, a dual-stack scheme requires a means of determining what inner IP header follows an MPLS header.

[3]A data link layer (or L2) Ethernet header contains a 16-bit field that represents the type of payload encapsulated within the Ethernet frame.

Determining the next-header is necessary prior to forwarding an inner L3 packet to its destination. If the incorrect EtherType (derived from the next-header) is set, the packet will be dropped before, or at, its destination. Note that when an IPv4 or IPv6 packet is encapsulated, the original EtherType is changed (from IPv4 or IPv6) to MPLS, and forwarded through the *SR-MPLS* domain. Leaving the EtherType set as MPLS after the final MPLS label has been popped will also ultimately result in the packet being dropped.

In an IPv4 and IPv6 dual-stack environment, several approaches could be taken to determine the correct next-header (and subsequently set the correct EtherType during decapsulation):

- Using the MPLS class of service bit-field – typically used for QoS purposes – to represent IPv4 or IPv6 EtherTypes (3-bit field).

- Using proposed MPLS-extensions that include an 8-bit field for a next-header [57].

- Using specific, separate label ranges for IPv4 and IPv6 addresses.

Alternatively, a network operator could simply use SRv6 for their IPv6 traffic engineering needs.

6.5 MPLS Forwarding

The typical operation of an *SR-MPLS* router follows a *strict source routing* paradigm. If instructions to match IP traffic to an explicit flow-path exist in the FIB – also referred to as the Label Forwarding Information Base (LFIB) – of an *SR-MPLS* router, traffic destined to the match address – e.g., *10.1.1.0/24* – will be encoded with an MPLS label stack. The label stack consists of an ordered set of MPLS labels, which represent segments in the *SR-MPLS* domain. As mentioned in Subsection 6.3, these can be adjacency SIDs, or node SIDs. The subsequent subsections detail the design of *SR-MPLS* modes of operation in terms of L3 functionality.

Also note that in MPLS networks, there are typically two approaches to popping MPLS labels (or formally, terminating an LSP):

- Implicit Null – also known as Penultimate Hop Popping (PHP).
- Explicit Null.

The PHP approach pops the last MPLS label on the second-to-last hop (the penultimate hop). PHP is used to avoid the final router (ultimate hop) having to perform two operations: (i) popping the remaining MPLS label, and (ii), performing an IP lookup. In contrast, the Explicit Null approach requires that MPLS labels are used until the ultimate hop is reached. Upon decapsulation of the final MPLS header, an IP lookup must be performed in order to forward the underlying packet. In an *SR-MPLS* domain, the Explicit Null approach is used, however, PHP support could be added in the future.

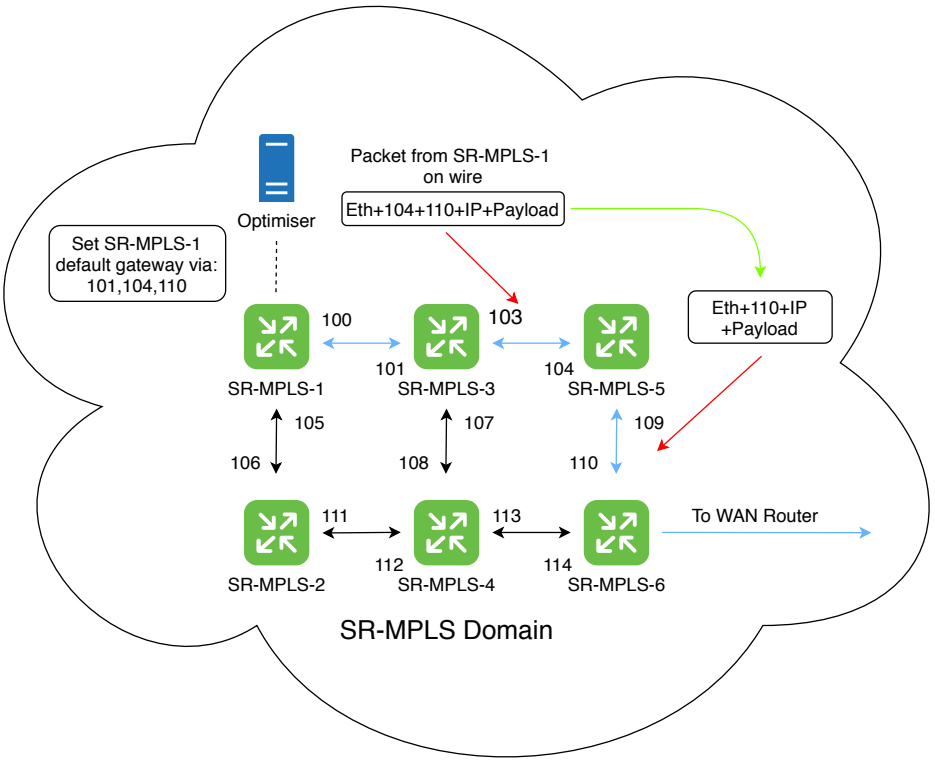

Figure 6.3: Strict Source Routing in an *SR-MPLS* Domain.

6.5.1 Strict Source Routing

In the typical – *ElasticWISP* integrated, minimum network-state – configuration of an *SR-MPLS* domain, explicit flow-paths are used. Using explicit flow-paths means that the entire path that the packet needs to traverse is encoded within the packet. The explicit flow-path approach is also known as *strict source routing*. When explicit flow-paths are used, the ordered list of paths *must* be formed with adjacency SIDs. The resulting MPLS label stack directs *SR-MPLS* routers to send MPLS packets over each adjacency SID, enabling the network operator to control specifically what backhaul radio

interface(s) are used for forwarding traffic. Fig. 6.3 shows the expected operation of an *SR-MPLS* router performing *strict source routing*.

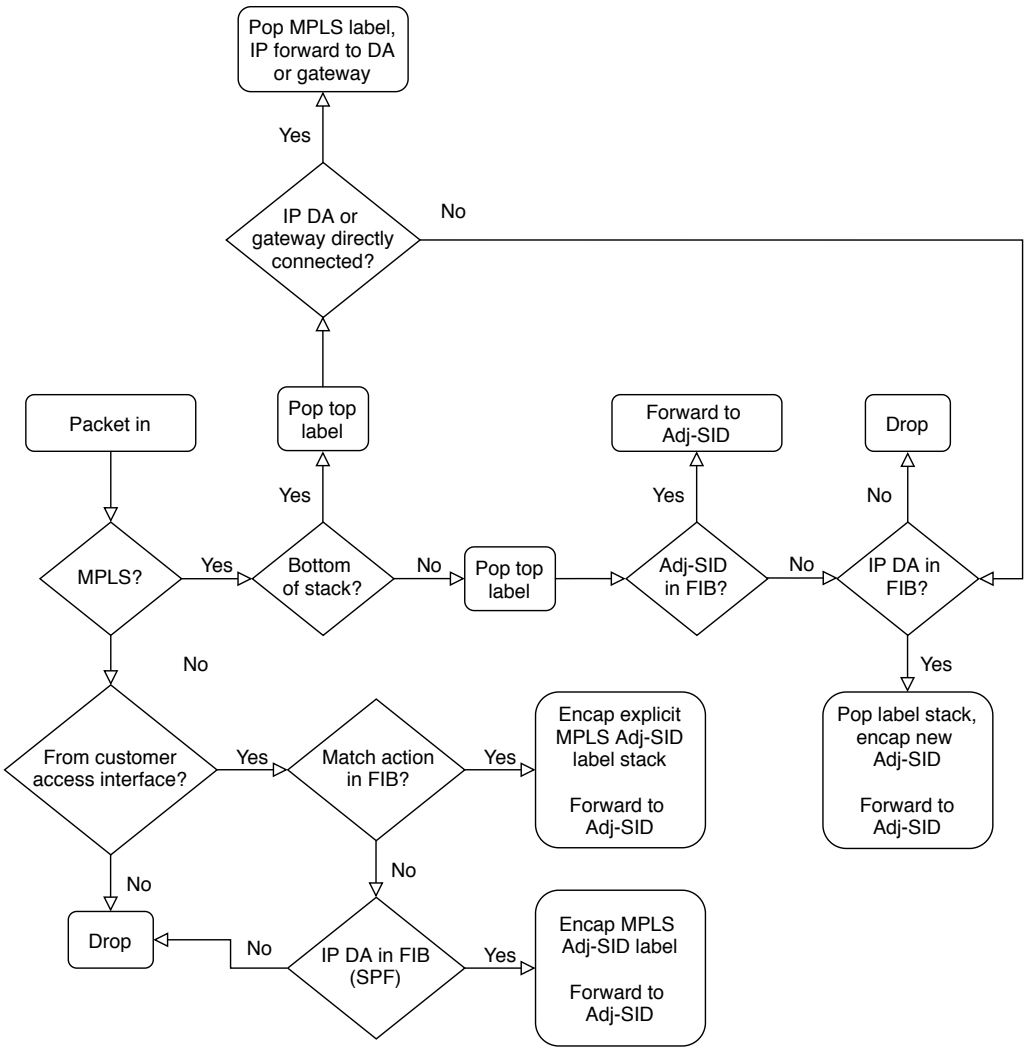

Figure 6.4: Flow Diagram of Strict Source Routing in an *SR-MPLS* Domain (L3 Operation).

6.5.2 Loose Source Routing

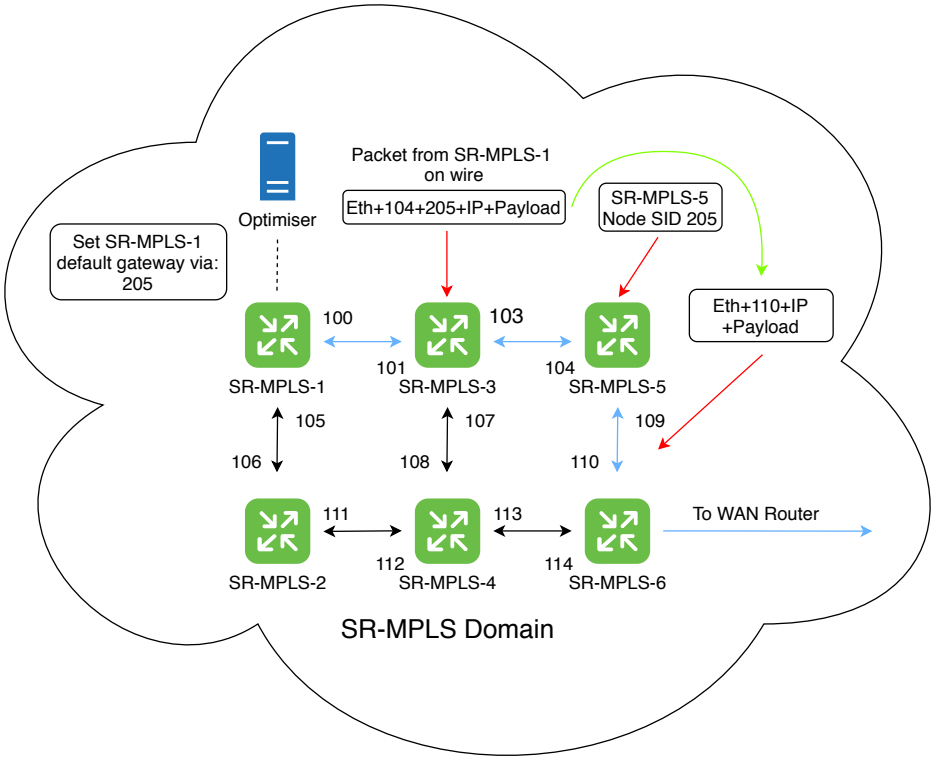

Figure 6.5: Loose Source Routing in an *SR-MPLS* Domain.

Unlike *strict source routing*, the use of *loose source routing* does not require explicitly defining the entire set of segments that packets must traverse. However, to enable *loose source routing* support, *SR-MPLS* routers must be able to learn the node SID from every other *SR-MPLS* router in the network. Generally, the node SID would be distributed using the underlying IGP. As *SR-MPLS* avoids modifying the IGP by design, an appropriate network-capable message queue – such as ZeroMQ – is used to distribute the node SID to devices that request it. The advantage of using a message queue

over a "roll-your-own" sockets-based approach is that more attention can be given to other core areas of *SR-MPLS*. Essentially, the message queue ensures reliable inter-node exchange can take place, without requiring an application specific protocol design to be developed. Routing protocols such as Facebook's Open/R have also leveraged the message-queue approach for state exchange between nodes [58].

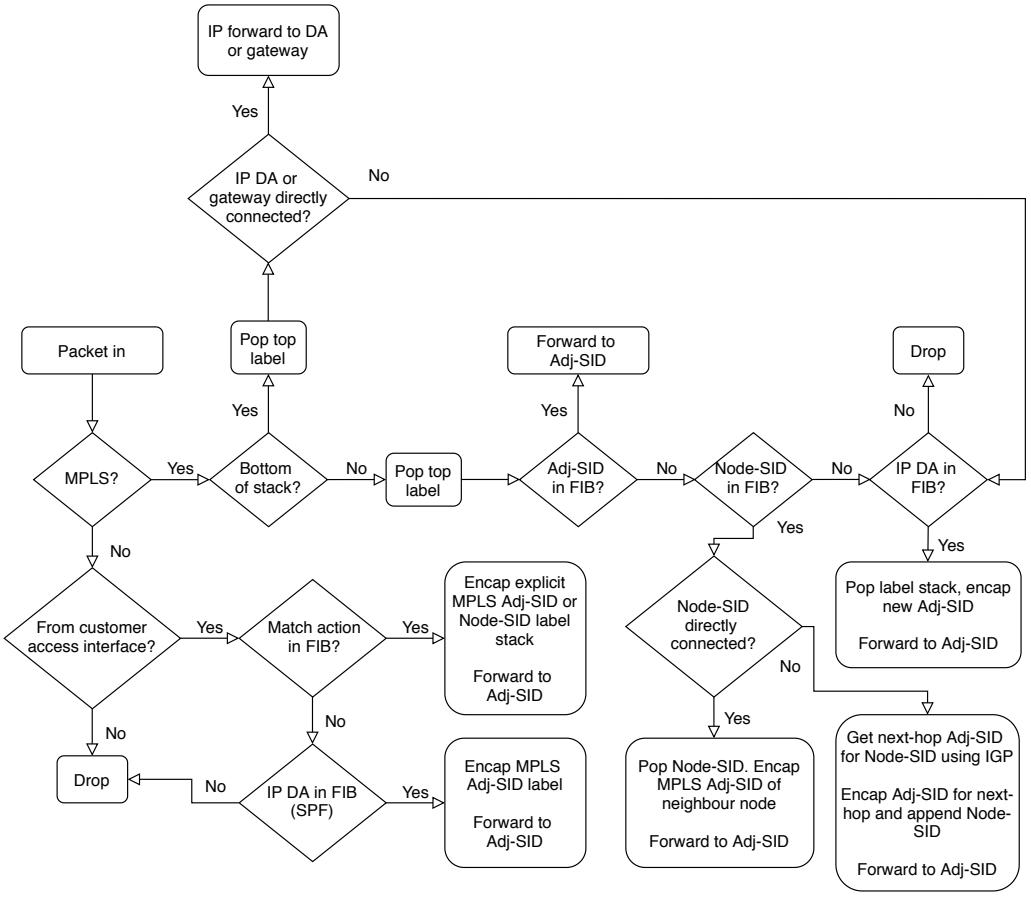

Figure 6.6: Flow Diagram of Loose Source Routing in an *SR-MPLS* Domain (L3 Operation).

It should also be noted that by default, *SR-MPLS* routers are not designed to perform *loose source routing* – at least when integrated with the *ElasticWISP*

optimiser – but *loose source routing* functionality is accounted for, should a network operator desire to use it. Fig. 6.5 shows the expected operation of an *SR-MPLS* router performing *loose source routing*.

6.5.3 Shortest Path Forwarding

Shortest path forwarding is used in several circumstances:

- When an explicit steering policy has not been installed in the *SR-MPLS* FIB.
- When *loose source routing* is being used[4].
- When a link failure occurs during normal *strict source routing* operation.

Also note that shortest path forwarding also does not require node SIDs to be enabled. When an adjacency SID is unavailable, or no steering policy has been installed in the *SR-MPLS* FIB, the packet is forwarded based on its IP destination address. To achieve this, and forward the packet over the shortest path, the IGP link-state database is queried. The database provides the best outgoing interface to forward the packet, which is then used to find an adjacency SID. The adjacency SID – which is an MPLS label – is then pushed onto the MPLS label header, and the packet is forwarded to the adjacent *SR-MPLS* router. The procedure is then repeated until the packet reaches its destination.

The shortest path forwarding mechanism is also important in the case of a link failure. Should a link go down while traffic flows are being forwarded over it – e.g., using the hop-by-hop *explicit source routing* approach – the *SR-MPLS* routers will detect that the neighbouring adjacency SID has been removed from the FIB. Subsequently, the *SR-MPLS* routers will pop all of the MPLS labels – i.e., the adjacency SIDs – in the MPLS stack. After

[4]Node SID distribution must be enabled.

performing a lookup using the IGP link-state database, a new adjacency SID is pushed onto the packet. Until the link failure is resolved, shortest path forwarding will continue. Additional details about how path-failure detection occurs is detailed in Subsection 6.8.

6.6 Control Plane

The design of a traditional Segment Routing control plane is left at the discretion of the network operator. In the case of *SR-MPLS* capable routers, the control plane is responsible for installing FIB entries. These FIB entries are the output of the *ElasticWISP* optimiser, and are necessary for explicit flow-path enforcement to take place. Of course, in addition to the output of the *ElasticWISP* optimiser, the *SR-MPLS* scheme is reliant on the underlying IGP control plane for normal operations to take place. While it would be possible to run an *SR-MPLS* router without an IGP, static routing would be required, and the network would lack fault tolerance.

The installation of FIB entries on *SR-MPLS* routers could take place with any number of control plane protocols. As we depend on an IGP for normal operation, the message queue approach mentioned earlier can also be leveraged for reliable – and in many cases, programming language agnostic – message delivery. An in-band control plane approach that leverages a message queue can coexist with MPLS-based *SR-MPLS* routers, as in addition to the *SR-MPLS* domain, there is still a functional IPv4 and/or IPv6 underlay network, meaning control packets pass through the network without being encapsulated.

Most modern message queues support a generic client–server model, in addition to the classical pub–sub and push–pull approaches. While other communication schemes are possible, the generic client–server model is used to establish communications between *SR-MPLS* routers and the *ElasticWISP* optimiser. In such a configuration, *SR-MPLS* routers act as servers

listening for requests from the *ElasticWISP* optimiser. The optimiser can then install new FIB entries, update them, delete them, and query them as required.

6.7 Neighbour Label Resolution

To minimise the amount of manual configuration required for setting up an *SR-MPLS* domain, a basic Media Access Control (MAC) address to MPLS label discovery mechanism was designed. Once IGP adjacencies have been established, an appropriate message queue is used to setup bidirectional communication between the pairs of adjacent *SR-MPLS* routers, solely for the purpose of label exchange. Once the label exchange takes place, *SR-MPLS* routers are able to begin forwarding MPLS packets. Following the initial label exchange, no additional communication is necessary between the neighbours. Should a failure occur, such as the reboot of an *SR-MPLS* router, the recovering node will reconnect to its neighbour following an adjacency being formed by the IGP, and subsequently regain the label associated with the adjacent interface MAC address.

Incorporating a message queue into the design of *SR-MPLS* means that label exchange and installation is reliable, and a range of authentication options are available, should they be required by the network operator, e.g., CURVE [59] if ZeroMQ is used as the message queue of choice. Even if a failure occurs in the message queue-based label installation mechanism, *SR-MPLS* routers *can* continue to forward packets normally, however must fall back to using the hop-by-hop method discussed earlier. Note that the MAC address of the neighbour IP is still found using the Address Resolution Protocol (ARP). The neighbour label resolution mechanism described here is solely for the purpose of learning adjacent labels. In the future this exchange mechanism could be extended to become a separate protocol, perhaps appropriately named the Label Resolution Protocol (LRP).

6.8 Bidirectional Forwarding Detection

In traditional Segment Routing, failure detection is handled completely by the IGP. Should a failure occur, the IGP can recompute the shortest path using its global view of the Segment Routing domain. In an IP network running an IGP such as OSPF, the same process occurs. When a failure happens, the IGP can recompute the shortest path as it has a global view of other routers in the network. In this "poor man's" approach to Segment Routing, each node in the network does not have a global view of segments in use across the network. Because of this, failure mitigation must be handled differently.

To detect link failure – ideally even faster than the IGP in use can – Bidirectional Forwarding Detection (BFD) is used to detect if an explicit flow-path is up or down. When the *ElasticWISP* optimiser computes explicit flow-paths, it installs them in the FIB of the *SR-MPLS* routers across the network. Each specific IP or IP subnetwork with explicit flow-paths will have a FIB entry. For unique source–destination pairs, BFD sessions are then established. An MPLS packet is then generated, with the explicit flow-path encoded into the packet. If no response is seen in the defined period, the link is deemed to be down.

As we consider if the entire explicit path (the LSP) is up or not, we do not get a "fine-grained" view of where link failure exists between pairs of *SR-MPLS* routers. While possible with BFD, detecting failure between adjacent *SR-MPLS* routers is performed automatically by the IGP in use[5]. Note that MPLS-specific BFD is defined in RFC 5884 [60]. The concept of using BFD for multihop paths – as intended for *SR-MPLS* routers – is also well established, and is defined in RFC 5883 [61].

Should a failure be detected in an explicit flow-path, the originating *SR-*

[5]Link-failure detection is typically handled by the IGP in a traditional Segment Routing domain.

MPLS router reverts back to shortest-path forwarding. Next, the *SR-MPLS* router alerts the optimiser that the explicit flow-path is down, and requests new explicit paths be installed. The response time between the *SR-MPLS* router reverting to shortest-path forwarding and then having a FIB update installed will depend on operator-defined settings, as it is generally desirable to limit how frequently the *ElasticWISP* optimiser is run within quick succession.

In addition, the BFD mechanism may detect link-failure before the IGP does. As *ElasticWISP* builds the topology used for optimisation from an IGP's link-state database, requesting new routes before the IGP can react to failure may result in the same routes being installed in *SR-MPLS* FIBs. As a result, limiting frequent runs of the optimiser should be done with careful consideration in respect to the configuration of the IGPs dead-interval timer.

It should also be noted that BFD is not intended to be run between adjacent pairs of *SR-MPLS* routers, although this is also possible, and is commonplace practice between adjacent OSPF routers. In this configuration, BFD monitors the forwarding plane availability between the adjacent routers. In the case of OSPF, *Hello* messages are used to ensure links are working as intended. BFD is used alongside this mechanism as it can be configured with much lower-latency exchange intervals, meaning link failure can be detected significantly faster [62].

Finally, the use of BFD with *SR-MPLS* routers can take two approaches:

- With an explicit return path encoded into each BFD packet.
- Using a return path as determined by the destination node.

These two modes of operation exist as in some cases the return path could be different to the path that was originally used to reach the destination. In most cases, the return path should be determined by the destination *SR-MPLS* router, as this will verify bidirectional connectivity as packets

would normally flow in either direction from source to destination *SR-MPLS* routers. However, the flexibility exists to enable network operators to probe specific flow-paths should they wish.

Of course, should explicit return paths be defined, the source router must be aware of the position of other *SR-MPLS* routers within the network. As adjacency SIDs are only shared between adjacent neighbours, this approach would generally require the use of node SIDs, subsequently introducing additional network state. Alternatively, a device, such as the *ElasticWISP* optimiser, which has global network knowledge, could be delegated to probing bidirectional connectivity between *SR-MPLS* routers. Leveraging the *ElasticWISP* optimiser for this purpose will be a topic of future exploration.

6.9 Time To Live Handling

When an IP packet is encapsulated by an *SR-MPLS* router, the TTL field of the IP packet is copied into the TTL field of the top label in the MPLS label stack, i.e., the stack of MPLS SIDs. The top MPLS label header TTL is decremented each time an MPLS packet is passed through an *SR-MPLS* router. The top label is popped, and the decremented TTL field is copied to the next label header in the label stack. When an MPLS label header with a bottom of stack flag is encountered – and an IP address entry exists in the *SR-MPLS* FIB – the MPLS TTL is then copied into the IP header TTL field of the outgoing packet.

6.10 MTU Considerations

Networks that leverage Segment Routing, should it be SRv6 or *SR-MPLS*, must take the lowest Maximum Transmission Unit (MTU), that network equipment within the domain supports, into careful consideration. Each

MPLS label header – or MPLS SID – is 4 Bytes, or 32 bits. In comparison, an SRv6 SID is 16 Bytes, or 128 bits. The MTU used between devices in the Segment Routing domain should be coordinated. Coordination should be completed in respect to the network topology, planned future expansion of the network, and the maximum segment depth – i.e., the number of SIDs – that will be encoded into packets within the domain. Unexpected situations will result if MTUs are configured incorrectly across the domain – or worse, the network will simply not work. In addition, if the maximum segment depth is high, and the MTU is set at the default 1500 Bytes, outgoing packets could be dropped if the Network Interface Controller (NIC) does not support jumbo frames.

6.11 MPLS Services

The core focus of *SR-MPLS* is to enable simple traffic engineering for layer 3 (IP) traffic. In MPLS terminology, the behaviour of *SR-MPLS* routers is what is referred to as a Layer 3 Virtual Private Network (L3VPN), also known as a Virtual Private Routed Network (VPRN). However, while not discussed in detail, it is also possible to utilise layer 2 MPLS services such as point-to-point (pseudowire) and Virtual Private LAN Service (VPLS) Layer 2 Virtual Private Networks (L2VPNs).

While possible, additional considerations need to be made regarding failure mitigation when using L2 services. In an *SR-MPLS* domain, failure mitigation leverages the IGP, and can reroute traffic based on L3 addresses. Should an entire L2 frame be encapsulated, and failure happen during transit, an *SR-MPLS* router can look at the inner L3 address to determine where to forward the packet – but this approach may not always result in desirable results.

Consider a L2VPN setup between two customer sites created by a WISP. The traffic being forwarded over the L2VPN may not be ever destined for

the public Internet. Private IP space – that the IGP is unaware of – could also be in use. Should a failure happen in transit, or should a link in an explicit flow-path fail, the hop-by-hop approach, and, to an extent, the *loose source routing* approach would not be suitable for mitigation. As the inner L3 address, in this example, is not destined for a prefix being advertised by OSPF, or the Internet, it provides no usable information to determine mitigation paths for forwarding.

Alternatively, a network operator could define a path for a L2VPN to traverse which **must** include the Node-SID of the destination node. Should a failure happen during transit, the Node-SID can still be used to determine an alternative hop-by-hop path to be taken.

6.12 Backhaul Changeover

One of the most important features of *SR-MPLS* is the ability to reroute traffic on demand, without having to wait for IGP reconvergence to take place. When the *ElasticWISP* optimiser installs FIB entries in *SR-MPLS* routers, the flow-paths being enforced can be changed immediately. In contrast, the *ElasticWISP* reference implementation must wait for IGP reconvergence, or risk packet loss, due to prematurely powering down backhaul radios prior to traffic being routed over the intended backhaul pair. However, when backhaul radios in a link are powered down, network reconvergence will still occur, irrespective of the scheme being used.

6.13 *SR-MPLS* Complexity

Let n be the number of *SR-MPLS* routers, k be the number of commodities originating at an *SR-MPLS* router, m be the number of adjacent neighbours of an *SR-MPLS* router, and d be the number of unidirectional links in an *SR-MPLS* domain. Firstly, adjacency SIDs must be setup, requiring $\mathcal{O}(n)$

SR-MPLS routers to be configured. The routing state of an *SR-MPLS* router using *explicit source routing* can be given as $\mathcal{O}(m)$. Resultantly, the routing state of an *SR-MPLS* router is minimal, as it is only necessary to store neighbour adjacency SID information in the FIB.

Should *explicit source routing* be used, the routing state is $\mathcal{O}(n+m)$, as each *SR-MPLS* router must be aware of the node SID of every other *SR-MPLS* router in the network, in addition to being aware of adjacency SIDs. When operating an *SR-MPLS* network, the complexity of running BFD on a per-router basis can be given as $\mathcal{O}(k)$, as a BFD session for each commodity k must be established to a remote peer. Finally, we expect that there will typically not be more than $\mathcal{O}(n^2)$ unidirectional links (d) in a given *SR-MPLS* domain.

6.14 Summary

This chapter has shown that *SR-MPLS* promises to radically simplify the way that WISPs perform traffic engineering in their backhaul networks. The design of *SR-MPLS* requires minimal routing state, and can integrate with IGPs without them requiring any further extensions, as is necessary with traditional Segment Routing. Most importantly, the design of *SR-MPLS* enables explicit flow-paths as computed by the *ElasticWISP* optimiser to be enforced, enabling the overall *ElasticWISP* scheme to function as intended. Next, Chapter 7 presents the power control hardware designed and implemented to enable real-world application of *ElasticWISP* and *SR-MPLS*.

Chapter 7

Backhaul Power Control

To enable the control of PoE devices, a power control "cape" was designed for the BeagleBone Green [63] Single-Board Computer (SBC). The cape includes 24V and 48V inputs, making it useful for low-cost 24V radios as well as higher-cost radios such as the Ubiquiti airFiber series. The BeagleBone Green was selected as it is open hardware, includes 10/100 Mbps Ethernet connectivity, has numerous General Purpose Input/Output (GPIO) pins, and includes 7 analogue inputs. Furthermore, these analogue inputs mean that no external Analogue to Digital Converter (ADC) is necessary, which is commonplace in many other SBCs, such as all variants of the Raspberry Pi [64]. Adding an external ADC to the already space constrained Printed Circuit Board (PCB) would mean that there is less room for the required power switching electronics.

In this case, analogue inputs are required for two reasons:

- To read the 24V and 48V output voltages.
- To read the current sense voltage of each high-side switch.

Subsequent sections in this chapter describe the design decisions made for the power control cape.

7.1 Cape Features

Unlike the BeagleBone Black, the BeagleBone Green has no barrel jack to power the board. Instead, the board is powered entirely through USB, at 5V. Alternatively, it is possible to power the BeagleBone Green by supplying 5V to the *VDD 5V* pin. In practice, this board should be powered by batteries that reside at a remote site used by WISPs, and powered with a 12V–24V input supply. To accommodate for this, a 5V switching regulator – with an ultra-wide 9V–72V input range – is included on the cape.

Power switching on the cape is performed using two Infineon [65] high-side switches. The high-side switch of choice – the BTT6050-1ERA [66] – supports voltages ranging from 5V–48V, with a nominal current of 4.5A, making it an ideal fit for powering both low-power and high-power backhaul radios. The BTT60X0 series high-side switches are also microcontroller friendly, and can be controlled with 3.3V and 5V logic inputs. The BTT60X0 series also have "current sense" capability, which eliminates the need for an external current sense amplifier, should we measure the current being used by a given radio.

Finally, the cape uses a fundamental circuit – a voltage divider – to measure the voltage of both the 24V and 48V outputs[1]. The voltage measurement requires either of the high-side switches to be *on*, as we wish to prevent back-powering the BeagleBone Green through the analogue input pins should an unexpected failure occur. Back-powering could also otherwise happen in the event that the 24V input is inactive, but the 48V input is live. This would mean that the BeagleBone is not receiving power from the 5V regulator, and would instead receive voltage from the voltage divider. As either high-side switch having an *on* state depends on the BeagleBone receiving power through its 5V regulator, these issues are eliminated. This approach is aligned with BeagleBone recommendations [67].

[1] As opposed to directly measuring the input voltages.

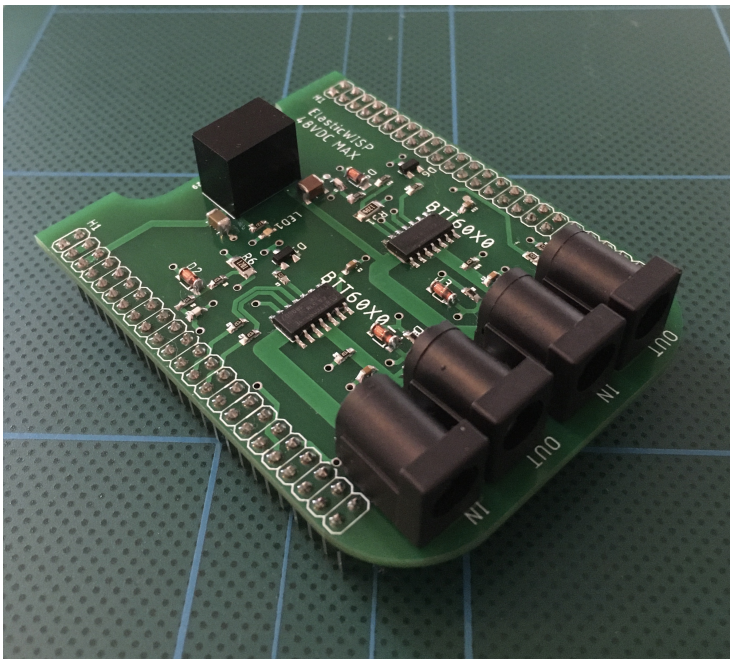

Figure 7.1: Backhaul Radio Power Control Board.

The assembled power control cape is shown in Fig. 7.1.

7.2 PoE Injector Design

To facilitate the operation of the power control mechanism, it is necessary to inject 24V/48V DC over the Alternating Current (AC) Ethernet signal used by the target backhaul radios. Typically, this is done using PoE switches. Commercial switches could be used, but managed, GbE capable PoE switches with DC input are expensive, and a product that enables external control – other than through a GUI – could not be identified. Ideally, *ElasticWISP* needs a PoE switch to expose an external control API that it can use to turn radios on and off.

As a suitable commercial product could not be identified, the power control cape was developed. It is important to reiterate that the power control

cape can only switch power. Due to this, PoE injectors are still necessary. Commercial PoE injectors are readily available, but are also often costly, especially when we must support up to 60 W to safely power the airFiber radios. Given the long-term desired integration with other components of the *ElasticWISP* architecture, these commercial injectors also may not be suitable. Consequently, two GbE-capable PoE injector designs were created, one suitable for up to 0.5A at 24V/48V, and another improved design that supports 2.5A at 24V/48V[2].

7.2.1 Design 1

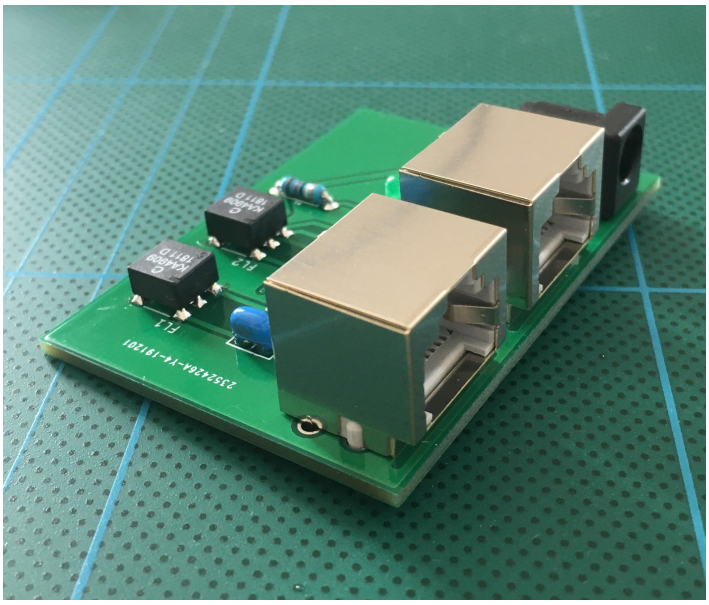

Figure 7.2: Design 1. Low-Cost Gigabit PoE Injector.

The purpose of the first PoE injector design was to serve as a low-cost Proof of Concept (PoC). In brief, two Coilcraft [68] KA4909-AL [69] bias injection chokes were used to inject a common-mode DC voltage into a Gigabit Ethernet signal. If necessary, the injector can also act as a PoE "splitter",

[2]Two-pair operation at 1.25A, four-pair operation at 2.5A.

extracting power using the centre-tap of the injection choke, allowing the device to be used with non-PoE devices, while still supporting Gigabit Ethernet capability. In this mode of operation, a high-pass filter is used to prevent DC passing back into devices that are not PoE compliant. This approach is typically seen in injecting DC power over high-speed serial applications [69]. The resulting implementation is shown in Fig. 7.2. This design was later improved by using a single Coilcraft FA2536-AL [70] in place of the two Coilcraft KA4909-AL chokes, as shown in Fig. 7.3. Unlike the KA4909-AL, the FA2536-AL is specifically designed to exceed the return loss requirements of Gigabit Ethernet.

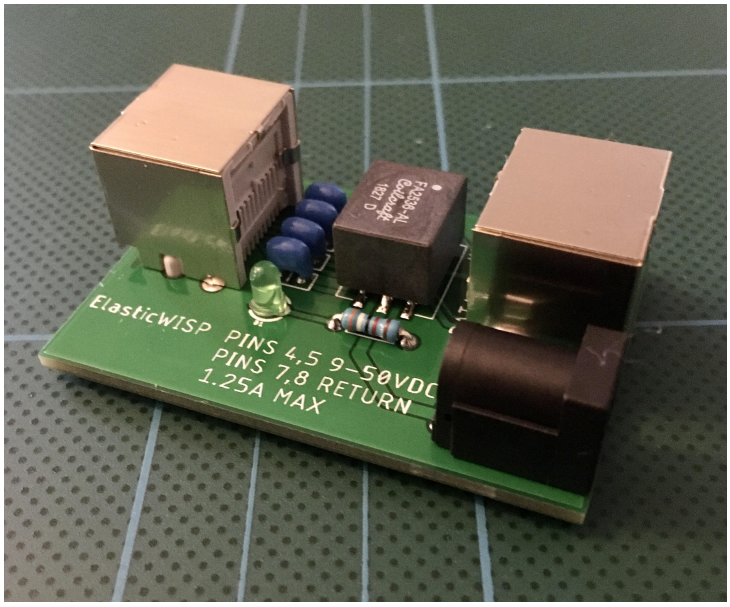

Figure 7.3: Design 1 Revision 1. Low-Cost Gigabit PoE Injector.

7.2.2 Design 2

The second design uses a Coilcraft ETH1-460L [71] high-power PoE signal path transformer, specifically designed to meet the requirements of the IEEE 802.3at-2009 and IEEE 802.3.bt Type 4 standards for PoE+ and PoE++,

making the injector suitable for use with radios that have very high power consumption. Practically, this means the power through the ETH1-460L can be up to 120 W. High-power transmission is achieved by sending power over all four Ethernet pairs, as opposed to over just two pairs as shown in the first design. Two-pair operation is also possible, as the injector requires two jumpers be connected for full four-pair operation. It was necessary to create this design as high-power radios such as as the airFiber 5 series require power over all four Ethernet pairs.

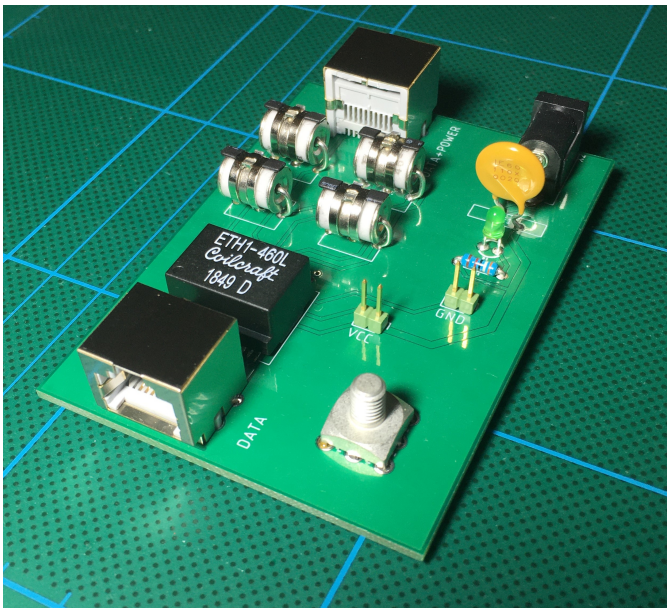

Figure 7.4: Design 2. High-Power Gigabit PoE Injector.

In addition to power injection, the second design includes Gas Discharge Tubes (GDTs) for lightning protection, a protective earth connector, and a resettable fuse for protection against overcurrent. Finally, it should be noted that both PoE injector designs are passive only, and will damage or destroy non-passive PoE compliant devices. This means that power is always being sent over (a minimum of) two Ethernet pairs. In the case of both injectors, this means a common-mode voltage is injected over the Ethernet twisted

pairs (4,5) and (7,8), which is also known as Mode B operation [72]. When the four-pair "enable jumpers" are connected on the second design, power is also injected over the Ethernet twisted pairs (1,2) and (3,6). This approach is known as Mode A operation [72]. When all Ethernet twisted pairs are utilised, the operation can be referred to as PoE four-pair mode.

The choice to design passive injectors was due to the backhaul radios targeted during the development of *ElasticWISP*. Particularly, Ubiquiti backhaul radios typically only support passive PoE. For high-end microwave radios, active PoE would likely be needed. Unfortunately, time constraints rendered further designs a subject of future work.

7.3 Summary

DC PoE switches can be expensive, closed source, and generally unsuitable for adaption to work with *ElasticWISP*. This chapter has presented two robust PoE injector designs: (i) a low-cost, low-power model (up to 30 W), and (ii), a higher-cost, high-power model (up to 120 W). To leverage these PoE injectors, a power control "cape" for the BeagleBone Green was designed. Together, the power control "cape" and PoE injectors enable WISPs to disaggregate backhaul power control from traditional PoE switches, subsequently enabling practical application of the *ElasticWISP* scheme. As the design of the three pillars within the wider *ElasticWISP* architecture has now been concluded, Chapter 8 presents an evaluation of *ElasticWISP* and *SR-MPLS*.

Chapter 8

Validation

In this chapter, an evaluation of the *ElasticWISP* and *SR-MPLS* schemes is provided. Firstly, a brief description of the *ElasticWISP* evaluation setup is given, followed by an overview of how traffic was captured and subsequently *"played back"* to validate the optimisation scheme. QoS preservation using the *best effort* OSPF link cost-adjustment approach is then evaluated, followed by an analysis of the energy-saving potential of *ElasticWISP*. Finally, performance of the *SR-MPLS* Segment Routing scheme is validated.

8.1 *ElasticWISP* Evaluation Setup

The formal model of *ElasticWISP*, presented in Section 4.2, was implemented using the state-of-the-art optimisation solver, Gurobi. For Gurobi to solve the optimisation problem, it must have an input network topology, the capacity and energy consumption of links, and finally, a traffic matrix. The Venture Networks backhaul network topology was first defined using NetJSON [73]. The NetJSON formatted topology was then extended by adding radio capacity and power consumption attributes. Next, a topology parser for Gurobi was written in Python. The parser takes the NetJSON file,

and processes it ready for optimisation using the Gurobi Python interface, GurobiPy.

The output of the Gurobi optimiser is a subset of the original network topology, and a set of optimal flow-paths that are used later for MPLS FIB installation with *SR-MPLS* capable routers. The topology subset is passed to the zerorpc based link-cost and power controller, which subsequently turns on and off backhaul links as appropriate. The time it takes Gurobi to output a subset of a given topology depends on the size of that topology. As most WISP topologies are comparatively small compared to traditional ISPs, the optimisation time is of negligible concern, and topology subsets can be computed rapidly, even on low-end hardware. In the evaluation of *ElasticWISP*, the computation time for the Venture Networks topology never exceeded 10 milliseconds. However, in very large networks, evaluations of optimiser performance would need to be made. Topically, the computational complexity of finding an optimal flow assignment for integer flows alone is NP-complete [74].

In extremely constrained environments, Gurobi optimisation tasks can also be offloaded to an external compute server [75]. Given this, the *ElasticWISP* architecture could be extended to support computational offloading. It should also be noted that Gurobi claims to be up to 50% faster than other state-of-the-art commercial solvers [76]. The substitution of Gurobi with a solver that is substantially slower may mean that an approximation algorithm of the model is necessary in constrained environments.

8.2 Traffic Playback

ElasticWISP is evaluated through the playback of captured network traffic, provided by Venture Networks. To use *ElasticWISP*, a network operator must first specify a link-saturation threshold for backhaul links. This can be per link, or on a network-wide basis. If the traffic passing over a link

exceeds a threshold for more than a given number of consecutive periods, the "safe capacity" of a given link will be used for solving the optimisation problem. Where faster links are available, this will result in them being turned on, as the sum of all measured flows across the link will exceed what the optimiser knows the active link can provide. Although this approach is naïve and cannot ascertain the actual demand, it is an effective operator-tunable means of powering up higher-capacity radios when they are most likely needed. Furthermore, due to unavoidable Ethernet overheads – i.e., preamble, inter-frame spacing, and Cyclic Redundancy Check (CRC) – specifying a link safety margin is necessary, as the recorded demand does not take these into account.

Reading from the packet capture, *ElasticWISP* "plays back" the recorded network traffic. The demand for network traffic across backhaul pairs is analysed, and Gurobi is invoked with the appropriate traffic demands as necessary[1]. *ElasticWISP* then calculates the sum of network-wide energy conservation. However, despite the *ElasticWISP* scheme having a complete overview of traffic demands between backhaul pairs, the responsiveness of *ElasticWISP* to network demand, and variation in traffic patterns, is dependent on the operator specified configuration. The ability of *ElasticWISP* to respond to change in traffic patterns is also inherently limited by the time it takes a pair of wireless radios to boot and reassociate. As expected, the time taken for radio pairs to boot and reassociate varies between make and model.

8.2.1 Traffic Capture

Traffic was captured from Venture Networks' core switch. Due to the high aggregate throughput in the network, there were concerns that traditional packet capture software such as tcpdump [77] might lack the performance to capture packets at 1 GbE line-rate or beyond. Consequently, a Data

[1]Formatted as a traffic matrix.

Plane Development Kit (DPDK)-based traffic dumping utility [78] – built using libmoon – was leveraged. The DPDK-based utility is capable of faster than 10 GbE line-rate traffic dumping, when connected to the appropriate hardware. In the case of Venture Networks' core infrastructure, 10 GbE connectivity is used between devices. On the 10 GbE core switch, a port mirror was established, and traffic was dumped to a server that subsequently recorded it to disk[2].

A diagram of the traffic capture setup is shown in Fig. 8.1.

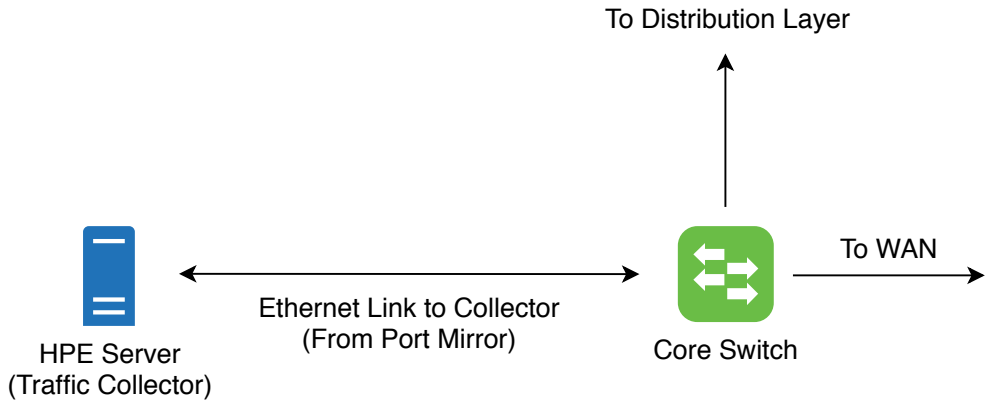

Figure 8.1: Traffic Capture Setup (Simplified).

Alternatively, ascertaining throughput demand using Netflow [79] or IP Flow Information Export (IPFIX) [80] flow-data would be possible, and require significantly less computational resources. Full packet capture was performed with aspirations that the dataset could be used for other purposes in the future. Of course, the complete dataset could also be used to generate flow-data.

Finally, determining when to invoke the optimiser means that knowing the traffic demand between active backhaul pairs is desirable. A consequence

[2] An HP Enterprise (HPE) ProLiant ML350 Gen9 Server with dual Intel Xeon E5-2609v4 Central Processing Units (CPUs) was used. The server is equipped with two 10 GbE Intel X540-AT2 NICs. The DPDK is covered in greater detail in Section 8.6.6.

of capturing data at a central location is that this may often be difficult. In the case of the Venture Networks topology, the process is simple, as we know what IP addresses are in use at each site, and that traffic must always pass between the same physical sites (irrespective of what backhaul radios are in use). However, in a more complicated network topology, this process might not be straightforward. Using Simple Network Management Protocol (SNMP) is a viable solution, as it is supported by default by most backhaul radios, and querying active throughput is generally simple.

8.2.2 Optimiser Invocation

A wide range of predictive models could be used with the *ElasticWISP* optimiser. For validation, a straightforward, Simple Moving Average (SMA) approach was used [81]. For each backhaul interface connected to a node – or *SR-MPLS* router – the network operator must:

- Specify an average interval for throughput sampling, e.g., 30 seconds.
- Specify the number of consecutive sampling periods (the time defined above) needed above a given (link throughput) safety margin in order to invoke the optimiser.

Next, the process of averaging the throughput using a SMA is straightforward. To determine when to invoke the optimiser, counters are applied to the consecutive periods of increase – or decrease – (at given thresholds) shown in the moving average:

1. Average the throughput over the given length of the moving average. If the rate is above the given invocation threshold, increment the counter. *Else, continue*.
2. If the rate is below the given threshold, *reset the counter*.
3. If the counter is equal to the number of consecutive periods required to invoke the optimiser, then *invoke the optimiser*. *Finally, reset the*

counter.

The same concept is applied for determining when to turn links off[3]. It can be observed from the traffic dataset that when the throughput over a link increases steadily, it will generally do so and remain high (or low if decreasing) for some time. Usually this period is long enough that we can justify powering up a faster radio, and hence the naïve approach above is effective. Other factors such as the average throughput over other link-pairs must also be considered[4]. For example, if the optimiser is invoked due to high-demand over a particular backhaul pair, and another backhaul pair is experiencing a transient spike in traffic, *ElasticWISP* may attempt to power up a faster radio pair to accommodate the surge. Having this situation occur is problematic, as even if the transient spike in demand does not meet the innovation criteria, the optimiser is unaware. One approach to mitigating this is using the average throughput over an extended time for links that did not invoke the optimiser, in order to avoid any unnecessary radio power-ups[5].

To leverage the SMA approach, link safety margins are set, meaning the optimiser is invoked with reduced link capacities. Effectively, this means that when the sum of all commodities going over any link i, j, exceeds the (safe) capacity of that link, the optimiser will be invoked, and try to find a alternative path. Of course, in some rare circumstances this can introduce model infeasibility. To deal with infeasibility, an overflow arc is set for any link i, j in the topology used by *ElasticWISP* for optimisation. The overflow arc has a very high-cost, and is only used when no alternative paths exist. Practically, the overflow arc results in the highest-throughput backhaul radio between two nodes being turned on. From practical experience working with Venture Networks, the self-reported capacities of backhaul

[3]The process is only necessary to avoid excessive runs of the *ElasticWISP* optimiser.

[4]"Smoothing" the throughput between backhaul pairs was performed using a SMA during evaluation.

[5]See Section 8.2.3

radio pairs are overly optimistic, which makes extracting their capacities in real time a challenge.

8.2.3 Traffic Matrix Generation

During the "playback" of network traffic, *ElasticWISP* must generate a traffic matrix that is used as an input for Gurobi. To solve the optimisation problem, we consider each WISP site to represent a single commodity[6]. Practically, this means the traffic matrix used for optimisation consists of a list of demands between network sites and the gateway. The average throughput from a network site to the network gateway over an arbitrary period of time could be used as an input. However, if the period of time is too small, the optimiser itself is unaware of any diurnal trends, and could turn links on or off unnecessarily. Of course, using a simple approach, such as a moving average, will reveal throughput trend more effectively than the instantaneous rate as reported using SNMP from routers[7].

8.3 QoS Preservation

One of the most important considerations for the design of *ElasticWISP* was ensuring QoS preservation. More than anything else, packet loss, latency spikes, and excessive jitter should always be avoided. From previous practical work, it was established that when OSPF link-costs are adjusted, Quagga will reroute traffic to the next shortest path without disrupting network performance. To determine the impact on QoS caused by the rerouting process, a 4x4 grid topology (see Fig. 8.2) was emulated using the CORE [82] network emulator.

[6]When using *SR-MPLS*, individual customer IP prefixes could be considered as individual commodities, which may result in improved distribution of flows under high network load.

[7]Manufacturer specific. Some SNMP-capable devices report filtered values.

A set of QoS samples was taken over consecutive 30-second periods. Round Trip Time (RTT) samples were taken at the same time that TCP traffic was being generated and used to saturate the links. Datagram loss and jitter was measured using the User Datagram Protocol (UDP) traffic generator and collector iperf [83]. Figs. 8.3–8.5 show the RTT, jitter, and datagram loss when (i) links are cost-adjusted, and (ii) when links are taken down without link cost-adjustment (reference). The results show that over all sampling intervals, the staggered link cost-adjustment approach does not result in any significant change in RTT, jitter, or datagram loss.

When compared to the reference approach, the results are evident. The simple approach of switching off the links for route changeover results in substantial datagram loss. The noticeable loss is due to the delay in the forwarding tables being updated to reflect the changeover, which does not happen until the IGP reconvergence is complete. In a congested network, IGP reconvergence can take a significant amount of time, and should be avoided.

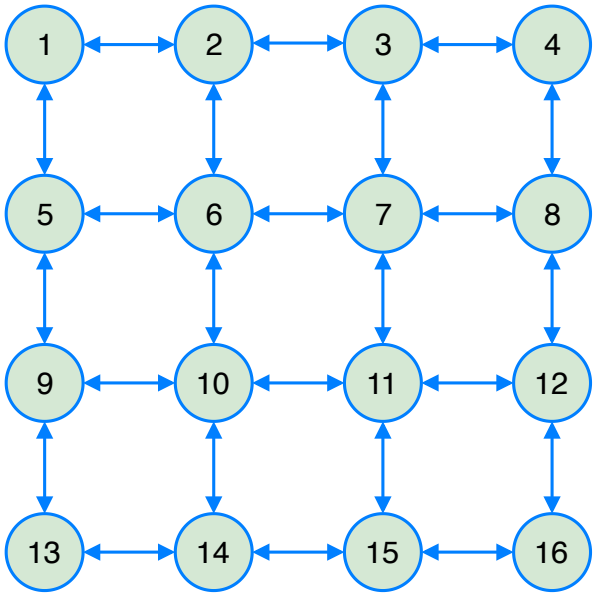

Figure 8.2: 4x4 Grid Topology.

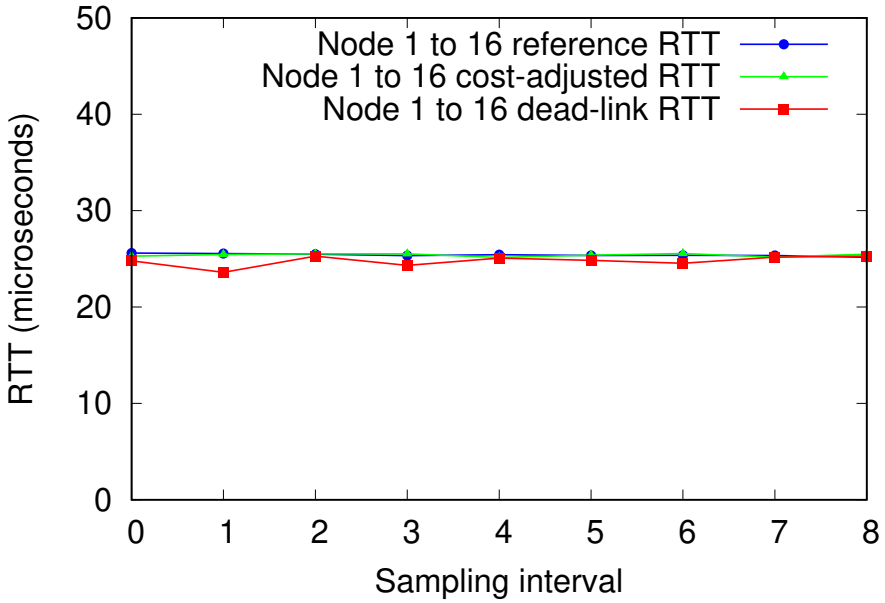

Figure 8.3: Node 1 to Node 16 RTT.

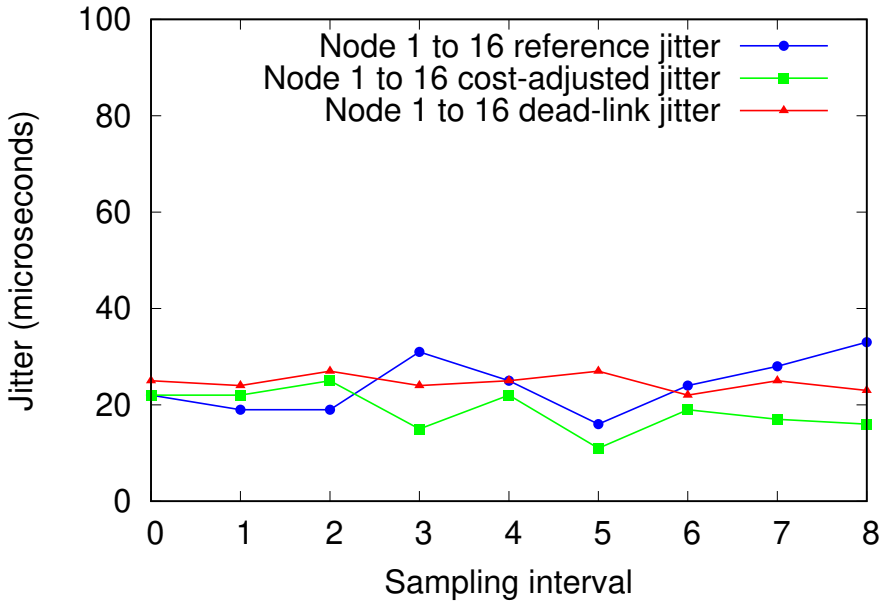

Figure 8.4: Node 1 to Node 16 Jitter.

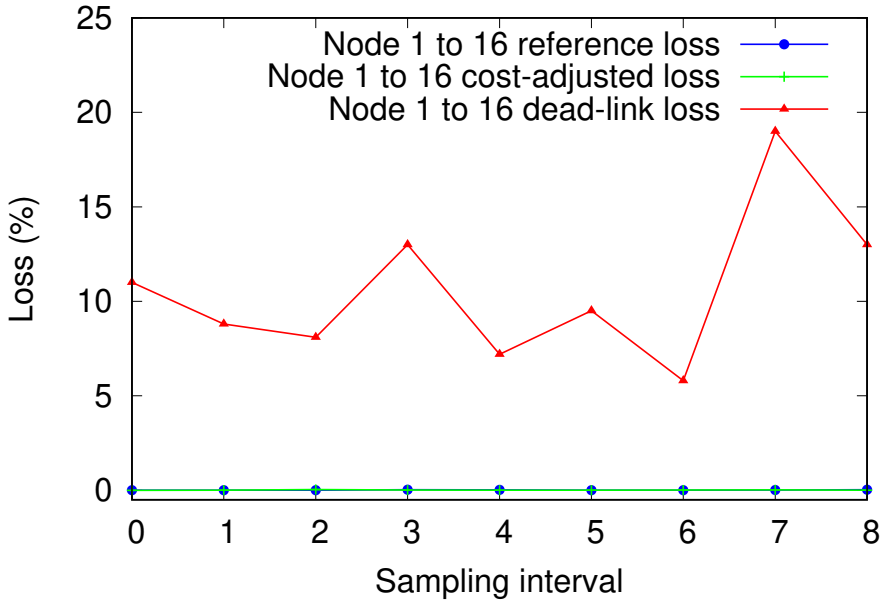

Figure 8.5: Node 1 to Node 16 Datagram Loss.

8.4 Energy Consumption

Next, the performance of *ElasticWISP* is evaluated in terms of energy consumption. Like many other optimisation schemes that aim to reduce energy consumption, the most important metric we are concerned with is the energy consumption of the reduced topology as a percentage of the original energy consumption:

$$\frac{\text{Energy consumed by } \textit{ElasticWISP}}{\text{Energy consumed by original WISP topology}} \times 100 \qquad (8.1)$$

8.4.1 Setup

The performance of *ElasticWISP* is evaluated using the network topology of Venture Networks' Horowhenua WISP network. In addition, the traffic cap-

ture used consists of ingress and egress traffic from Venture Networks' core switch. The traffic capture consists of two weeks of network traffic, split into numerous smaller captures for manageability. The Venture Networks' WISP topology, shown in Fig. 8.6, is used for evaluation. In the emulated evaluation, it is assumed that every node in the network is equipped with two different backhaul radio types: (i) a high-power, high-throughput radio (airFiber 5); and (ii) a low-power, low-throughput radio (PowerBeam 5AC G2).

It should be noted that the energy saving potential of *ElasticWISP* is much greater where a meshed or partially meshed topology is used. In Fig. 8.6, no redundant links between sites exist. If the topology had redundant links, e.g., between Ohau–Able, *ElasticWISP* – in certain circumstances – might reduce the topology further, forming a "minimum power tree" where constraints are satisfied.

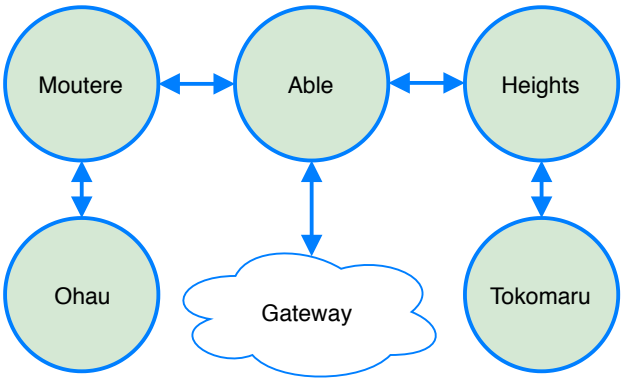

Figure 8.6: Evaluation Topology.

From the traffic capture, a one week period was evaluated using *ElasticWISP*. We evaluate how many times the optimiser produces a different subset of the original topology, and record the energy used during each 24-hour period. The network traffic from all nodes of the topology follows a predictable diurnal pattern, with little variation across days of the week. We establish that the power consumption of the network topology running entirely with high-power airFiber 5 radios (50 W) is a maximum of 9600 Watt hours (Wh) per day. The formula used for finding the daily consumption is simple:

$$\text{Consumption (Watts)} \times 8 \text{ radios} \times 24 \text{ hours} \tag{8.2}$$

In the case of Venture Networks, bidirectional backhaul links exist between:

- Ohau – Moutere
- Moutere – Able
- Able – Heights
- Heights – Tokomaru

The site "Able" is connected to an upstream gateway in Fig. 8.6 using fibre optics. airFiber radios interconnect the other sites.

If the network were to run entirely on low-power PowerBeam 5AC G2 radios (8.5 W), the daily maximum consumption would be 1632 Wh. For validation, we assume that each site will be interconnected to each neighbouring site using two radios: (i) an airFiber 5, and (ii) a PowerBeam 5AC G2. Other than the period during radio boot time, Service Set Identifier (SSID) reassociation, and the subsequent rerouting of traffic, only one of the two radios will be active at a time. Radios that are inactive will be powered off to conserve energy.

We conservatively estimate that the low-power radio can support a maxi-

mum of 100 Mbps of capacity when transmitting or receiving L2 frames. We assume the high-power radio can support up to 500 Mbps while transmitting or receiving L2 frames. We know in reality these values will be different. The performance of unlicensed radios is – at a minimum – challenging to accurately predict, and is subject to factors such as external interference, rain fade, and obstructions in the line of sight and Fresnel zone. Lastly, in this topology we expect that distant links, i.e., Ohau–Moutere and Heights–Tokomaru, will offer the best potential for energy conservation as they carry the least amount of transient traffic from other nodes.

8.4.2 Results

Upon running our evaluation of *ElasticWISP* we discovered promising results. We assume that using a straw man configuration with only high-power airFiber 5 radios, the evaluation topology would consume approximately:

$$50\text{ W} \times 8\text{ radios} \times 24\text{ hours per day} \times 7\text{ days of the week} \tag{8.3}$$

Overall, the weekly consumption should be approximately 67,200 Wh, or 67.2 Kilowatt hour (kWh). Utilising *ElasticWISP* and the provided data set with a combination of airFiber 5 and PowerBeam 5AC G2 radios, we were able to reduce the consumption for a given week to 23,765 Wh, or $\approx$23.8 kWh. Overall, this saves 65% of network-wide energy, a significant reduction. Of course, this figure uses the maximum power consumption of the radios. In practice, the energy reduction may in fact be greater, as we know that low-power backhaul radios – such as the PowerBeam 5AC G2 – available to WISPs typically use much less energy than their design maximum, whereas high-power, full-duplex radios – such as the airFiber 5 – use near their design maximum energy consumption continuously.

Fig. 8.7 shows that network-wide energy consumption throughout the week is substantially reduced. On most days, *ElasticWISP* will turn on high-power radios for less than four hours. The exception to this is the weekend, where traffic demand is far less predictable. We see that network-wide energy consumption is lower on Friday than any other day throughout this week, perhaps due to social factors.

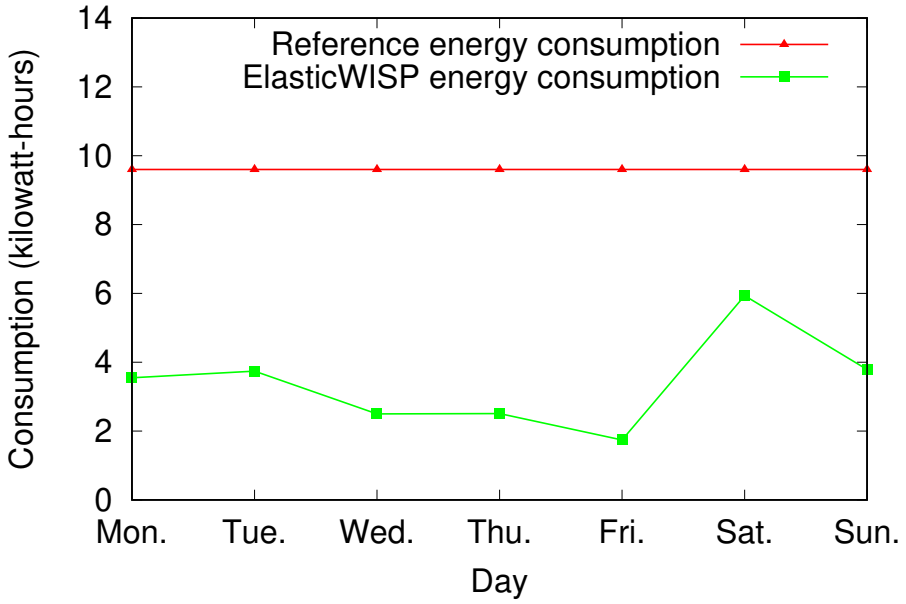

Figure 8.7: Energy Consumption Over a 7-Day Period.

Table 8.1 compares *ElasticWISP* to other state-of-the-art schemes: OSPF-based Routing on Demand [84], Segment Routing-based Energy Efficient Backbone Networks [85], and EAR [12][8]. Note that both "Routing on Demand" and "Energy Efficient Backbone Networks" are formulated as multi-commodity flow problems, and are focused toward wired service provider backbone networks.

[8]The cited implementation of EAR is designed for microwave backhaul networks, but was evaluated on a series of wired backhaul topologies, potentially resulting in lower-than-possible energy-savings.

Table 8.1: Validation: Competitors

Platform	***Flow path enforcement***	***Energy saving***	***Approach***
ElasticWISP	IGP best-path or *SR-MPLS*	$\approx 65\%$	Radio changeover, turning off links
Routing on Demand [84]	IGP best-path (with optimal link-weights)	$\approx 43\%$	Turning off links
Energy Efficient Backbone Networks [85]	Explicit (Segment Routing)	$\approx 44\%$	Turning off links
Energy-Aware Routing [12]	Numerical evaluation	$\approx 28\%$	Turning off links

A key advantage *ElasticWISP* has over competitor schemes, and specifically for WISP – and perhaps microwave – use, is the ability to perform radio changeover based on traffic demand. Practically, this is similar to approaches that turn off links in a bundle – e.g., of line-cards – to conserve energy in a wired service provider network. However, *ElasticWISP*'s energy saving potential is high due to the disparity of energy consumption between the backhaul radios that were studied. As a result, in comparison to other schemes, *ElasticWISP* can conserve more energy in WISP networks where both high and low power backhaul radios are used.

Formally, *ElasticWISP* is similar to ElasticTree[9], which itself is formally similar to earlier efforts to conserve energy in service provider networks [86]. As *ElasticWISP* uses a subset of the energy minimisation constraints proposed by ElasticTree and earlier efforts, the performance difference between the two platforms will be negligible for any given test topology. However,

[9]See Chapter 2.1.2.

unlike ElasticTree, we do not consider turning off switches, or in our case, nodes at each network site. As a result, ElasticTree can conserve additional energy where it is able to turn off redundant switches.

Functionally, *ElasticWISP* is very different to ElasticTree. The use of OSPF and later Segment Routing, as opposed to OpenFlow, introduces its own unique challenges, as does dealing with the unpredictable nature of unlicensed radio links. In addition, ElasticTree deals with links that can be brought up and down very quickly, whereas radio links will always have comparatively expensive boot and reassociation times. In addition to this, practical deployment of *ElasticWISP* itself required a new architecture of remotely managed PoE devices to be developed.

8.5 Implementation Considerations

While the OSPF-based link cost-adjustment design and implementation of *ElasticWISP* can help WISP operators realise substantial energy savings, it is not without caveats. The time taken for traffic to be rerouted is not instantaneous; the OSPF link-costs of a given link (radio pair) must be adjusted prior to powering them down. The time it takes for traffic to be routed over a lower-cost path is OSPF implementation-specific. While these changeover times were consistently low in this implementation, there is no guarantee that other OSPF daemons – and their given configurations – will perform in the same way.

If an appropriate traffic engineering mechanism was to be implemented alongside *ElasticWISP* instances – such as the Segment Routing approach described later in this chapter – this problem could be eliminated. If the path of a given flow was computed in advance with packets set to traverse a specific set of segments, then a WISP operator would not need to wait for an IGP-computed path reroute to happen prior to powering a link down.

Lastly, *ElasticWISP* is designed for the optimisation of backhaul links on

WISP networks. While it was shown that full-duplex (airFiber) backhaul radios used by WISPs account for a significant portion of energy consumption per site, we must also consider the point-to-multipoint radios that are used for customer access purposes. Much like their point-to-point counterparts, customer access radios typically used by WISPs have very little variation in energy consumption, irrespective of how much traffic is being passed over them. Typically these access radios are connected to sector antennas that distribute Internet access to customers. Sector antennas used will have a manufacturer specified azimuth and elevation radiation pattern. For example, in a high-capacity configuration, a WISP may have numerous sector antennas, each with a 30 degree azimuth radiation pattern serving a collection of customers.

During off-peak periods of network consumption, we propose that WISP operators could offload traffic from numerous high-capacity radios onto a lesser number of (low-power) radios that have antennas covering a wider azimuth. Essentially, customers would be transitioned to use an access point that can only handle low-demand for throughput. This approach has similarities to historical work proposed for energy conservation [87]. While existing equipment from the likes of Ubiquiti Networks enables WISPs to set a fallback SSID, there is currently no way of carrying out a graceful transition between access radios.

A recently developed handover scheme, BigAP [88], has shown that Dynamic Frequency Selection (DFS) Channel Switch Announcement Information Element (CSA-IE) can be used as a seamless infrastructure initiated handover mechanism between access points. Unfortunately for WISPs, this mechanism is likely not an option, as BigAP relies on radios being compliant with the IEEE 802.11n/ac standards, which both include the IEEE 802.11h amendment. Radios from the likes of Ubiquiti Networks et al. make no claim to be compliant with such IEEE standards, and now typically use custom radio chipsets for improved spectral efficiency [89].

8.6 *SR-MPLS* Benchmarks

To evaluate the performance of the *SR-MPLS* scheme, a simple linear topology[10] consisting of four hosts was created in the CORE emulator[11]. The *SR-MPLS* evaluation setup consisted of two standard Linux hosts. The first host was responsible for UDP packet generation, with the second host acting as a traffic collector. The two other hosts performed MPLS encapsulation and decapsulation duties. While simple, this topology enables us to evaluate how well the scheme can perform in a worst-case scenario, i.e., performing encapsulation or decapsulation. In contrast, MPLS to MPLS forwarding within an *SR-MPLS* domain is less taxing, as shown earlier in Fig. 6.4.

The *SR-MPLS* setup described was evaluated against an equivalent IP routing based implementation. The two *SR-MPLS* encapsulation and decapsulation hosts were replaced with standard Linux hosts running the Quagga routing suite, with OSPFv2 enabled for the creation of IPv4 routes. An overview of this reference setup is shown in Fig. 8.8.

Furthermore, the design of *SR-MPLS* as described in Chapter 6 is implemented using the systems programming language Rust [90], which has competitive performance with *C*. Rust was used due to familiarity with the low-level libpnet [91] networking library, which offers an excellent trade-off between ease-of-programming and performance. In merit of using Rust and the libpnet library for implementation, UDP packet generators written using libpnet have – marginally – outperformed their equivalent *C* counterparts[12]. Finally, ZeroMQ was used as the message queue of choice in this implementation.

[10] Using a simple linear topology for evaluation is consistent with previous Segment Routing research [43].

[11] Evaluation was performed on an Ubuntu Server 18.04.3 LTS host with an Intel i7 9700K CPU and 16GB of 2666 MHz DDR4 RAM.

[12] Rust and *C* benchmarks are available on the libpnet GitHub repository [92].

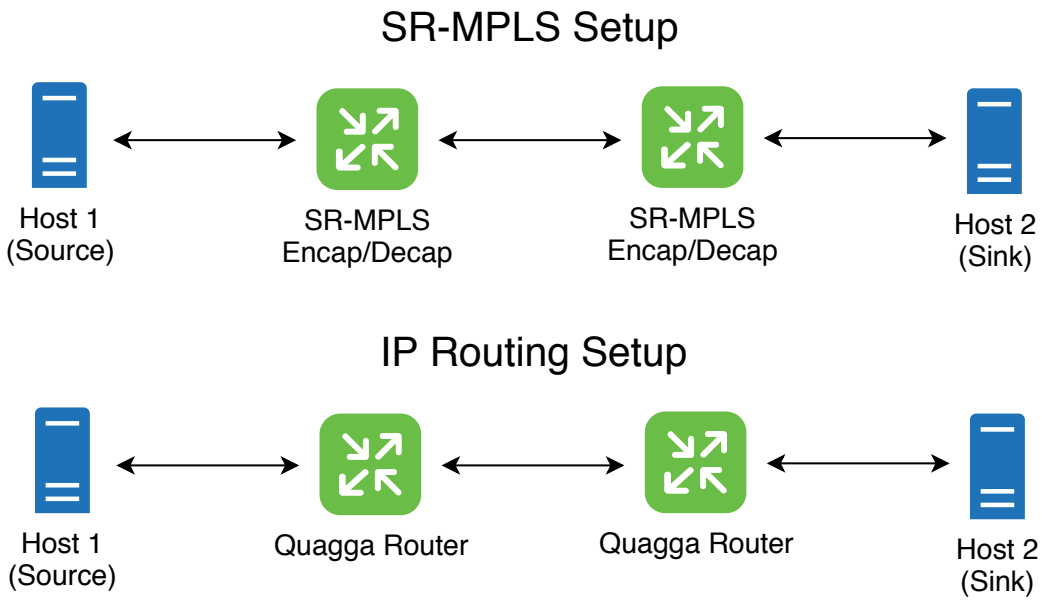

Figure 8.8: *SR-MPLS* Evaluation Setup.

8.6.1 Constraints

The testing of the *SR-MPLS* implementation in a virtual environment is a good initial indicator of performance, but should not be taken as a substitute for testing on real hardware. Unfortunately, acquiring an appropriate set of hardware for dedicated testing was out of budget. However, the CORE network emulator is a good trade-off, as it uses Linux network namespaces [93] for lightweight containerisation. As opposed to full virtual environments, such as Kernel Virtual Machine (KVM), network namespaces are lightweight, generally offering better performance.

8.6.2 RTT Benchmarks

One of the most important metrics of performance in data networks is latency, or RTT. The baseline Linux versus *SR-MPLS* RTT measurements are shown in Fig. 8.9. In a testament to the *SR-MPLS* implementation, the latency between *Host 1* and *Host 2* is consistently lower than what

is achieved using standard IP forwarding, with the use of Quagga for route setup. One of the reasons *SR-MPLS* can achieve very low-latency forwarding is the minimal processing overhead required in the evaluation setup. In the linear topology, the MPLS FIB of the *SR-MPLS* routers consists of one MPLS-capable *SR-MPLS* neighbour. When IP packets come into either *SR-MPLS* router, only the single FIB entry is found, and the IP packets subsequently have a single adjacency SID pushed onto them.

The process is repeated when the MPLS packets are decapsulated, or "popped". The IP packet must be forwarded out an interface, but there will only be one customer attached subnetwork to either *SR-MPLS* router in the topology – either *Host 1* or *Host 2*. In short, the *SR-MPLS* routers behave in a very simple, predictable way. Another consideration, as the *SR-MPLS* routers run in user space, is process priority. The priority of the *SR-MPLS* routers in this example – and all other figures in this series of subsections – is normal, and their *niceness* values have not been altered.

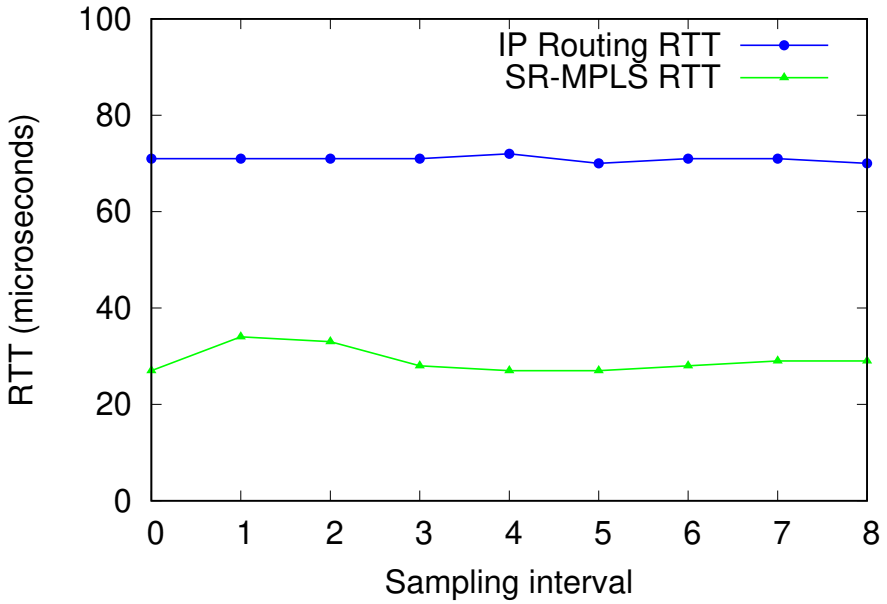

Figure 8.9: Host 1 to Host 2 RTT.

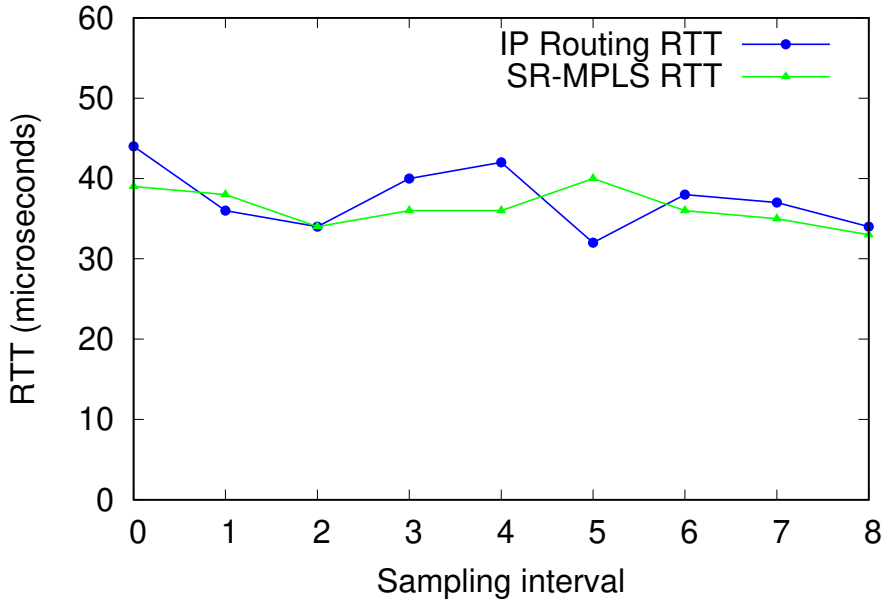

Figure 8.10: Host 1 to Host 2 RTT (1 Gbps UDP Load).

Interestingly, Fig. 8.10 shows a curious phenomenon. As expected, when using *SR-MPLS* routers, the latency between *Host 1* to *Host 2* increases with throughput. Unexpectedly, the latency *decreases* when using standard IP forwarding. When running the tests again with an additional 1 Gbps of UDP load, as shown in Fig. 8.11, the results are reinforced. The *SR-MPLS* latency further increases – albeit marginally – while the Linux IP forwarding latency is consistently lower. This phenomenon can not be observed when using real hardware. In a test between two physical Linux hosts, the latency between them increased – as expected – with throughput.

Upon further investigation, it was determined that the decrease in RTT with load, when using CORE, is the result of Intel Dynamic Voltage and Frequency Scaling (DVFS) [94]. When the CPU is under very-low load, such as sending RTT packets, it was recorded running at a meagre 798.09 MHz (multiplier of 7.98). When iPerf is started, the CPU frequency increases massively, to 4687.46 MHz (multiplier of 46.87). When the *SR-MPLS* process

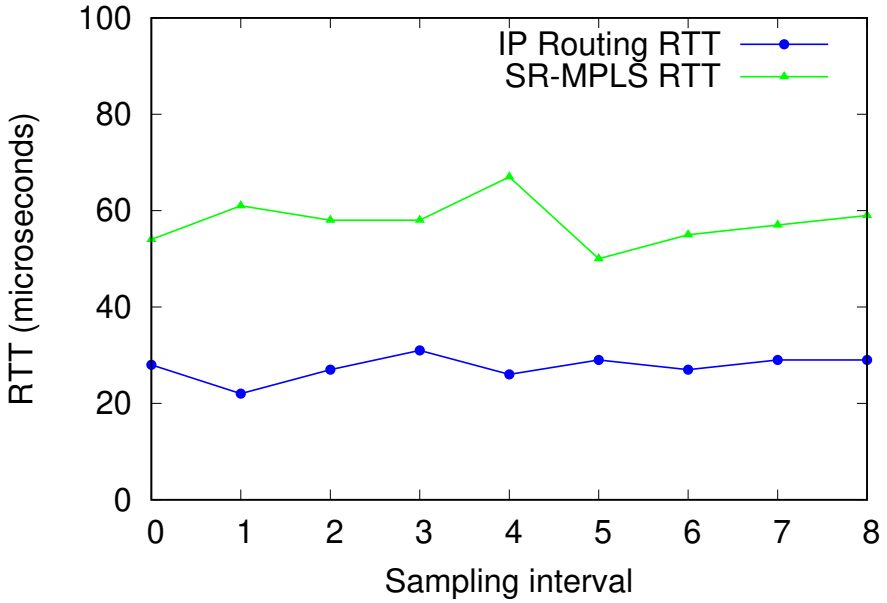

Figure 8.11: Host 1 to Host 2 RTT (2 Gbps UDP Load).

is run, DVFS scales the CPU frequency up, meaning the two *SR-MPLS* routers in the evaluation topology can forward RTT packets marginally faster than their standard IP counterpart.

Consequently, any process (running on any node within the evaluation topology) that causes DVFS to increase CPU frequency results in marginally improved latency between traffic source and sink. As shown in Fig. 8.9–8.11, the decrease in RTT with load is visible, but in respect to the RTT between physical devices, insignificant. Furthermore, the transmission and propagation delays experienced when using real hardware likely render the observed gains of *SR-MPLS* at low-load to be unsubstantial.

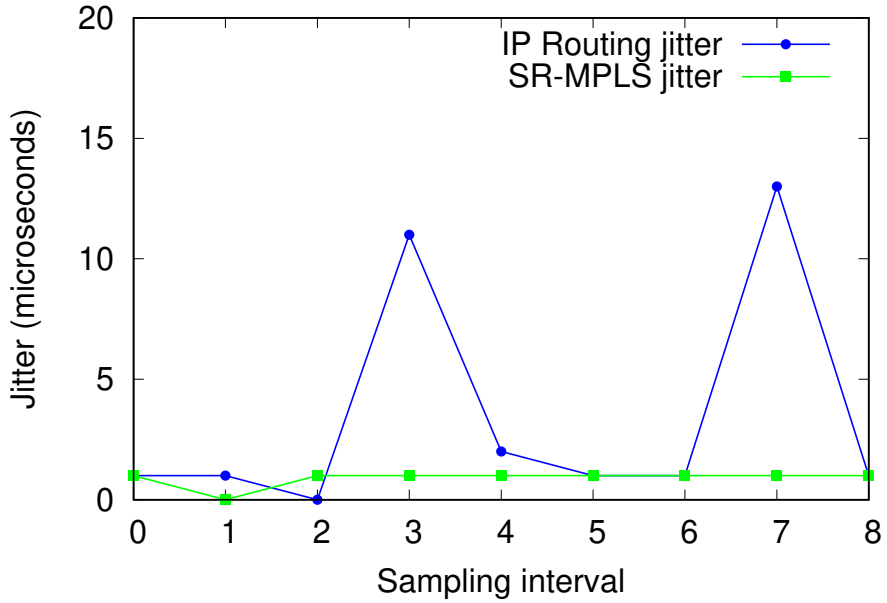

Figure 8.12: Host 1 to Host 2 Jitter (1 Gbps UDP Load).

8.6.3 Jitter Benchmarks

The jitter between *Host 1* and *Host 2* when using *SR-MPLS* routers or standard IP forwarding was consistently low, as expected. Fig. 8.12 shows two spikes in the recorded jitter when using the IP forwarding approach. While the large increases look bad, the jitter – which is consistently under 15 microseconds – is still very low. In comparison, a jitter test performed between a workstation and a server both situated in Wellington – in different physical locations, but connected to the same Metropolitan Area Network (MAN) – yielded an average of 3 milliseconds of jitter. In a physical testbed it is expected that the jitter would be much higher due to the additional processing delays of packets being sent and received by real hardware. Of course, higher latency is also expected for the same reasons.

8.6.4 Throughput Benchmarks

Fig. 8.13 shows the throughput rate of *SR-MPLS* routers versus packet loss, within the CORE environment. Despite running in lightweight network namespace containers, the throughput of *SR-MPLS* routers is competitive with what is offered by normal Linux hosts, easily surpassing 3 Gbps of throughput with 1492-Byte frames. Fig. 8.14 graphs the same dataset, but only up to 3300 Mbps. The linear increase in packet loss as throughput increases is seen in this figure. For future research, note that CORE, at the time of writing, cannot handle Ethernet frames larger than 1512 Bytes. Jumbo frames do not appear to be supported and are not functional. While CORE is still useful for validation of *SR-MPLS* routers, it is desirable to support jumbo frames for future, unexplored experiments with Segment Routing and *SR-MPLS* routers.

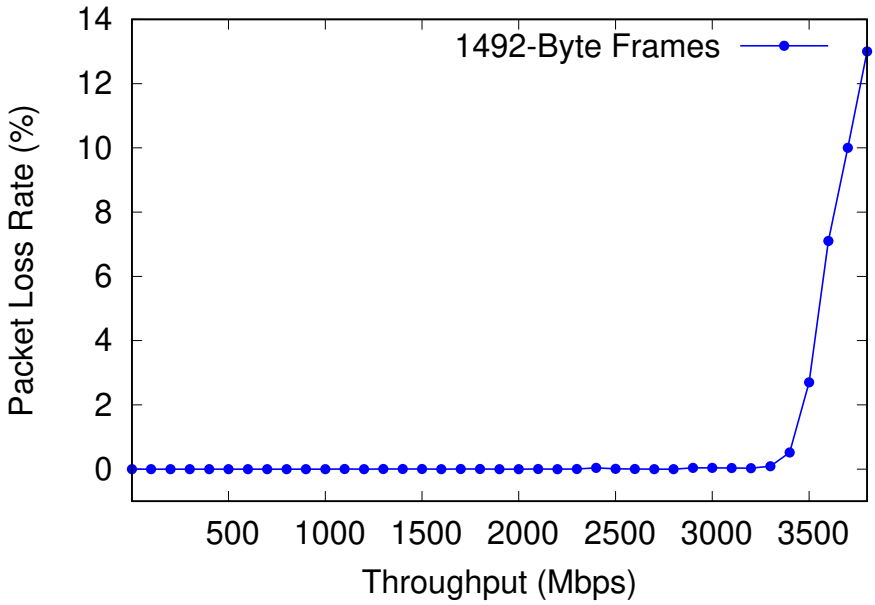

Figure 8.13: UDP Throughput vs. Packet Loss (0–3800 Mbps).

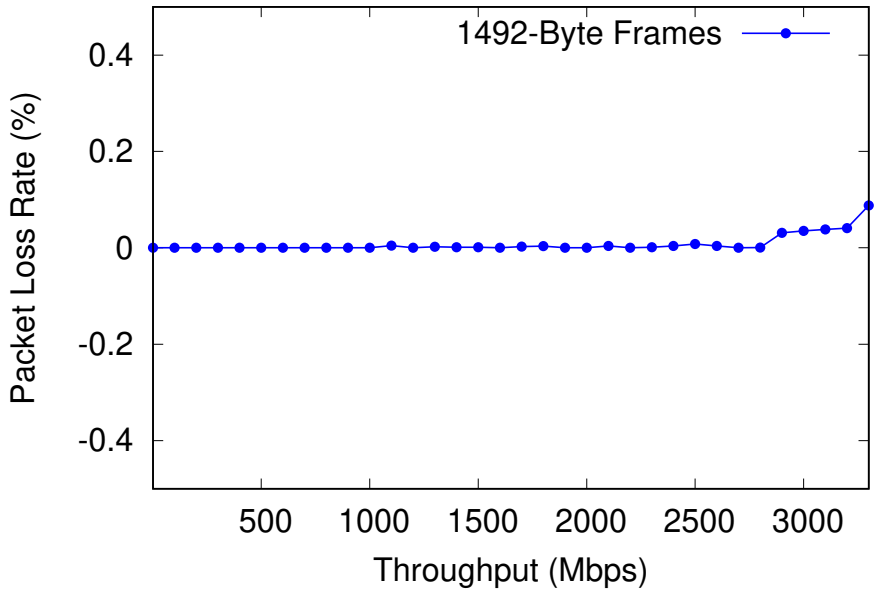

Figure 8.14: UDP Throughput vs. Packet Loss (0–3300 Mbps).

8.6.5 Encapsulation and Decapsulation Benchmarks

Fig. 8.15 shows the results from a set of tests to determine the Packets Per Second (PPS) – of encapsulation and decapsulation rates – of *SR-MPLS* routers running in CORE, and on real hardware. The encapsulation rate of the Linux kernel implementation of SRv6 is also shown [43, 95]. Results using CORE are acceptable, but measurably worse than results recorded on real hardware. The Rust implementation of *SR-MPLS* performs marginally better than the Linux kernel implementation of SRv6. However, in fairness to SRv6, an IPv6 SID is 128 bits [96]. In comparison, an MPLS SID is only 32 bits. The difference in processing overhead is significant, especially if you consider stacking multiple SIDs. Note that the SRv6 evaluation [43] that *SR-MPLS* is compared against in Fig. 8.15 does not include decapsulation performance. Hence, SRv6 and *SR-MPLS* decapsulation performance when using physical hosts is excluded from the graphic. Additionally, note that Figs. 8.13–8.15 and 8.17–8.18 show single-flow, single CPU performance.

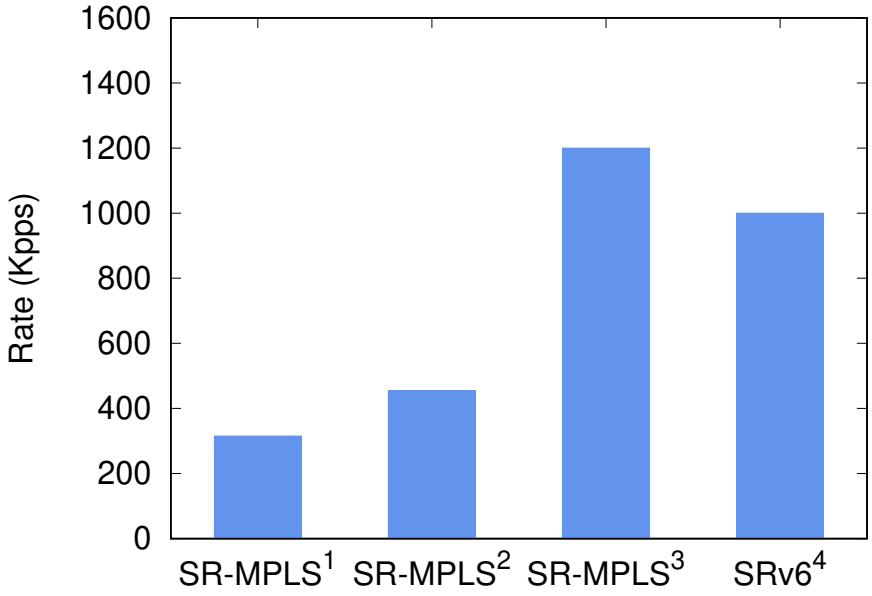

Figure 8.15: Packets Per Second, 84-Byte Frames on Wire.

Table 8.2: Segment Routing Performance

Platform	***Description***
SR-MPLS[1]	Worst case scenario using CORE. Decapsulation rate.
SR-MPLS[2]	Best case scenario using CORE. Encapsulation rate.
SR-MPLS[3]	Best case scenario using real hardware. Encapsulation rate.
SRv6[4]	Best case scenario using real hardware. Encapsulation rate.

8.6.6 Improving Performance

The *SR-MPLS* architecture was adapted into a minimal proof of concept using libmoon [97], a wrapper library for the Intel DPDK [98]. The Intel DPDK is a kernel bypass scheme used to achieve very high performance packet processing. As the name implies, performance gains are achieved

through bypassing the packet processing inefficiencies of the Linux kernel. Intel has shown DPDK to be capable of L3 forwarding – with 64-Byte packet sizes – at rates up to 233 Gbps, or 347 Mpps [99]. Even on commodity x86 hardware – using 10 GbE Intel NICs – the DPDK can be used to forward 64-Byte packets at line-rate using a single CPU core.

The evaluation setup for the libmoon/DPDK implementation is shown in Fig. 8.16. The setup was limited in comparison to the emulated (CORE) environment due to the prohibitive cost of high-end workstation hardware, and 10 GbE Intel NICs. The first workstation, equipped with an Intel i7 5820k, was used for packet generation. The workstation used for evaluating encapsulation performance is equipped with an Intel i7 9700k, which has substantially faster per-core CPU performance than the i7 5820k. To compensate for the slower CPU, libmoon was used for UDP packet generation. Finally, the workstation responsible for encapsulation was also connected to a 10 GbE switch. The switch was monitored to ensure MPLS packets were being received.

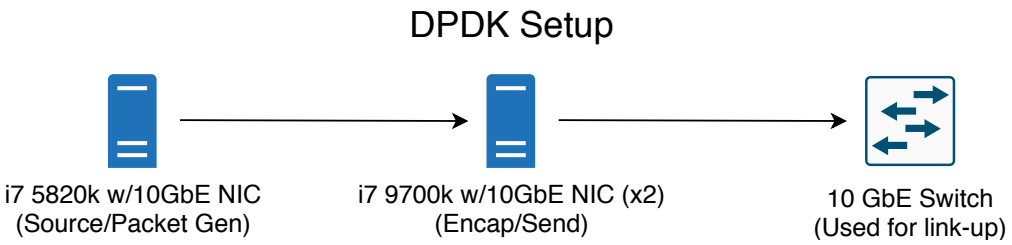

Figure 8.16: SR-DPDK Evaluation Setup.

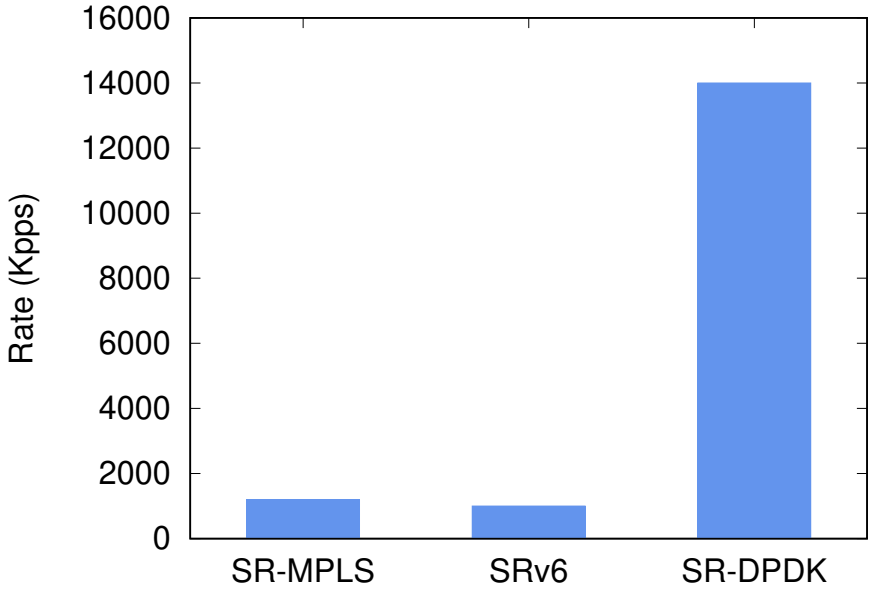

Figure 8.17: Packets Per Second, 84-Byte Frames on Wire (DPDK).

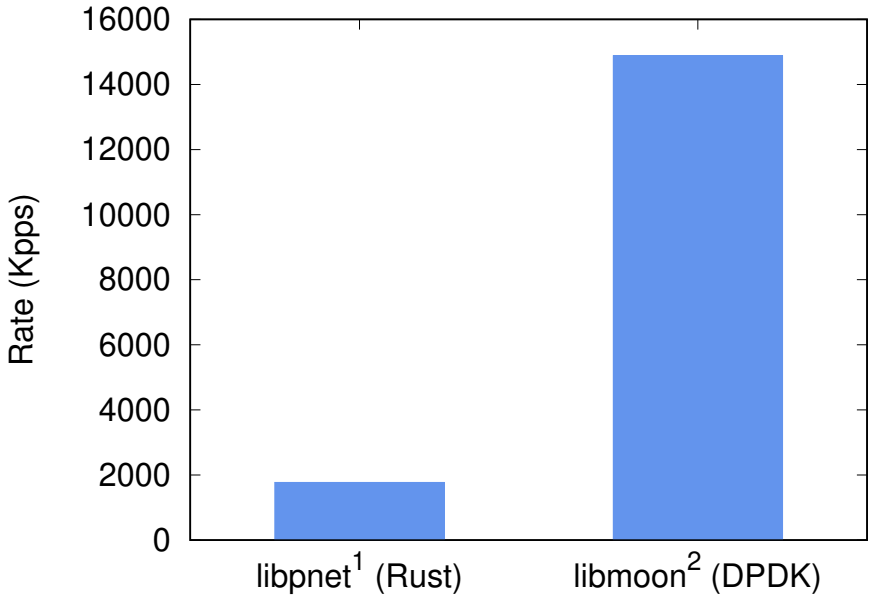

Figure 8.18: Packets Per Second, 84-Byte Frames on Wire (PacketGen).

Table 8.3: Segment Routing (PacketGen) Performance

Platform	*Description*
libpnet[1] libmoon[2]	Ubuntu Server 18.04.3 LTS host with a 10GbE Intel NIC, an Intel i7 9700K CPU, and 16GB of 2666 MHz DDR4 RAM.

Fig. 8.17 shows the performance gain achieved when using the DPDK for an *SR-MPLS* proof of concept. The DPDK can forward MPLS packets over 10 Mpps faster than the *SR-MPLS* implementation written in Rust using libpnet. These results are parallel with very recent Segment Routing research [100]. The performance of the DPDK is further shown in Fig. 8.18, where both libpnet and libmoon based MPLS packet generators were evaluated. The libpnet implementation can generate minimum size packets faster than line-rate Gigabit Ethernet, however it is still substantially slower than the libmoon implementation. In further support of the DPDK, the authors of libmoon also developed a proof of concept software router, known as MoonRoute [101]. In an impressive achievement, MoonRoute can route 64-Byte L3 IP packets – i.e., 84-Byte Frames – at 14.88 Mpps, which is 10 GbE line-rate.

8.6.7 Scalability

The Rust and libpnet implementation of *SR-MPLS* is fast; however, as stated earlier, the emulated environment should not be taken as a substitute for testing on real hardware. Of course, the best performance was observed when *SR-MPLS* was tested on real hardware, as expected. However, if *SR-MPLS* was tested on low-power hardware, e.g., that you may expect to see at a remote WISP site, performance could vary significantly. For example, a realistic testbed could consist of the same linear topology as described in Fig. 8.8, but instead of emulated hosts, could use low-power single board computers, such as the PC Engines APU4 series [102]. Note

that the APU4 series also appears to be capable of running the DPDK.

While the choice of hardware is important in determining the scalability of an *SR-MPLS* domain, it is also important to consider the maximum segment depth, and the the MTU being used across the network. A key advantage *SR-MPLS* has over its IPv6 counterpart, SRv6, is SID size. The 32-bit SID is substantially smaller than the 128-bit IPv6 SID, meaning even if a network is constrained to using an MTU of 1500 Bytes, the performance – as shown in Fig. 8.13–8.14 – is still acceptable. If the maximum segment depth is large, e.g., 32 SIDs, the overhead is still only 128 Bytes. If using SRv6, the equivalent overhead would be 512 Bytes.

The overhead of both *SR-MPLS* and SRv6 should also be taken into consideration when determining the scalability of a domain running either architecture. For most WISPs, dealing with larger-than-normal Ethernet frames should not be problematic, as common backhaul radios such as the airFiber series support jumbo frames of up to 9600 Bytes [103].

8.6.8 Why Kernel Bypass

Despite the current implementation of *SR-MPLS* having acceptable performance for use in real WISP networks, the ever-growing demand for throughput capacity must be considered. While the Linux kernel now supports basic MPLS functionality, as of version 4.3 [104, 105], the performance limitations of the kernel network stack mean that other options should also be considered. While familiarity with the libpnet library was useful in completing this evaluation, the library can also leverage netmap [106] as a backend for sending and receiving packets. Like the DPDK, netmap is a kernel bypass framework that focuses on implementing a faster-than-kernel network stack.

The benefit of using a library, such as libpnet, over utilising in-kernel MPLS handling functionality is simplicity. It also means that straightforward

integration with kernel bypass frameworks such as netmap and the DPDK can be leveraged at a later date[13]. Looking to the future, and at at potential throughput capacity requirements, leveraging kernel bypass approaches could be mandatory to meet demand. Furthermore, implementing *SR-MPLS* from the ground up using Rust and libpnet meant that complete control was maintained over the development of the scheme; eliminating kernel dependence and any associated development complexities[14].

8.7 Summary

The results presented throughout this validation chapter have shown that *ElasticWISP* and *SR-MPLS* are all but ready for deployment in real WISP networks. Following further validation on a greater range of hardware, it is expected that both schemes will be trialled at Venture Networks. The complete set of hardware developed for power control will enable straightforward deployment, without having to further modify devices such as PoE switches to integrate with the *ElasticWISP* optimiser. Finally, the results of the DPDK evaluation have shown that Linux kernel bypass approaches to networking have merit, and are an exciting avenue for further research. Concluding this thesis, Chapter 9 will next present a summary of findings and planned future work.

[13]DPDK support for libpnet is a work in progress.

[14]Normal operation of *SR-MPLS* using libpnet utilises the *AF_PACKET* socket for sending and receiving packets.

Chapter 9

Conclusion

This thesis has presented *ElasticWISP*, a practical architecture to enable the design of high-throughput, energy-proportional WISP backhaul networks. Through "playing back" daily network traffic, it has been shown that it is possible to employ high-throughput radios on demand while maintaining energy-proportionality during off-peak periods, reducing overall network energy consumption by around 65%. For a WISP constrained by their local environment, this may mean they can build a higher-performance network than they otherwise would be able to, due to limited energy harvesting capability.

Throughout this thesis, it has been shown that the *ElasticWISP* architecture has been divided into three key categories:

- The *ElasticWISP* optimiser, for computing optimal flow-paths.
- The *SR-MPLS* Segment Routing approach, for explicit flow-path enforcement.
- The backhaul power control hardware, for switching radios on and off based on the output of the *ElasticWISP* optimiser.

Together, these categories form a cohesive platform that has application

not only for WISPs, but for microwave operators, cellular providers, and even carriers with wired topologies. It is hoped that the concepts used for *ElasticWISP* can be applied to a variety of different physical hardware.

The power control hardware developed for integration with *ElasticWISP* and *SR-MPLS* further asserts the practicality of the architecture. While PoE switches specifically targeted at WISPs are readily available for purchase, they lack features that would enable them to integrate with the *ElasticWISP* architecture. In addition, such switches typically lack sophisticated energy monitoring capability, often lacking even minimal per-port power consumption metrics. Using the *ElasticWISP* power control hardware will enable future researchers to not only to control the power to radios, but to have a fine-grained view of their energy consumption.

Finally, the design and validation of the Segment Routing architecture, *SR-MPLS*, shows that it is capable of being used as-is on real WISP networks. When deployed, the *SR-MPLS* implementation enables network operators to steer packets across an ordered list of segments with explicit or arbitrary precision. Validation also showed that the minimal *SR-MPLS* implementation using the DPDK is competitive with even the most recent state-of-the-art Segment Routing research.

9.1 Future Work

The performance of the minimal *SR-MPLS* implementation using the DPDK makes it an appealing topic for future work. Should a full Segment Routing implementation be completed utilising the DPDK, it would have competitive packet processing rates to commercial Segment Routing solutions from vendors such as Cisco and Juniper. Releasing such an implementation as open-source could also have great benefit to the wider Internet community. Additionally, extending the open-source Linux kernel implementation of SRv6 to support IPv4 encapsulation would also mean that IPv6 only

networks could easily be used for IPv4 over IPv6 transportation.

The dataset used to evaluate the performance of the *ElasticWISP* optimiser was not straightforward to collect. However, it is still desirable to evaluate the performance of *ElasticWISP* on other WISP topologies, ideally also using data collected from each respective network. Working with larger WISPs would generally require using flow-data rather than full traffic dumps to run the *ElasticWISP* optimiser, due to the computational resources needed, even when using platforms such as the DPDK. Consequently, data collection should be easier, as most commercial routers are able to export flow-data, often in a variety of formats, e.g., Netflow and IPFIX.

Furthermore, deploying *ElasticWISP* in a real WISP network introduces new and exciting challenges. The predictive – or in the *ElasticWISP* case, reactive – model responsible for invoking the optimiser must use link safety margins. In a scenario where a link becomes congested before a faster radio can be turned on, it is important a WISP can still maintain QoS and QoE for important latency-sensitive traffic such as Voice over IP (VoIP).

In a situation where links become saturated before faster links can be turned on, standard First in, First out (FIFO) queues may result in network users – who are likely paying customers – experiencing poor network QoE. One of the most problematic issues that can arise is bufferbloat [107]. Essentially, bufferbloat is undesirable latency, and is caused by devices within the network buffering excessive data. To avoid service degradation for VoIP and other latency-sensitive real-time applications, bufferbloat must be carefully managed.

A common approach to managing bufferbloat is using Active Queue Management (AQM) [107]. One recent innovation, Flow Queue Controlled Delay (FQ-CoDel) [108, 109], is a hybrid packet scheduler and AQM algorithm, and has specifically been shown to reduce bufferbloat in capacity-constrained rural broadband access networks [110]. Future research will examine how AQM approaches, such as FQ-CoDel, can be used to im-

prove network QoE alongside the existing *ElasticWISP* energy-optimisation scheme.

Appendices

Appendix A

Evaluation Dataset

The dataset used for evaluation of *ElasticWISP* has not been anonymised, and could potentially be used to identify Venture Networks' customers and their browsing habits. Researchers wishing to access the dataset, or to run additional packet captures, are advised to contact Venture Networks directly.

Appendix B

Measurement and Control

In order to use a voltage divider to measure the voltage of the high-side switch output ports, we must first determine the correct resistors to use to build the voltage division circuit. Using the voltage division rule [111], we know that:

$$V_{OUT} = V_{IN}(\frac{R_2}{R_1 + R_2}) \tag{B.1}$$

Following the Infineon design guide for their PROFET+ series high-side switches [112], we consider using 1.5kΩ and 47kΩ resistors to form the voltage divider. Using the voltage division rule, we find that where $R_1 = 47k\Omega$ and $R_2 = 1.5k\Omega$, $V_{OUT} \approx 1.48V$ when V_{IN} is at a nominal 48V. This is within the 1.8V limit of the ADC. Of course, the resistors used in the voltage divider should be chosen so that their output does not exceed the maximum input voltage of the ADC used:

$$V_{IN} = \frac{V_{OUT} \times (R_1 + R_2)}{R_2} \tag{B.2}$$

Using the formula above, we find that $V_{IN} = 58.2V$ when $R_1 = 47k\Omega$,

$R_2 = 1.5k\Omega$ and $V_{OUT} = 1.8V$. Although we do not expect V_{IN} to exceed 48V, a Zener diode is added on each voltage sense circuit to protect the analogue input pins on the BeagleBone from overvoltage. Now suitable resistor values have been found and incorporated into the design, we must read the input from the ADC. This process is straightforward. Once we have read from the ADC we then need to convert the digital value back to the original analogue value, measured in volts:

$$V_{OUT} = \frac{digitalInput \times V_{REF}}{2^n - 1} \tag{B.3}$$

Where n is the number of ADC bits, V_{REF} is the reference voltage of the ADC being used, and $digitalInput$ is the ADC reading. Now V_{OUT} is known, we can calculate the voltage passing through the high-side switch. We use the following formula:

$$V_{IN} = \frac{V_{OUT}}{(\frac{R_2}{(R_1+R_2)})} \tag{B.4}$$

Likewise, if we know V_{IN} and need to determine what the ADC converted digital value will be, we can use:

$$digitalInput = \frac{V_{IN} \times (2^n - 1)}{V_{REF}} \tag{B.5}$$

Additional steps are necessary in order to determine the load current through the high-side switches. The process for attaining the analogue input voltage remains the same, however we must determine the current at the current sense pin. We place a resistor – R_{IS} – between the current sense pin on the high-side switch and the analogue input pin on the BeagleBone. Using Ohm's law, we can find the current using:

$$I_{IS} = \frac{V_{IS}}{R_{IS}} \tag{B.6}$$

Table B.1: Current Sense Ratio for I_L

Load	***Min***	***Typical***	***Max***
$I_{L0} = 50$mA	-50%	1600	+50%
$I_{L1} = 500$mA	-25%	1500	+25%
$I_{L2} = 1$A	-12%	1500	+12%
$I_{L3} = 2$A	-9%	1500	+9%
$I_{L4} = 4$A	-8%	1500	+8%

Where I_{IS} is the sense current proportional to the load current, V_{IS} is the voltage out of the current sense pin on the high-side switch, and R_{IS} is the value of the pull-down resistor, measured in ohms. The ratio between the sense current and the load current is given by:

$$k_{ILIS} = \frac{I_L}{I_{IS}} \tag{B.7}$$

Given this, we can show that:

$$I_{IS} = \frac{I_L}{k_{ILIS}} \tag{B.8}$$

Finally, we determine the current, in amperes, being used by backhaul radios connected to the high-side switch:

$$I_L = I_{IS} \times k_{ILIS} \tag{B.9}$$

Using the correct k_{ILIS} ratio is important to determine the load I_L with accuracy. Table B.1 shows the typical current sense ratio – k_{ILIS} – for loads ranging from 50mA to 4A [66]. While not mandatory, given the variation of the current sense ration when $I_L < 500$mA, a precise current sense amplifier can be used for more precise measurement at low load current. As an example, the Texas Instruments INA282 [113] is a precise current

sense amplifier. At 24V/500mA, the Root Sum Squared (RSS) error of the amplifier would be $< 2.5\%$. At 24V/50mA, the total RSS error of the amplifier would be $\approx 15\%$ [114]. Even when calibrated, the variation in the current sense ratio k_{ILIS} of the high-side switches makes the INA282 a desirable choice for precision load current measurement.

Appendix C

ElasticWISP Hardware

The hardware developed for enabling the use of *ElasticWISP* on real WISP networks is open source and freely available for download on GitHub [115].

Bibliography

[1] Y. Ben-David, "Connecting the Last Billion," Ph.D. dissertation, EECS Department, University of California, Berkeley, Dec. 2015. [Online]. Available: http://www2.eecs.berkeley.edu/Pubs/TechRpts/2015/EECS-2015-233.html

[2] S. Hasan, Y. Ben-David, M. Bittman, and B. Raghavan, "The Challenges of Scaling WISPs," in *Proceedings of the 2015 Annual Symposium on Computing for Development (DEV '15)*, 2015.

[3] T. Pötsch, S. Yousaf, B. Raghavan, and J. Chen, "Zyxt: A Network Planning Tool for Rural Wireless ISPs," in *Proceedings of the 1st ACM SIGCAS Conference on Computing and Sustainable Societies (COMPASS '18)*. New York, NY, USA: ACM, 2018, pp. 16:1–16:11. [Online]. Available: http://doi.acm.org/10.1145/3209811.3209874

[4] C. Rey-Moreno, Z. Roro, W. D. Tucker, M. J. Siya, N. J. Bidwell, and J. Simo-Reigadas, "Experiences, Challenges and Lessons from Rolling out a Rural WiFi Mesh Network," in *Proceedings of the 3rd ACM Symposium on Computing for Development (DEV '13)*. New York, NY, USA: ACM, 2013, pp. 11:1–11:10. [Online]. Available: http://doi.acm.org/10.1145/2442882.2442897

[5] S. M. Mishra, J. Hwang, D. Filippini, R. Moazzami, L. Subramanian, and T. Du, "Economic Analysis of Networking Technologies for Rural Developing Regions," in *Internet and Network Economics*, X. Deng and

Y. Ye, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2005, pp. 184–194.

[6] W. Waites, J. Sweet, R. Baig, P. Buneman, M. Fayed, G. Hughes, M. Fourman, and R. Simmons, "RemIX: A Distributed Internet Exchange for Remote and Rural Networks," in *Proceedings of the 2016 Workshop on Global Access to the Internet for All (GAIA '16)*. New York, NY, USA: Association for Computing Machinery, 2016, p. 25–30. [Online]. Available: https://doi.org/10.1145/2940157.2940162

[7] W. I. S. P. A. of New Zealand Inc. (2019) Don't Waste Our Spectrum: A message to rural New Zealand. Accessed: 8 Feb. 2020. [Online]. Available: http://www.wispa.nz/wp/wp-content/uploads/2019/06/dont_waste_our_spectrum.pdf

[8] D. Cameron, "Broadband for Remote and Rural Communities," Master's thesis, Victoria University of Wellington, 2020.

[9] S. Surana, R. Patra, S. Nedevschi, M. Ramos, L. Subramanian, Y. Ben-David, and E. Brewer, "Beyond Pilots: Keeping Rural Wireless Networks Alive," in *Proceedings of the 5th USENIX Symposium on Networked Systems Design & Implementation (NSDI '08)*, 2008.

[10] A. Nungu, R. Olsson, and B. Pehrson, "On Powering Communication Networks in Developing Regions," in *Proceedings of the 2011 IEEE Symposium on Computers and Communications (ISCC '11)*, Jun. 2011, pp. 383–390.

[11] Ubiquiti Networks. (no date) Ubiquiti Networks. Accessed: 23 Aug. 2019. [Online]. Available: https://www.ui.com/

[12] K. Alvinice, "Design and optimization of wireless backhaul networks," Ph.D. dissertation, Université Nice Sophia Antipolis, Dec. 2014. [Online]. Available: https://tel.archives-ouvertes.fr/tel-01165064/document

[13] C. Rey-Moreno. (2017, May) Supporting the Creation and Scalability of Affordable Access Solutions: Understanding Community Networks in Africa. Accessed: 21 Aug. 2019. [Online]. Available: https://www.internetsociety.org/wp-content/uploads/2017/08/CommunityNetworkingAfrica_report_May2017_1.pdf/

[14] E. Blantz and B. Baike, "Case Study: The Haiti Rural Broadband Initiative," in *Proceedings of the 6th USENIX/ACM Workshop on Networked Systems for Developing Regions (NSDR '12)*, 2012.

[15] A. Nungu, R. Olsson, and B. Pehrson, "On the Design of Inclusive Ubiquitous Access," in *Proceedings of the 3rd International Conference on Ubiquitous and Future Networks (ICUFN '11)*, 2011.

[16] S. Tombaz, P. Monti, F. Farias, M. Fiorani, L. Wosinska, and J. Zander, "Is Backhaul Becoming a Bottleneck for Green Wireless Access Networks?" in *Proceedings of the 2014 IEEE International Conference on Communications (ICC '14)*, Jun. 2014, pp. 4029–4035.

[17] M. Fiorani, S. Tombaz, P. Monti, M. Casoni, and L. Wosinska, "Green Backhauling for Rural Areas," in *Proceedings of the 2014 International Conference on Optical Network Design and Modelling (ONDM '14)*, May 2014, pp. 114–119.

[18] E. Zola, A. J. Kassler, and W. Kim, "Joint User Association and Energy Aware Routing for Green Small Cell mmWave Backhaul Networks," in *Proceedings of the 2017 IEEE Wireless Communications and Networking Conference (WCNC '17)*, Mar. 2017, pp. 1–6.

[19] W. K. Seah, B. Ng, and M. Libunao, "Autonomic Link Management in Wireless Backhaul Networks with OpenFlow and Traffic Prediction," in *Proceedings of the 2017 IEEE/CIC International Conference on Communications in China (ICCC '17)*, 2017.

[20] L. Dong. (2017) Application of SDN in Microwave Transmission. Accessed: 31 Jan. 2020. [Online]. Available: https://res-www.zte.com.cn/mediares/magazine/publication/tech_en/pdf/201706.pdf

[21] C. Dehos, J. L. González, A. D. Domenico, D. Kténas, and L. Dussopt, "Millimeter-wave access and backhauling: the solution to the exponential data traffic increase in 5G mobile communications systems?" *IEEE Communications Magazine*, vol. 52, no. 9, pp. 88–95, Sep. 2014.

[22] D. Yuan, H. Lin, J. Widmer, and M. Hollick, "Optimal Joint Routing and Scheduling in Millimeter-Wave Cellular Networks," in *Proceedings of the 2018 IEEE International Conference on Computer Communications (INFOCOM '18)*, Apr. 2018, pp. 1205–1213.

[23] M. Jaber, M. A. Imran, R. Tafazolli, and A. Tukmanov, "5G Backhaul Challenges and Emerging Research Directions: A Survey," *IEEE Access*, vol. 4, pp. 1743–1766, Apr. 2016.

[24] Facebook Connectivity. (2020) Terragraph. Accessed: 31 Jan. 2020. [Online]. Available: https://connectivity.fb.com/terragraph/

[25] Cisco. (2019) Cisco Visual Networking Index: Forecast and Trends, 2017–2022. Accessed: 31 Jan. 2020. [Online]. Available: https://www.cisco.com/c/en/us/solutions/collateral/service-provider/visual-networking-index-vni/white-paper-c11-741490.pdf

[26] M. Zhang, C. Yi, B. Liu, and B. Zhang, "GreenTE: Power-Aware Traffic Engineering," in *Proceedings of the 18th IEEE International Conference on Network Protocols (ICNP '10)*, Oct. 2010, pp. 21–30.

[27] B. Heller, S. Seetharaman, P. Mahadevan, Y. Yiakoumis, P. Sharma, S. Banerjee, and N. McKeown, "ElasticTree: Saving Energy in Data Center Networks," in *Proceedings of the 7th USENIX Conference on Networked Systems Design and Implementation (NSDI '10)*. USA: USENIX Association, 2010, p. 17.

[28] N. McKeown, T. Anderson, H. Balakrishnan, G. Parulkar, L. Peterson, J. Rexford, S. Shenker, and J. Turner, "OpenFlow: Enabling Innovation in Campus Networks," *SIGCOMM Comput. Commun. Rev.*, vol. 38, no. 2, pp. 69–74, Mar. 2008. [Online]. Available: http://doi.acm.org/10.1145/1355734.1355746

[29] K. M. S. Huq, S. Mumtaz, J. Rodriguez, and C. Verikoukis, "Investigation on Energy Efficiency in HetNet CoMP Architecture," in *Proceedings of the 2014 IEEE International Conference on Communications (ICC '14)*, Jun. 2014, pp. 1112–1117.

[30] K. M. S. Huq, S. Mumtaz, and J. Rodriguez, "QoS Aware Energy-Efficient Resource Scheduling for HetNet CoMP," in *Proceedings of the 2015 IEEE International Conference on Communications (ICC '15)*, Jun. 2015, pp. 5954–5960.

[31] K. M. S. Huq, S. Mumtaz, J. Bachmatiuk, J. Rodriguez, X. Wang, and R. L. Aguiar, "Green HetNet CoMP: Energy Efficiency Analysis and Optimization," *IEEE Transactions on Vehicular Technology*, vol. 64, no. 10, pp. 4670–4683, Oct. 2015.

[32] Faucet. (no date) Faucet Conference. Accessed: 26 Jan. 2020. [Online]. Available: https://conference.faucet.nz/

[33] ——. (no date) Faucet: Open source SDN Controller for production networks. Accessed: 26 Jan. 2020. [Online]. Available: https://faucet.nz/

[34] K. Myers and S. Parikh. (2019) SDN Traffic Engineering for Wireless ISPs. Accessed: 26 Jan. 2020. [Online]. Available: https://youtu.be/HyFXpEJdkNI

[35] K. Myers. (2018) WISP Design: OSPF "Leapfrog" path for traffic engineering. Accessed: 26 Jan. 2020.

[Online]. Available: https://stubarea51.net/2018/06/01/wisp-design-ospf-leapfrog-path-for-traffic-engineering/

[36] ——. (2017) WISP Design: Using eBGP and OSPF transit fabric for traffic engineering. Accessed: 26 Jan. 2020. [Online]. Available: https://mum.mikrotik.com/presentations/US17/presentation_4519_1496062656.pdf

[37] ——. (2016) WISP Design: Using OSPF to build a transit fabric over unequal links. Accessed: 26 Jan. 2020. [Online]. Available: https://stubarea51.net/2016/10/27/wisp-design-using-ospf-to-build-a-transit-fabric-over-unequal-links/#comment-2694

[38] Noction. (2018) BGP and Traffic Engineering Mechanisms for WISPs. Accessed: 26 Jan. 2020. [Online]. Available: https://www.noction.com/blog/bgp-and-traffic-engineering-mechanisms-for-wisps

[39] Xipeng Xiao, A. Hannan, B. Bailey, and L. M. Ni, "Traffic Engineering with MPLS in the Internet," *IEEE Network*, vol. 14, no. 2, pp. 28–33, Mar. 2000.

[40] G. Swallow and L. Andersson, "The Multiprotocol Label Switching (MPLS) Working Group decision on MPLS signaling protocols," RFC 3468, Feb. 2003. [Online]. Available: https://rfc-editor.org/rfc/rfc3468.txt

[41] C. Filsfils, S. Previdi, L. Ginsberg, B. Decraene, S. Litkowski, and R. Shakir, "Segment Routing Architecture," RFC 8402, Jul. 2018. [Online]. Available: https://rfc-editor.org/rfc/rfc8402.txt

[42] P. L. Ventre, S. Salsano, M. Polverini, A. Cianfrani, A. Abdelsalam, C. Filsfils, P. Camarillo, and F. Clad, "Segment Routing: a Comprehensive Survey of Research Activities, Standardization

Efforts and Implementation Results," *CoRR*, vol. abs/1904.03471, 2019. [Online]. Available: http://arxiv.org/abs/1904.03471

[43] D. Lebrun and O. Bonaventure, "Implementing IPv6 Segment Routing in the Linux Kernel," in *Proceedings of the Applied Networking Research Workshop (ANRW '17)*. New York, NY, USA: Association for Computing Machinery, 2017, p. 35–41. [Online]. Available: https://doi.org/10.1145/3106328.3106329

[44] S. Salsano, L. Veltri, L. Davoli, P. L. Ventre, and G. Siracusano, "PMSR — Poor Man's Segment Routing, a minimalistic approach to Segment Routing and a Traffic Engineering use case," in *Proceedings of the 2016 IEEE/IFIP Network Operations and Management Symposium (NOMS '16)*, Apr. 2016, pp. 598–604.

[45] Venture Networks. (no date) Super-Fast Rural Broadband. Accessed: 21 Aug. 2019. [Online]. Available: https://www.venturenetworks.co.nz/

[46] Texas Instruments. (2015) INA219: Zerø-Drift, Bidirectional Current/Power Monitor With I2C Interface. Accessed: 31 Jan. 2020. [Online]. Available: http://www.ti.com/lit/ds/symlink/ina219.pdf

[47] Netonix. (2014) Our Story. Accessed: 26 Aug. 2019. [Online]. Available: https://www.netonix.com/about-netonix/

[48] S. Even, A. Itai, and A. Shamir, "On the Complexity of Time Table and Multi-Commodity Flow Problems," in *Proceedings of the 16th Annual Symposium on Foundations of Computer Science (SFCS '75)*, Oct. 1975, pp. 184–193.

[49] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein, *Introduction to Algorithms, Third Edition*, 3rd ed. The MIT Press, 2009.

[50] S. Kandula, D. Katabi, S. Sinha, and A. Berger, "Dynamic Load Balancing Without Packet Reordering," *SIGCOMM Comput. Commun.*

Rev., vol. 37, no. 2, pp. 51–62, Mar. 2007. [Online]. Available: http://doi.acm.org/10.1145/1232919.1232925

[51] The ZeroMQ Community. (2019) ZeroMQ - An open-source universal messaging library. Accessed: 21 May 2019. [Online]. Available: https://zeromq.org/

[52] S. Furuhashi. (2019) MessagePack. Accessed: 21 May 2019. [Online]. Available: https://msgpack.org/index.html

[53] F.-X. Bourlet. (2015, May) zerorpc Protocol. Accessed: 14 Apr. 2019. [Online]. Available: https://github.com/0rpc/zerorpc-python/blob/master/doc/protocol.md

[54] Gurobi Optimization, LLC. (no date) About Gurobi. Accessed: 21 Aug. 2019. [Online]. Available: https://www.gurobi.com/company/about-gurobi/

[55] P. Psenak, S. Previdi, C. Filsfils, H. Gredler, R. Shakir, W. Henderickx, and J. Tantsura, "OSPF Extensions for Segment Routing," RFC 8665, Dec. 2019. [Online]. Available: https://rfc-editor.org/rfc/rfc8665.txt

[56] J. W. Evans and C. Filsfils, *Deploying IP and MPLS QoS for Multiservice Networks: Theory & Practice*. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 2007.

[57] H. Song, Z. Li, T. Zhou, and L. Andersson, "MPLS Extension Header," Internet Engineering Task Force, Internet-Draft draft-song-mpls-extension-header-02, Feb. 2019, work in Progress. [Online]. Available: https://datatracker.ietf.org/doc/html/draft-song-mpls-extension-header-02

[58] S. Hasan, P. Lapukhov, A. Madan, and O. Baldonado. (2017) Open/R: Open routing for modern networks. Accessed: 31 Jan. 2020. [Online]. Available: https://engineering.fb.com/connectivity/open-r-open-routing-for-modern-networks/

[59] The ZeroMQ Community. (no date) ZMTP CURVE. Accessed: 31 Jan. 2020. [Online]. Available: https://rfc.zeromq.org/spec/25/

[60] T. Nadeau, R. Aggarwal, K. Kompella, and G. Swallow, "Bidirectional Forwarding Detection (BFD) for MPLS Label Switched Paths (LSPs)," RFC 5884, Jun. 2010. [Online]. Available: https://rfc-editor.org/rfc/rfc5884.txt

[61] D. Katz and D. Ward, "Bidirectional Forwarding Detection (BFD) for Multihop Paths," RFC 5883, Jun. 2010. [Online]. Available: https://rfc-editor.org/rfc/rfc5883.txt

[62] Juniper. (2019) Understanding BFD for OSPF. Accessed: 28 Jan. 2020. [Online]. Available: https://www.juniper.net/documentation/en_US/junos/topics/concept/ospf-bfd-overview.html

[63] BeagleBone.org Foundation. (2018) SeeedStudio BeagleBone Green. Accessed: 14 Jan. 2020. [Online]. Available: https://beagleboard.org/green

[64] Raspberry Pi Foundation. (2020) About Us. Accessed: 14 Jan. 2020. [Online]. Available: https://www.raspberrypi.org/about/

[65] Infineon Technologies AG. (no date) About Us. Accessed: 21 Aug. 2019. [Online]. Available: https://www.infineon.com/cms/en/about-infineon/

[66] ——. (2019) BTT6050-1ERA. Accessed: 16 Jan. 2020. [Online]. Available: https://www.infineon.com/cms/en/product/power/smart-low-side-high-side-switches/automotive-smart-high-side-switch-profet/profet-24v/btt6050-1era/

[67] BeagleBone.org Foundation. (2015) SeeedStudio BeagleBone Green. Accessed: 17 Jan. 2020. [Online]. Available: https://

raw.githubusercontent.com/SeeedDocument/BeagleBone_Green/master/resources/BBG_SRM_V1a_20151009.pdf

[68] Coilcraft. (no date) The one company to call for all your magnetics. Accessed: 19 Jan. 2020. [Online]. Available: https://www.coilcraft.com/general/about.cfm

[69] ——. (2016) Bias Injection Choke. Accessed: 19 Jan. 2020. [Online]. Available: https://www.coilcraft.com/pdfs/ka4909.pdf

[70] ——. (2017) Mid-Span PoE Plus Magnetics. Accessed: 19 Jan. 2020. [Online]. Available: https://www.coilcraft.com/pdfs/fa2536.pdf

[71] ——. (no date) PoE Signal Path Transformer. Accessed: 20 Jan. 2020. [Online]. Available: https://www.coilcraft.com/pdfs/eth1-460.pdf

[72] Microsemi. (2014) IEEE802.3bt 4-Pair Power over Ethernet Task Force: 4P PoE systems use cases and proposed requirements. Accessed: 21 Jan. 2020. [Online]. Available: http://www.ieee802.org/3/bt/public/jan14/darshan_01_0114.pdf

[73] F. Capoano and A. Kaplan. (2015, Jul.) NetJSON: data interchange format for networks. Accessed: 13 May 2019. [Online]. Available: http://netjson.org/rfc.html

[74] C. Larsson, "Chapter 3 - Advanced Flow Theory," in *Design of Modern Communication Networks*, C. Larsson, Ed. Oxford: Academic Press, 2014, pp. 57 – 84. [Online]. Available: http://www.sciencedirect.com/science/article/pii/B9780124072381000038

[75] Gurobi Optimization, LLC. (no date) Gurobi Compute Server. Accessed: 21 Aug. 2019. [Online]. Available: https://www.gurobi.com/products/gurobi-compute-server-2/

[76] ——. (no date) Use Benchmarks to Find the Best Solver for Your Needs. Accessed: 31 Jan. 2020. [Online]. Available: https://www.gurobi.com/case_study/benchmarks/

[77] T. T. Group. (2019) Manpage of TCPDUMP. Accessed: 31 Jan. 2020. [Online]. Available: https://www.tcpdump.org/manpages/tcpdump.1.html

[78] P. Emmerich. (2018) libmoon. Accessed: 31 Jan. 2020. [Online]. Available: https://github.com/libmoon/libmoon/blob/master/examples/dump-pkts.lua

[79] B. Claise, "Cisco Systems NetFlow Services Export Version 9," RFC 3954, Oct. 2004. [Online]. Available: https://rfc-editor.org/rfc/rfc3954.txt

[80] P. Aitken, B. Claise, and B. Trammell, "Specification of the IP Flow Information Export (IPFIX) Protocol for the Exchange of Flow Information," RFC 7011, Sep. 2013. [Online]. Available: https://rfc-editor.org/rfc/rfc7011.txt

[81] A. Pyattaev. (2012) QoS metrics and requirements. Accessed: 31 Jan. 2020. [Online]. Available: http://www.cs.tut.fi/kurssit/TLT-2727/lect06.pdf

[82] J. Ahrenholz, "Comparison of CORE Network Emulation Platforms," in *Proceedings of the 2010 Military Communications Conference (MILCOM '10)*, Oct. 2010, pp. 166–171.

[83] J. Dugan, S. Elliott, B. A. Mah, J. Poskanzer, and K. Prabhu. (no date) What is iPerf / iPerf3 ? Accessed: 31 Jan. 2020. [Online]. Available: https://iperf.fr/

[84] M. Shen, H. Liu, K. Xu, N. Wang, and Y. Zhong, "Routing On Demand: Toward the Energy-Aware Traffic Engineering with OSPF," in *Proceedings of the 11th International IFIP TC 6 Conference on Networking (IFIP '12)*. Berlin, Heidelberg: Springer-Verlag, 2012, pp. 232–246.

[85] R. Carpa, O. Glück, and L. Lefevre, "Segment Routing based Traffic Engineering for Energy Efficient Backbone Networks," in *Proceedings*

of the 2014 IEEE International Conference on Advanced Networks and Telecommunications Systems (ANTS '14), Dec. 2014, pp. 1–6.

[86] J. Chabarek, J. Sommers, P. Barford, C. Estan, D. Tsiang, and S. Wright, "Power Awareness in Network Design and Routing," in *Proceedings of the 27th IEEE International Conference on Computer Communications (INFOCOM '08)*, Apr. 2008, pp. 457–465.

[87] Zhuochuan Huang, Chien-Chung Shen, C. Srisathapornphat, and C. Jaikaeo, "Topology Control for Ad hoc Networks with Directional Antennas," in *Proceedings of the Eleventh International Conference on Computer Communications and Networks (ICCCN '02)*, Oct. 2002, pp. 16–21.

[88] S. Zehl, A. Zubow, and A. Wolisz, "BIGAP – Seamless Handover in High Performance Enterprise IEEE 802.11 Networks," in *Proceedings of the 2016 IEEE/IFIP Network Operations and Management Symposium (NOMS '16)*, Apr. 2016, pp. 1015–1016.

[89] Ubiquiti Networks. (2018) airFiber 5XHD. Accessed: 17 Jan. 2020. [Online]. Available: https://www.ui.com/airfiber/airfiber5xhd/

[90] S. Klabnik and C. Nichols, *The Rust Programming Language*. USA: No Starch Press, 2018.

[91] R. Clipsham. (2015) Safe, Correct, and Fast Low-Level Networking. Accessed: 31 Jan. 2020. [Online]. Available: https://csperkins.org/research/thesis-msci-clipsham.pdf

[92] R. Clipsham, B. Decker, L. Färnstrand, R. Scheibe, and M. Gehring. (2018) libpnet. Accessed: 31 Jan. 2020. [Online]. Available: https://github.com/libpnet/libpnet/tree/master/benches

[93] D. P. García. (2016) Network namespaces. Accessed: 31 Jan. 2020. [Online]. Available: https://blogs.igalia.com/dpino/2016/04/10/network-namespaces/

[94] W. Bao, C. Hong, S. Chunduri, S. Krishnamoorthy, L.-N. Pouchet, F. Rastello, and P. Sadayappan, "Static and Dynamic Frequency Scaling on Multicore CPUs," *ACM Transactions on Architecture and Code Optimization*, vol. 13, no. 4, Dec. 2016.

[95] A. Abdelsalam, P. L. Ventre, A. Mayer, S. Salsano, P. Camarillo, F. Clad, and C. Filsfils, "Performance of IPv6 Segment Routing in Linux Kernel," in *Proceedings of the 14th International Conference on Network and Service Management (CNSM '18)*, Nov. 2018, pp. 414–419.

[96] C. Filsfils, D. Dukes, S. Previdi, J. Leddy, S. Matsushima, and D. Voyer, "IPv6 Segment Routing Header (SRH)," Internet Engineering Task Force, Internet-Draft draft-ietf-6man-segment-routing-header-26, Oct. 2019, work in Progress. [Online]. Available: https://datatracker.ietf.org/doc/html/draft-ietf-6man-segment-routing-header-26

[97] P. Emmerich, S. Gallenmüller, D. Raumer, F. Wohlfart, and G. Carle, "MoonGen: A Scriptable High-Speed Packet Generator," in *Proceedings of the Internet Measurement Conference 2015 (IMC '15)*, Tokyo, Japan, Oct. 2015.

[98] DPDK Project. (no date) Developer Quick Start Guide. Accessed: 23 Jan. 2020. [Online]. Available: https://www.dpdk.org/about/

[99] Intel Corporation. (no date) DPDK Boosts Packet Processing, Performance, and Throughput. Accessed: 23 Jan. 2020. [Online]. Available: https://www.intel.com/content/www/us/en/communications/data-plane-development-kit.html

[100] A. Abdelsalam, P. L. Ventre, C. Scarpitta, A. Mayer, S. Salsano, P. Camarillo, F. Clad, and C. Filsfils, "SRPerf: a Performance Evaluation Framework for IPv6 Segment Routing," 2020.

[101] S. Gallenmüller, P. Emmerich, R. Schönberger, D. Raumer, and G. Carle, "Building Fast but Flexible Software Routers," in *Proceed-*

ings of the ACM/IEEE Symposium on Architectures for Networking and Communications Systems (ANCS '17), May 2017, pp. 101–102.

[102] PC Engines GmbH. (no date) apu4d4. Accessed: 31 Jan. 2020. [Online]. Available: https://www.pcengines.ch/apu4d4.htm

[103] Ubiquiti Networks. airFiber Datasheet. Accessed: 21 Aug. 2019. [Online]. Available: https://dl.ubnt.com/datasheets/airfiber/airFiber_DS.pdf

[104] D. S. Miller, L. Torvalds, and E. W. Biederman. (2015) index : kernel/git/torvalds/linux.git. Accessed: 23 Feb. 2020. [Online]. Available: https://git.kernel.org/pub/scm/linux/kernel/git/torvalds/linux.git/commit/?id=e69724f32e62502a6e686eae36b7aadfeea60dca&h=master

[105] E. W. Biederman. (2015) Basic MPLS support. Accessed: 23 Feb. 2020. [Online]. Available: https://lwn.net/Articles/634788/

[106] L. Rizzo, "netmap: A Novel Framework for Fast Packet I/O," in *Proceedings of the 2012 USENIX Annual Technical Conference (ATC '12)*. Boston, MA: USENIX Association, 2012, pp. 101–112. [Online]. Available: https://www.usenix.org/conference/atc12/technical-sessions/presentation/rizzo

[107] V. J. Kathleen Nichols. (2012) Controlling Queue Delay. Accessed: 31 Jan. 2020. [Online]. Available: https://queue.acm.org/detail.cfm?id=2209336

[108] K. Nichols, V. Jacobson, A. McGregor, and J. Iyengar, "Controlled Delay Active Queue Management," RFC 8289, Jan. 2018. [Online]. Available: https://rfc-editor.org/rfc/rfc8289.txt

[109] T. Hoeiland-Joergensen, P. McKenney, D. Taht, J. Gettys, and E. Dumazet, "The Flow Queue CoDel Packet Scheduler and Active

Queue Management Algorithm," RFC 8290, Jan. 2018. [Online]. Available: https://rfc-editor.org/rfc/rfc8290.txt

[110] C. Kulatunga, N. Kuhn, G. Fairhurst, and D. Ros, "Tackling Bufferbloat in Capacity-limited Networks," in *Proceedings of the 2015 European Conference on Networks and Communications (EuCNC '15)*, Jun. 2015, pp. 381–385.

[111] M. A. Salam and Q. M. Rahman, *Electrical Laws*. Singapore: Springer Singapore, 2018, pp. 25–73. [Online]. Available: https://doi.org/10.1007/978-981-10-8624-3_2

[112] Infineon Technologies AG. (2014) PROFET+ Current Sense – What the designer should know. Accessed: 17 Jan. 2020. [Online]. Available: https://www.infineon.com/dgdl/Infineon-Infineon-PROFET+_Current_Sense-AN-v01_01-EN.pdf-AN-v01_00-EN.pdf?fileId=5546d4625b62cd8a015bcee64e406db4

[113] Texas Instruments. (2015) INA28x High-Accuracy, Wide Common-Mode Range, Bidirectional Current Shunt Monitors, Zero-Drift Series. Accessed: 17 Jan. 2020. [Online]. Available: http://www.ti.com/lit/ds/symlink/ina282.pdf

[114] ——. (2015) INA282 Error Analysis. Accessed: 17 Jan. 2020. [Online]. Available: http://www.ti.com/product/INA282

[115] D. Cameron. (2020) ElasticWISP Hardware. Accessed: 30 Jan. 2020. [Online]. Available: https://github.com/twominutesalad/ElasticWISP-Hardware