Universiteit Gent
Faculteit Ingenieurswetenschappen en Architectuur
Vakgroep Informatietechnologie

Foutbestendige toekomstige internetarchitecturen

Resilient Future Internet Architectures

Wouter Tavernier

Proefschrift tot het behalen van de graad van
Doctor in de Ingenieurswetenschappen:
Computerwetenschappen
Academiejaar 2012-2013

Universiteit Gent
Faculteit Ingenieurswetenschappen en Architectuur
Vakgroep Informatietechnologie

Promotoren: prof. dr. ir. Didier Colle
prof. dr. ir. Mario Pickavet

Universiteit Gent
Faculteit Ingenieurswetenschappen en Architectuur

Vakgroep Informatietechnologie
Gaston Crommenlaan 8 bus 201, B-9050 Gent, België

Tel.: +32-9-331.49.00
Fax.: +32-9-331.48.99

Proefschrift tot het behalen van de graad van
Doctor in de Ingenieurswetenschappen:
Computerwetenschappen
Academiejaar 2012-2013

Dankwoord

De wereld is gered! Mijn doctoraatsonderzoek is klaar, alle wereldproblemen zijn opgelost, het paradijs lonkt om de hoek! Zo droomde ik als doctoraatsstudent, en velen met mij, van het finale moment suprême waarop het proefschrift wordt afgeleverd ... of toch die eerste onderzoeksdag. Na de wittebroodsweken of -maanden ontdekte ik proefondervindelijk dat er zelfs voor netwerkonderzoekers geen kortste pad is naar die bestemming. Vele losse eindjes en doodlopende paden later, drong het steeds meer tot me door dat de triomf niet zozeer in de bestemming zit maar eerder in de boeiende en terzelfdertijd eindeloze zoektocht erheen.

Het epische karakter van deze zoektocht weerspiegelde zich nochtans niet alleen in de academische aspecten, maar vooral in de mensen en de omgeving die haar mogelijk maakten en ze een extra dimensie gaven.

Daarom wens ik eerst en vooral mijn promotoren prof. Didier Colle en prof. Pickavet te bedanken voor de onderhoudende en stimulerende discussies, en constructieve feedback doorheen mijn onderzoek. Bovendien heb ik mogen genieten van de aangename werkomgeving die de IBCN-onderzoeksgroep te bieden heeft. Ik wens prof. Piet Demeester dan ook uitdrukkelijk te bedanken voor het leiden van een zeer aantrekkelijke en terzelfdertijd productieve onderzoeksgroep. Dankzij hen bleef mijn onderzoek niet altijd binnenskamers, of zelfs binnen onze landsgrenzen. Ik kreeg de kans om mee te werken aan verschillende Europese onderzoeksprojecten en mijn resultaten te presenteren op internationale conferenties. In die context wens ik dan ook heel specifiek Dimitri Papadimitriou te bedanken voor de inspirerende discussies, ideeën en hulp bij het schrijven van papers. De onvergetelijke "quotes" konden me steeds opvrolijken, ook al waren ze niet altijd even "crystal clear".

Niet alleen de onderzoeksprojecten hadden een internationaal karakter, ook in het onderzoeksleven van elke dag kon ik mijn horizon verruimen via het contact met de Indische, Iranese en Japanse collega's. Graag wens ik dan ook Sachin, Abishek, Sahel en Sho Shimizu te bedanken voor het samenwerken en het uitwisselen van de leuke anekdotes. Minder ver van huis, maar daarom niet minder gewaardeerd, waren de momenten met de West-Vlaamse collega's Dimitri, Daan, Steven, Sofie D., Bart P., Sofie L. en Thijs. Bedankt om onze provincie in ere te houden! De Oost-Vlaamse bureaugenoten zorgden dan weer voor het aangename tegenwicht. Bedankt Ward, om de groene mens in mij naar boven te halen (of toch een beetje), Sofie D. om een streepje kunst in onze bureau te brengen, en de andere (ex-)collega's: Maarten, Sander, Willem, Florian om dit alles nog wat in toom te houden. Graag wens ik ook uitdrukkelijk de collega's Pieter, Jan, Philip, Daan,

Steven, Kristof S., Jeroen, Wim, Stijn, Kristof L. van het voormalige 3.15-bureau te bedanken voor de onvergelijkbare sfeer die we in onze bureau hadden.

Een onderzoeksgroep draait niet alleen op proffen en onderzoekers, IBCN zou niet dezelfde groep zijn zonder het behulpzame secretariaat van Martine en Davinia, de financiële administratie van Joke, Nathalie, Karien en Bernadette, en het werk van ons admin-team van Brecht, Bert, Jonathan en Joeri. Ook wens ik het iMinds als werkgever te bedanken voor het aanreiken van een geslaagd onderzoeksplatform waarin Vlaamse Universiteiten samen met nationale en internationale onderzoeksinstituten hoogtechnologisch onderzoek kunnen doen.

Gelukkig kreeg ik ook buiten de werkomgeving ruimschoots de kans om wat stoom af te blazen. Bedankt aan de tafeltennisvrienden om ervoor te zorgen dat ik bal vaker buiten het werk dan tijdens de werkuren missloeg. Dankzij de vrienden Gert, Bart, Mathias, Dieter, Benjamin en Diederik is mijn fysieke, muzikale en alcoholische niveau op peil gebleven: waarvoor uitdrukkelijk dank!

Tot slot kan ik niet genoeg benadrukken hoe belangrijk mijn dichtste familie voor dit verhaal was. Ma, pa, zus en Tim: bedankt voor jullie steun, vertouwen en kansen die ik van jullie heb gekregen, en de tijd die we samen mochten doorbrengen. Caroline en John, bedankt voor jullie schitterende dochter, de betrokkenheid, en de hartverwarmende zorgen voor Nora.

De allerbelangrijkste personen voor mij heb ik voor het laatst bewaard. Het is voor niemand een geheim. Marthe, bedankt dat ik met jou ”zij aan zij” door het leven mag gaan. Je steun was onbetaalbaar. Zonder jou was dit nooit gelukt. Nora, zolang ik thuiskom en jij spontaan begint te lachen, zijn voor mij alle wereldproblemen toch een beetje opgelost.

Tot slot nog aan allen die niet met naam en toenaam vermeld staan in dit dankwoord: een welgemeende dank-je-wel.

Brugge, oktober 2012
Wouter Tavernier

Table of Contents

List of Figures

List of Tables

List of Acronyms

A

ALFA	Automated Loop-Free Alternate
ARIMA	Auto-Regressive Integrated Moving Average
ARMA	Auto-Regressive Moving Average
ARP	Address Resolution Protocol
AS	Autonomous System

B

BFD	Bi-directional Forwarding Detection
BFS	Breadth-First Search
BGP	Border Gateway Protocol
bps	bit per second
B-VID	Backbone VLAN ID
BVLAN	Backbone VLAN

C

CAM	Content Addressable Memory
CE	Carrier Ethernet
CGE	Carrier-Grade Ethernet
CIDR	Classless Inter-Domain Routing
CL	Connection-less
CO	Connection-oriented
C-VID	Customer VLAN ID
CVLAN	Customer VLAN
CR	Constraint-based Routing

D

DHCP	Dynamic Host Configuration Protocol
DNS	Domain Name Server

E

E2E	End-to-end
ECMP	Equal Cost Multi-path Routing
EGP	Exterior Gateway Protocol
ELS	Ethernet Label Switching
E-LSP	Ethernet-Label Switched Path
EVL	Ethernet Virtual Line

F

FEC	Forwarding Equivalence Class
FIB	Forwarding Information Base
FTN	FEC-To-NHLFE

G

GARCH	Generalized Autoregressive Conditional Heteroskedasticity
GELS	GMPLS-enabled Ethernet Label Switching
GFP	Generic Framing Procedure
GMPLS	Generalized Multi-Protocol Label Switching
GRE	Generic Routing Encapsulation

I

IBBT	Interdisciplinary Institute for Broadband Technology
ICMP	Internet Control Message Protocol
IEEE	Institute of Electrical and Electronics Engineers
IETF	Internet Engineering Task Force
IGP	Interior Gateway Protocol
ILM	Incoming Label Map
IPFRR	IP-FastReRoute

IRTF	Internet Research Task Force
ISPF	Interactive Shortest Path First

K

kbps	kilobit per second

L

L1	Layer 1 (physical layer in TCP/IP model)
L2	Layer 2 (datalink layer in TCP/IP model)
L3	Layer 3 (network layer in TCP/IP model
LAN	Local Area Network
LBL	Link By Link (detection)
LER	Label Edge Router
LFA	Loop-Free Alternate
LFIB	Linecard FIB
LFN	Loop-Free Node
LPM	Longest Prefix Match
LSA	Link State Advertisement
LSDB	Link State DataBase
LSR	Label Switching Router
LSP	Label Switched Path
LSU	Link State Unit

M

MAN	Metropolitan Area Network
MEF	Metro Ethernet Forum
MPLS	Multi-Protocol Label Switching
MTBF	Mean Time Between Failures
MTTF	Mean Time To Failure
MTTR	Mean Time To Repair

N

NHLFE	Next Hop Label Forwarding Entry

O

OAM	Operations, administration and management
OSI	Open Systems Interconnection
OSPF	Open Shortest Path First
OSPF-TE	Open Shortest Path First-Traffic Engineering
OTN	Optical Transport Network

P

PB	Provider Bridge
PBB	Provider Backbone Bridge
PBB-TE	Provider Backbone Bridge-Traffic Engineering
PCE	Path Computation Engine
PDU	Protocol Data Unit
POS	Packet Over SONET/SDH

Q

QoS	Quality of Service

R

RIB	Routing Information Base
RSTP	Rapid Spanning Tree Protocol
RSVP	ReSource reserVation Protocol
RSVP-TE	ReSource reserVation Protocol-Traffic Engineering
RUP	Router Update Process

S

SDH	Synchronous Digital Hierarchy
SLA	Service Level Agreement
SONET	Synchronous Optical Networking
SPT	SPanning Tree
STP	Spanning Tree Protocol

S-VID	Service VLAN ID
SVLAN	Service VLAN

T

TCAM	Ternary Content-Addressable Memory
TE	Traffic Engineering
TCP	Transmission Control Protocol
TDM	Time Division Multiplexing
TTL	Time To Live

U

UDP	User Datagram Protocol

V

VCAT	Virtual conCATenation
VID	VLAN ID
VLAN	Virtual Local Area Network
VLSR	Virtual Label Switch Router
VPN	Virtual Private Network
VPLS	Virtual Private Line Service
VPWS	Virtual Private Wire Service

W

WAN	Wide Area Network
WDM	Wavelength Division Multiplexing

Samenvatting
– Summary in Dutch –

Communicatienetwerken zijn verweven met bijna ieder aspect van ons privé- en werkleven. Nieuws uit de wereld, maar evengoed uit onze sociale kring bereikt ons via het web, we gebruiken IPTV op onze televisie, laptop of tablet, en delen allerhande informatie en bestanden via cloud services. De nood aan bandbreedte groeit dan ook jaarlijks aan een tempo van 30 tot 40 procent. Bovendien stellen allerhande realtime toepassingen zoals (video-)telefonie steeds strengere eisen aan de beschikbaarheid van de netwerkinfrastructuur. De vuistregel van 'vijf negens' stelt dat we slechts 5 minuten op jaarbasis aan netwerkpannes tolereren (99.999 procent van de tijd).

Oorspronkelijk werd het internet nochtans niet ontwikkeld om aan dergelijke strenge eisen te voldoen. Het was immers onvoorspelbaar dat een experimentele netwerkinfrastructuur tussen een viertal Amerikaanse universiteiten uit de jaren '70 zou uitgroeien tot het internet van vandaag met 2 tot 3 miljard eindgebruikers, en miljoenen tussenliggende switches en routers. De technische uitdagingen die dit met zich meebrengt zijn dan ook niet te verwaarlozen en hebben geleid tot een kluwen van verschillende technologische oplossingen van al bijna evenveel verschillende fabrikanten. Een hedendaagse router is uitgegroeid tot een duur en complex toestel dat zowel een grote datadoorvoer moet garanderen als ondersteuning moet bieden voor een grote verscheidenheid aan controleprotocollen voor routering en signalisatie. Het voorzien van efficiënte en foutbestendige netwerkconnectiviteit wordt hierdoor steeds moeilijker voor de hedendaagse netwerkbeheerder van grotere communicatienetwerken. Om toch aan de strenge eisen te voldoen, wordt dan ook vaak voor de eenvoudigste en meestal duurste oplossing gekozen: overvoorziening van het netwerk door het reserveren van te grote bandbreedtes en ondergebruikte reservepaden.

In een eerste studie onderzoekt dit proefschrift daarom de complexe netwerkstructuur van het internet en de daarbijhorende problemen met betrekking tot routering en foutherstel. Hierbij stellen we vast dat er een kloof te vinden is op gebied van kostprijs en complexiteit tussen drie technologieën: de Ethernet-technologie in LAN-netwerken (Local Area Network), de IP-gebaseerde routers, en de circuitgebaseerde optische technologie in aggregatie- en kernnetwerken (de internetbackbone). Anderzijds merken we dat er een spanning en vervaging optreedt met betrekking tot de functionaliteit tussen diezelfde technologieën. Voor grotere bedrijfsnetwerken, aggregatienetwerken of delen van de internetbackbone, ontstaat

dan ook het idee dat Ethernet-technologie met een beperkt aantal uitbreidingen een (kost-)efficiënter alternatief zou kunnen bieden op de complexere IP-gebaseerde of SDH-gebaseerde technologie (Synchronous Digital Hierarchy). Vanuit dit perspectief wordt in deze thesis Ethernet Label Switching (ELS) uitgewerkt en geëvalueerd. Deze technologie laat toe om connecties op te zetten op basis van het bestaande concept van Virtual Local Area Networks (VLAN's). In een emulatieomgeving werd hiervan een prototype geïmplementeerd, en werd onder andere aangetoond dat het resulterende ontwerp en de implementatie ervan in staat zijn om netwerkenfouten te herstellen in minder dan 50 ms, gebruikmakend van het foutdetectieprotocol BFD (Bi-directional Forwarding Detection) en de GMPLS-controleprotocollen (Generalized Multi-Protocol Label Switching). Samen met de vrijheid die dergelijke technologie aanbiedt om verschillende paden doorheen het netwerk te configureren (traffic engineering), kan de resulterende technologie gebruikt worden als geschikt alternatief in transportnetwerken. Dit geeft aanleiding tot de term Carrier Ethernet. In de context van dit onderzoek werd zowel de point-to-point connectiviteit als de multipoint connectiviteit geëvalueerd en gedemonstreerd op verschillende internationale evenementen. In samenwerking met de Japanse Keio universiteit werd bovendien de interoperabiliteit van de geïmplementeerde Ethernet Label Switching-technologie aangetoond.

Waar (kost-)efficiënte en foutbestendige connectiegeoriënteerde switching centraal staat in het eerste deel van het geleverde onderzoekswerk, wordt in het tweede luik van het onderzoek de nadruk gelegd op adaptief en automatisch configurerend foutherstel binnen connectieloze IP-netwerken. Vaak wordt er nog geopteerd voor het gebruik van connectieloze IP-routering op basis van traditionele kortste-pad-routeerprotocollen zoals Open Shortest Path First (OSPF) omwille van de schaalbaarheid. Deze protocollen configureren automatisch kortste-pad-routes in de verspreide routers van het netwerk. Helaas is deze oplossing niet zomaar in staat om "carrier grade" foutherstel te realiseren, omdat het vinden van nieuwe kortste paden in een veranderde netwerktopologie vaak seconden tot minuten in beslag kan nemen. Daarom wordt in dit proefschrift een techniek voorgesteld en geëvalueerd die zowel snelle foutopvang kan garanderen, als volautomatisch geconfigureerd en geactiveerd ingezet kan worden met behulp van foutdetectietechnieken. Omdat routers in IP-netwerken dikwijls worden verbonden via multipoint Ethernet-netwerken, werd in het onderzoek een Ethernet-gebaseerde uitbreiding op het detectieprotocol Bi-directional Forwarding Detection (BFD) ontworpen en geëvalueerd op de testinfrastructuur van de onderzoeksgroep. Hierdoor kan foutdetectie over deze media efficiënter verlopen. In combinatie met de voorgestelde herrouteringstechniek, kunnen op die manier zelfs netwerken van een paar honderd routers voorzien worden van kortstepadrouting en snel herstel van mogelijke netwerkfouten, met een minimum aan configuratie.

Naast het automatiseren van snelle foutvang, gaat het onderzoek ook na in welke mate het gebruik van "netwerkverkeersinformatie" kan bijdragen tot het efficiënter opvangen en behandelen van netwerkfouten. Het vertrekpunt hierbij is dat bij het optreden van een netwerkfout vaak meerdere elektronische verkeersstromen worden aangetast. Waar bestaande methoden voor foutafhandeling zich vaak toe-

spitsen op enerzijds de snelle foutdetectie, of anderzijds het snel activeren van een alternatief pad, is het de bedoeling van deze studie om vanuit het perspectief van een enkele router zo snel mogelijk die verkeersstromen te herstellen die er het meeste nood aan hebben. Hiermee wordt verwezen naar de verkeersstromen die een grote bandbreedte consumeren en dus mogelijks tot aanzienlijk pakketverlies kunnen leiden op het ogenblik dat hun pad doorheen het netwerk wordt onderbroken. Dit lijkt eenvoudiger dan het is, omwille van de grote verkeersvariatie die kan optreden in erg korte tijd. In deze context werd enerzijds een wiskundige formulering opgesteld om het mogelijke pakketverlies te karakteriseren bij verschillende herstelvolgordes, en werd anderzijds aangetoond dat het pakketverlies resulterend bij een netwerkfout, kan gereduceerd worden aan de hand van een heuristiek die gebruik maakt van een voorspellend ”netwerk-verkeersmodel”.

Met het geleverde onderzoek reikt dit proefschrift een aantal componenten aan voor meer (kost-)efficiënte en meer beheersbare routering en foutherstel in het kader van de steeds complexer wordende internetinfrastructuur. Hierbij wordt aangetoond dat connectiegeoriënteerde technologieën zoals Ethernet Label Switching voldoende routeer- en foutherstelmogelijkheden bieden, en dat de concepten van automatische configuratie en adaptatie met betrekking tot het getransporteerde netwerkverkeer van bijzondere meerwaarde kunnen zijn bij het afhandelen van netwerkfouten.

Summary

Communication networks impact almost every aspect of our private and business environment. World news, as well as news within our social circles, reaches us via the web. We use IPTV on our television, laptop or tablet, and share various information and files via cloud services. It is no surprise that the resulting need for bandwidth is growing at an annual pace of 30 to 40 percent. In addition, all kinds of real-time applications such as (video-) telephony impose ever stricter demands on the availability of the network infrastructure. The rule of thumb of ”five nines” suggests that we only tolerate 5 minutes of network outage on an annual basis (99.999 percent of the time).

However, originally the Internet was not developed to meet such stringent requirements. It could not be predicted that an experimental network between four American universities from the 70s would grow to the Internet of today with 2 to 3 billion end users and millions of intermediate switches and routers. The technical challenges that emerge from this situation cannot be ignored, and have led to a tangle of different technological solutions from almost as many different manufacturers. A contemporary router has become an expensive and complex device that combines high data throughput with support for a wide variety of control protocols for routing and signaling. Providing efficient and fault-tolerant network connectivity is becoming increasingly difficult for the network administrator of larger networks. To still meet the stringent requirements, the easiest and often most expensive solution is used: over-provisioning of the network by providing too large bandwidths and under-utilized backup paths.

In a first study this thesis examines the complex network structure of the Internet and the associated problems related to routing and network recovery. We note that in terms of complexity, cost and efficiency, a large gap occurs between three technologies: Ethernet technology in LANs (Local Area Network), IP-based routing and the circuit-switched optical technology in aggregation and core networks (the Internet backbone). On the other hand, a tension and blurred situation is emerging between the same technologies with respect to their functionality. For larger enterprise networks, aggregation networks or parts of the Internet backbone, this gives rise to the idea that Ethernet technology could possibly be upgraded with a limited number of extensions, forming a (cost-) efficient alternative for more complex and expensive IP-based or SDH-based technology (Synchronous Digital Hierarchy). From this perspective, Ethernet Label Switching (ELS) was further designed and evaluated in this thesis. This technology allows establishing connections, relying on the existing concept of Virtual Local Area Networks

(VLANs). An ELS prototype is developed in an emulation environment, and it has been demonstrated that the resulting design and implementation is capable of performing network recovery in less than 50 ms, relying on Bi-Directional Forwarding Detection (BFD) for failure detection, and Generalized Multi-Protocol Label Switching (GMPLS) for controlling the network. Along with the support for traffic engineering, ELS can be used as a suitable alternative in transportation networks, hence the term Carrier Ethernet. The research in this context has evaluated both point-to-point connectivity as well as multipoint connectivity, and has been demonstrated on several European events. In cooperation with the Japanese Keio University, the interoperability of the Ethernet Label Switching implementation has been demonstrated.

Whereas (cost-) efficient and fault-tolerant connection-oriented switching is the central topic of the first research part, the second part of the research focuses on adaptive and self-configuring network recovery within connectionless IP networks. Many networks still rely on connectionless IP routing using traditional shortest-path routing protocols such as Open-Shortest Path First (OSPF), because of its scalability. Routing protocols automatically configure shortest-path routes between distributed routers in the network. However, by default, these protocols are not capable of providing ”carrier grade” network recovery, as the time required to find new shortest paths in changed network topologies can take seconds to minutes for these protocols. In this context, a technique is proposed and evaluated which is self-configuring and can deliver fast network recovery of almost any network link failure. Because routers in IP networks are often connected via multipoint Ethernet networks, an Ethernet-based extension of the Bi-Directional Forwarding Detection (BFD) protocol has been designed and evaluated on the test infrastructure of the research group. This allows more efficient failure detection over these network segments. The resulting scheme enables networks of routers of up to hundreds of routers, to rely on traditional shortest path routing protocol, and still providing carrier-grade network recovery with a minimum of configuration.

In addition to self-configuring and fast network recovery, the research also examines to what extent network traffic information can contribute to efficient IP-based rerouting. This stems from the observation that upon a network failure often more than one network traffic flow is affected, for which the corresponding routing entries must be updated. Whereas most existing network recovery research focuses on either rapid fault detection or fast activation of an alternative path, the aim of this study is to recover first those network traffic flows which need it most. The latter refers to the traffic flows consuming large bandwidths, and thus potentially can lead to large amounts of packet loss when the network path they use is broken. This is not a trivial task because of the large traffic variation that can occur in very short time. For this purpose, a mathematical formulation is first made to characterize the packet loss for different traffic-driven rerouting schemes. In a second phase it is demonstrated that the packet loss resulting of the rerouting event upon a network failure possibly can be reduced by means of a heuristic relying on a predictive ”network traffic model”.

With the performed research, this dissertation proposes a number of compo-

nents for (cost-)efficient and manageable routing and error recovery in the context of the increasingly complex Internet infrastructure. The documented research illustrates that connection-oriented technologies such as ELS provide sufficient carrier-grade routing- and network recovery facilities and that the concepts of automatic configuration and adaption to monitored network traffic can be of added value in handling network failures.

1 Introduction

Communication networks have become an indispensable part of our lives. The Internet of today serves all kinds of applications for both professional and leisure purposes. Many businesses count and trust on the quality, security and reliability of the Internet infrastructure. However, the Internet infrastructure was not designed to meet these high requirements (best-effort service). As a consequence, a wide spectrum of technological solutions, implemented by equally many vendors, have been proposed and deployed to deal with this situation. The combination of the exponential network growth and this heterogeneous network landscape makes it increasingly difficult to guarantee the required network reliability and quality of service.

This chapter introduces network recovery within the context of the current Internet technology infrastructure. It indicates the emerging challenges with respect to routing and recovery, and sketches the outline of the research documented in this dissertation.

1.1 The network infrastructure of the Internet

The Internet can be segmented into five topologically determined network parts (see Figure 1.1): the home network, the access network, the (metro-) aggregation network, the core network (backbone), and the campus network. Each of these network parts has its own function.

Home (or small company) networks typically consist of a limited number of

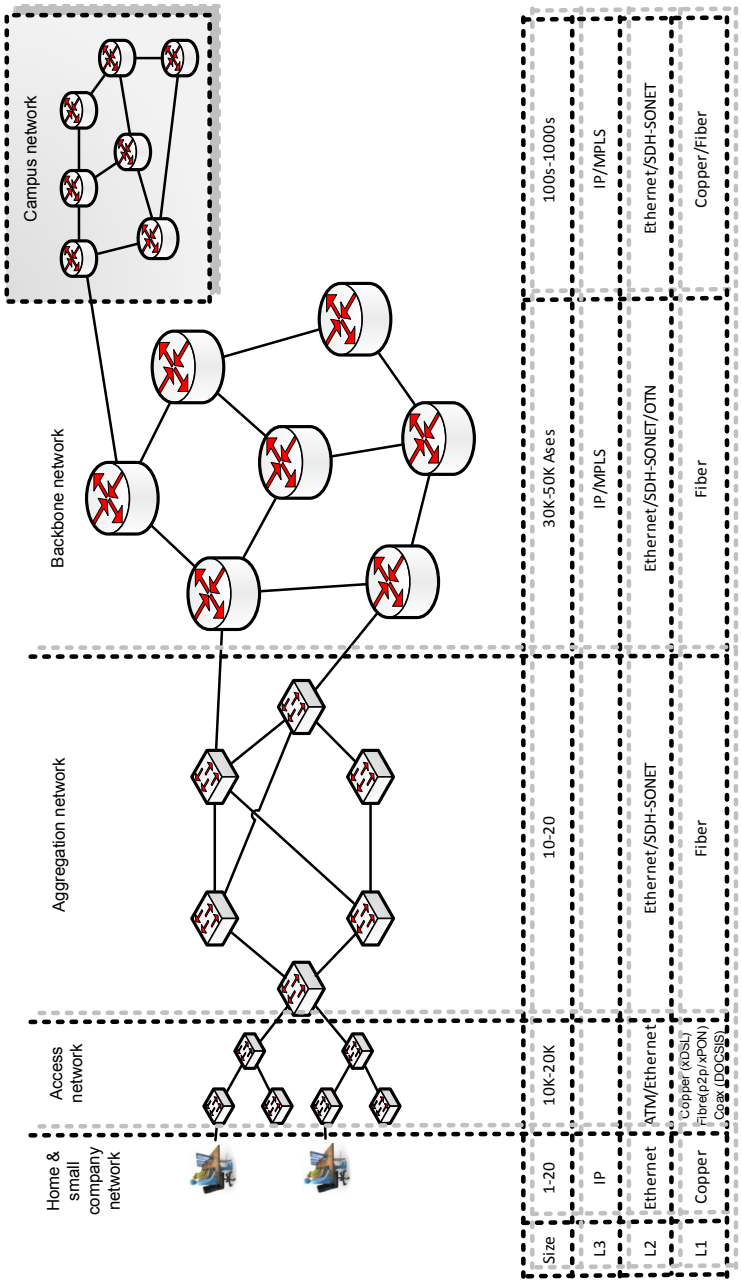


Figure 1.1: Overview of the Internet infrastructure

end hosts (pc's or other devices) which are interconnected via a wired or wireless Local Area Network (LAN).

The (wired) *access network* attaches thousands of home and/or small business networks via service providers to the Internet. This network normally has a tree-like structure, and spans a couple of kilometers (scale of a city). For this reason, access networks are often referred as the "first mile".

Aggregation or metro networks interconnect access networks in a ring, star or slightly meshed network topology of tens of network nodes. These networks can accommodate up to 200K customers, and have a regional coverage. These networks aggregate all traffic from access networks towards (metro-)core networks, and typically span areas with diameters of up to 50 km. Both access and aggregation networks are owned by Internet Service Providers (ISP).

The *core or backbone network* interconnects the aggregation networks and forms the core of the Internet network, resulting into a large meshed topology of about 40K autonomous systems (AS) or domains, structured in 'tier levels'. These ASes are owned by Internet Backbone Providers (IBP) forming the main structure of the Internet. Every AS forms an administratively autonomous entity under the operation of a single party, and can consist of tens to thousands of nodes. These networks can span distances of hundreds of kilometers. Sometimes an intermediate level is introduced before the Internet core, referred as the *metro-core network*.

Some universities or corporate-sized companies (enterprises) run large *campus networks* of up to thousands of switches and hundreds routers [1, 2]. Data centers can be considered to belong to this category [3]. These networks normally have direct access to the Internet core network (or to an aggregation network).

1.2 Network recovery and requirements

Communication technology plays a crucial role in our social and economical activities. Telephone service, email and Internet connectivity in general are for many businesses the communication channel of choice. A disruption in these services may completely interrupt the company's internal and external communication channels, causing a significant loss in activity, customers and corresponding revenue. From this perspective, the well-known saying could be changed to: "*connectivity is money*". Therefore, availability requirements have become a crucial part of the *Service Level Agreements (SLA)* between companies (customers of the network) and network providers.

In this context, *network availability* can be defined as the probability of a network element to be operational at one particular point in time. Five nines avail-

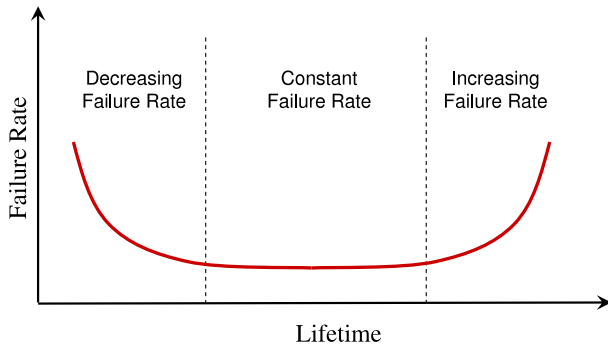


Figure 1.2: Failure rate vs. the lifetime of network equipment

	MTBF	***MTTR***
cable cut per 1000 km on long distance cables	50 days-200 days	days - weeks
webserver	$10^4 - 10^6$ hours	1 hour
Ethernet switch	$10^4 - 10^6$ hours	2 hours
IP interface card	$10^4 - 10^5$ hours	2 hours
IP router	$10^5 - 10^6$ hours	2 hours
ATM switch	$10^5 - 10^6$ hours	2 hours
SDH/SONET DXC	$10^5 - 10^6$ hours	4 hours
WDM OXC	$10^5 - 10^6$ hours	6 hours

Table 1.1: Ranges of MTBF and MTTR values of network equipment, taken from [4, 5]

ability (99.999 percent of the time) is often used as a benchmark for carrier-grade[1] availability. This means that only about 5 minutes of network outage is tolerated in a year.

In practice, the *mean network availability* (MNA) can be expressed in terms of the *Mean Time To Failure* (MTTF), the *Mean Time Between Failures* (MTBF), and the *Mean Time To Repair* (MTTR), as follows.

$$MNA = \frac{MTTF}{MTTF + MTTR}$$

The relationship between MTTF, MTBF and MTTR is depicted in Figure 1.3. The mean unavailability of a network (MNU) is characterized by the following fraction. The MTTF is a function of the *failure rate* of the equipment. In Table 1.1 a list of MTBF and MTTR numbers is shown for typical network equipment.

$$MNU = 1 - MNA = \frac{MTTR}{MTTF + MTTR} = \frac{MTTR}{MTBF}$$

[1]The term ”carrier-grade” is often used for indicating that some technique or technology is able to provide such a high level of service that the technology could be used by a network provider for *carrying* network traffic of customers. The term can be interchanged with ”extremely reliable, well tested and proven in its capabilities”.

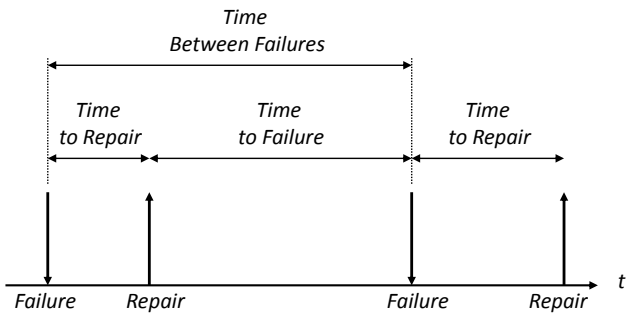


Figure 1.3: Relationship between concepts in availability terminology

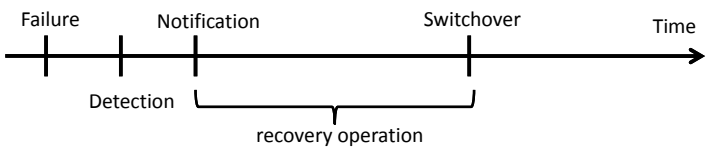


Figure 1.4: The events in the process of network recovery

In many cases it is assumed that the failure rate is time dependent. This implies that the probability of a failing element changes in relationship with its lifetime. In this case, the function is referred as the *Hazard function* $h(t)$ (the rate of failure of an element given that this element has survived this long), which often follows a *bathtub curve* (see Figure 1.2). The bathtub curve follows roughly three stages: i) a higher, but decreasing failure rate in the beginning due to "infant mortality" characteristics, ii) a more or less constant failure rate because of 'random failures' during the 'useful life' of the material, and iii) again an increasing failure rate due to the "wear out" of the material. The broader statistical model (capable of reproducing a bathtub shape) which is usually applied for characterizing the time between network failures, is the Weibull distribution [6]. The number of failures per component often follows a power-law [6].

Network survivability is a term to indicate the ability of a network to recover traffic in the event of a network failure, causing few or no consequences for the users. The benchmark for carrier-grade survivability is around 50 ms. This means that network failures need to be recovered within this time (including all phases of the network recovery process indicated below).

The process of *network recovery* is typically decomposed into the following events and associated time intervals (see Figure 1.4):

- *failure event*: the event through which connectivity is lost
- *failure detection*: the event at which the failure is detected by the installed control mechanisms
- *failure notification*: the event at which the recovery procedures are notified to be activated
- *failure recovery*: the event at which the network connectivity is repaired

Network recovery processes can be either *pre-planned or dynamic*. In the first case, the path of the recovery flow is calculated in advance for all failure scenarios, while in the second this is applied on the fly[2].

Along another dimension, recovery processes can implement protection or restoration. Upon failure detection, *network protection* only needs a very limited amount[3] or no signaling to activate the recovery path. This enables very fast switch-overs and network recovery. *Network restoration* still needs signaling (although the failure scenario's could have been pre-planned) upon failure detection.

In general, protection is not shared. In $1+1$ *protection*, the signal is duplicated over two pre-established alternative paths. The end-node thus receives two signals, and can select the signal with the best quality. In *shared protection*, backup paths are used for the protection of multiple paths . This allows to reduce the capacity needed for the backup path, assuming that failures are independent. In $M : N$ *shared protection*, M protection paths are used to protect N different working paths between a given node pair.

As will be further detailed in the next sections, network functionality follows a layered approach in which network technologies at different layers potentially can perform network recovery. Because this can lead to the duplication of network functions, coordination strategies between these layers can greatly increase the efficiency of the network recovery [4]. In this dissertation we will further focus on single network layer recovery in the context of Ethernet and IP technology.

1.2.1 Recovery scopes

The recovery scope relates to the (spatial) locality of the recovery. When *end-to-end (or global) recovery* is applied, an entire disjoint path is required to provide recovery. If *local recovery* (e.g., link, node or segment recovery) is used, a local alternate path can bypass the network failure to recover network connectivity.

[2] a part from signaling, as will be discussed later

[3] the required signaling for protection typically only relates to the synchronization of the protecting path, or reversion of it.

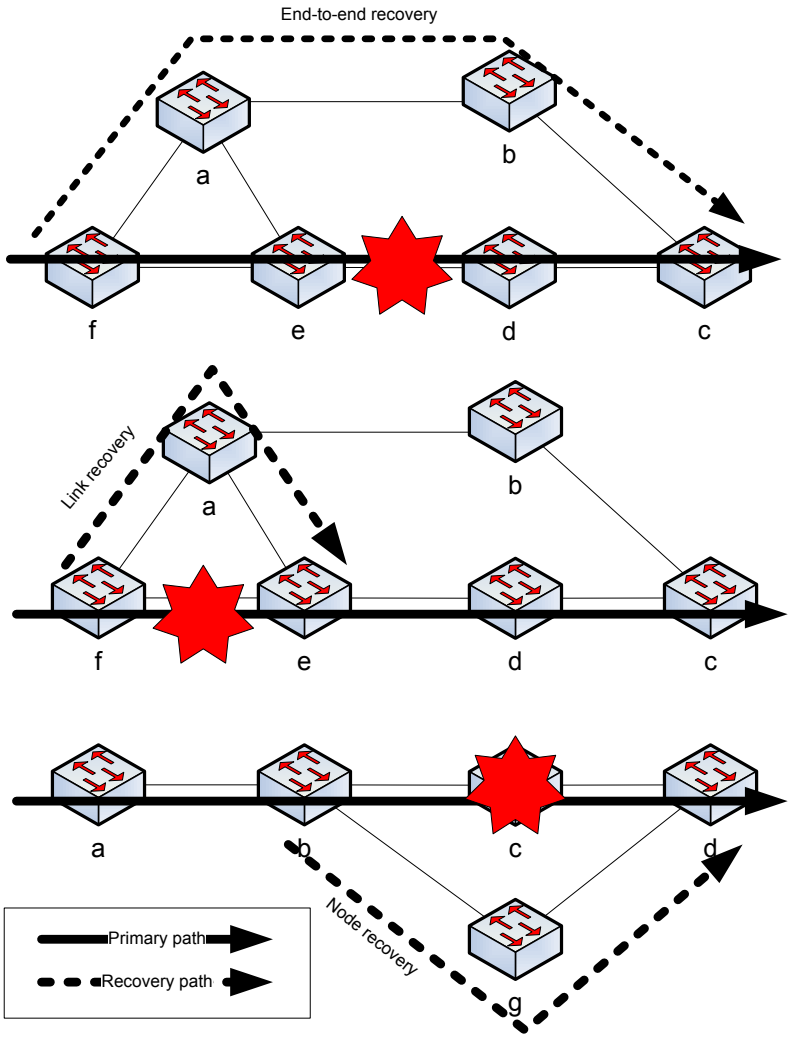


Figure 1.5: End-to-end vs. link and node recovery

	Medium	*Bandwidth*	*Delay/recovery*	*Jitter*	*Error*
End-user web applications	Audio	4-13 Kbps	<1s	<1ms	<3% FER
	Data	NA	<4s	NA	0
Streaming	Audio	5-128 Kbps	<10s	<2s	<1% loss
	Video	20-384 Kbps	<10s	<2s	<2 % loss
	Data	<384 Kbps	<10s	NA	0
Interactive streaming applications	Audio	4-25 Kbps	<150ms	<1ms	<3% FER
	Video	32-384 Kbps	<150ms	NA	<1% FER
	Data	NA	<250ms	NA	0
Metro core networks	BULK	NA	<50ms	NA	0
Industrial Ethernet Network	Data	64 Kbps-3.2 Mbps	5-10ms	<1ms	0

Table 1.2: QoS requirement overview (from [7])

Figure 1.5 illustrates the difference between end-to-end, link and node recovery in the context of network connectivity between two nodes: f and c. The topmost figure illustrates how the path (f,e,d,c) can be recovered via an end-to-end backup path (f,a,b,c). Link recovery is illustrated in the second example, where the connectivity between node f and e is recoverable via node a. At last, the third figure illustrates how node c can be bypassed in case of node failure, via node g. Most carrier-grade switching and routing technologies support these forms of recovery. Further information about network recovery terminology can be found in [4].

1.2.2 QoS requirements of network services

The required level of Quality Of Service (QoS) and reliability of the network depends on the environment and the specific application needs. Typical requirements in the networks that are studied in this dissertation are depicted in Table 1.2 obtained from [7]. The table illustrates that typical end-user web-applications can deal with network delays in the order of 1 to 4 seconds, together up to 3 percent Frame Error Rate (FER) which can be accommodated through correction mechanisms to improve the perception at end-user points. Interactive streaming applications impose stronger delay requirements with delays of maximally 150 to 250 ms. The most severe requirements can be found in Industrial Ethernet networks where machines in factories are controlled via networks which can only tolerate up to 10 ms delays.

1.3 Network layering and technology overview

From a technological point of view, the Internet is structured according to the layered TCP/IP model, containing five layers as depicted in Figure 1.6. In this model, every lower layer in the model provides service to a higher level.

The central layer is the *network layer* (layer 3 or L3), implemented by Internet Protocol (IP [8]). The core function of this protocol is to provide the following service to higher layers: i) connection-less connectivity between end-hosts

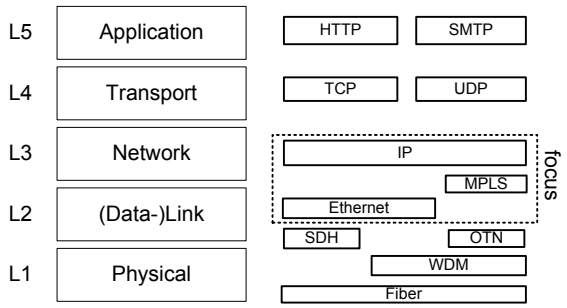


Figure 1.6: Technologies within the TCP/IP model layering

(packet-based messaging), ii) node addressing and address aggregation of end-hosts and intermediate nodes, and iii) efficient message forwarding and path determination (routing) between source and destination nodes via intermediate gateways or routers.

The transport and application layers (layer 4-5) allow reliable end-to-end communication between end-hosts, allowing typical Internet services and applications such as web-browsing and e-mail.

Technologies at lower layers of the TCP/IP model provide (reliable) connectivity between a small number of network nodes connected to the same network segment (layer 2 or *data link layer*), or ensure physical transmission of the data over a given medium (layer 1 or *physical layer*).

In practice, the functionality provided by technologies is difficult to map to the layered model. Nevertheless, Figure 1.6 provides a first orientation of the set of technologies and protocols of interest within the layered TCP/IP model. The focus of the research in this dissertation is on layer 3 and layer 2 technologies. Technologies are related to the type of network, as next sections and Figure 1.1 indicate.

A more fundamental way to structure the spectrum of network technologies is by characterizing their operational framework as determined by the networking mode and the switching type.

Networking modes In *connection-oriented* (co) networks, a connection needs to be set up from beforehand to enable connectivity between two or more nodes in the network. This involves either manual configuration, centralized management, or the use of a *signaling protocol*[4] to ensure that information can be sent between nodes. All data which makes use of this connection will be handled in a similar way (e.g., routing).

[4]A signaling protocol is used to configure the state of a distributed set of nodes, e.g. for setting up connections, setting up backup connections, activating backup connections, etc.

In *Connection-less* (cl) networks, the forwarding decision of intermediate network nodes is a purely local decision, without requiring to set up a connection before communication can be started.

Switching types The switching type of a technology determines how the forwarding[5] decision is taken. In *packet-switched* networks, forwarding is performed on a per-packet basis, and the forwarding decision is determined by a feature of the received packet. In the context of IP routing, the forwarding decision is based on the IP destination address from IP packet header of the received packet (see next section).

In *circuit-switched* networks, the forwarding decision takes place at circuit-level (switching based on the ”position” of arriving bits, where ”position” is defined by space (port), time and wavelength).

Both switching types have their advantages and drawbacks, as will become clear in the next sections.

Data plane vs. control plane Communication networks do not only transport end-user data, but also need to exchange control-related data and implement related functionality to ensure that the network operates as desired. All functionality that relates to the transport of end-user data is abstracted into the *data or forwarding plane*, all other/control-related functionality leads to the *control plane*. Sometimes management-related operations are not considered as control functionality, and therefore are referenced as being part of the *management plane*.

1.3.1 IP routing

The IP protocol[6] forms the common basis for almost all packet-switched connectivity between different types of networks of the Internet. IP-based network connectivity is one of the most scalable network layer technologies currently available. In IP-based networks, data traffic between end-hosts is split into packets or datagrams which travel through a network of routers (intermediate IP-enabled network nodes). Routers process incoming packets according to the *store-and-forward* principle based on the destination address contained in the IP packet header, as shown in Figure 1.7. IP routing is the technology of choice to interconnect large campus/enterprise networks, as well as the Internet backbone.

[5]Forwarding refers to the process where a data unit (either a packet or a signal) is received on a given port of a network device and needs to be transmitted towards one or more other port(s).

[6]either IPv4 [8], or IPv6 [9]

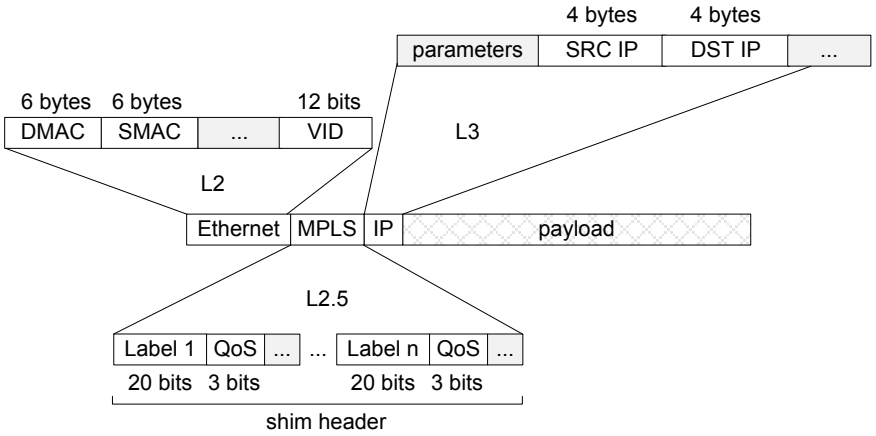


Figure 1.7: Packet headers of Ethernet at layer 2, MPLS at layer 2.5 and IP at layer 3

1.3.1.1 IP addressing, aggregation and forwarding

IP networks use IP addressing for the identification and location of network nodes (i.e., interfaces). Every interface interconnecting end hosts[7] and routers in the network receives an IP address, a binary identifier of 32 bits in IPv4. These addresses are commonly written in the dot-notation, consisting of 4 numbers between 0 and 256 (e.g., 196.2.3.234). IP addresses can be grouped into spatially clustered subnets. This is reflected in the IP address by a *(subnet) prefix*, referring to a fixed number of leftmost bits within the IP address. The remaining (right-most) bits of the IP-address identify an interface within that subnet. The subnet part can have any length according to the Classless Inter-Domain Routing (CIDR [10]) which is used to structure the IP address space of the Internet. For example the prefix 196.2.3.0/24 refers to the subnet of all IP-addresses in the range 196.2.3.0 to 196.2.3.255 (256 potential addresses).

Every interface connected to the Internet, receives an IP address. Service providers and large organizations are typically assigned a contiguous block of addresses, as determined by a prefix of fixed length.

Routers, which are intermediate nodes of the Internet, forward IP packets based on the IP destination address indicated in the IP packet header (Figure 1.7). The concept of subnets and prefixes allows IP routers to reduce the required memory for taking these forwarding decisions. Because IP addresses are usually assigned hierarchically in contiguous blocks, it often happens that a router can reach an

[7] an end-user device is typically connected to the network via one link

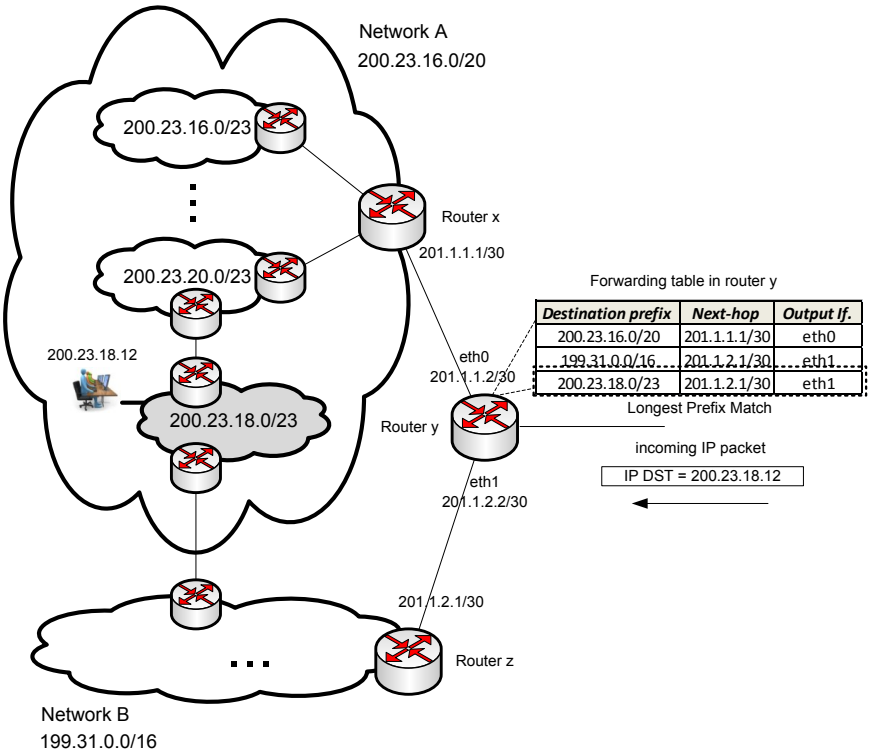


Figure 1.8: Address/route aggregation and longest prefix matching

entire subnet via the same interface. In this case, routers only need to store the corresponding prefix in their forwarding table[8], to reach the entire network part covered by the prefix. This allows an efficient way of *route aggregation*. The resulting tree-like structure would allow very efficient address/route aggregation in routers. The reachability of networks can be announced via these prefixes, instead of a list of IP addresses. However, in practice the Internet is less hierarchical, and parts of networks have shortcuts (shorter paths) to smaller subnets. To enable efficient use of these connections, routers implement a *longest-prefix match algorithm* (LPM) to find prefix in the routing table which best matches the destination IP address of the received packet which needs to be forwarded. High-end routers implement these lookups in hardware using Ternary Content-Addressable Memory (TCAM). Routers in the Internet backbone can contain up to hundreds of thousands of forwarding entries.

The use of IP prefixes within the forwarding tables of routers is illustrated in Figure 1.8. The figure depicts a router y at the right side, connected to network A via router x and connected to network B via router z. Network A consists of nodes within the subnet 200.23.16.0/20, and network B consists of nodes within subnet 199.31.0.0/16. From this perspective it makes sense for router y, to install a forwarding entry towards network A via router x, and a forwarding entry towards network B via router y, relying on their corresponding prefixes. However, network A consists of 3 smaller subnets, for which one of them is marked in gray and has subnet 200.23.18.0/23. The latter network has a shorter path from router x via network B (a shortcut), contrary to the other parts of network A. This can be taken into account by router x, by installing a more specific forwarding entry towards the gray network, indicating to use router z. Indeed, if router y now receives a packet towards an end host with address 200.23.18.12 (as illustrated on the figure), a longest prefix match will return the entry 200.23.18.0/23 leading to the path via network B. While the longest-prefix matching lookup algorithm is perfectly able to cope with the above example, it also illustrates that the availability of additional routes (e.g., due to shorter paths) can significantly increase the number of forwarding entries in routers. This is one of this aspects which contributes to the explosion of forwarding tables in backbone routers (*routing de-aggregation*).

1.3.1.2 Routing protocols

In most cases, the forwarding table of IP routers is populated by *routing protocols* exchanging topology- or path-related information over the network (or AS). Based on the received routing information, routers can calculate or derive paths towards any part of the network.

Link-state routing protocols such as Open-shortest Path First (OSPF [11]) or

[8]also referred as Forwarding Information Base (FIB)

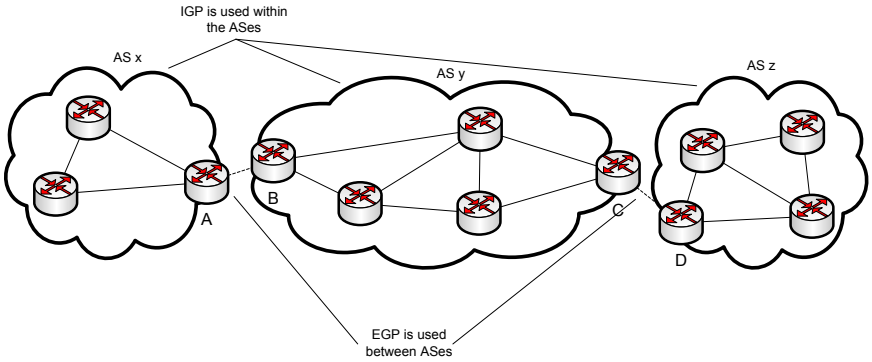


Figure 1.9: Routing within and between ASes

Intermediate System to Intermediate System (IS-IS [12]) distribute Link-State Advertisement and Update (LSA/LSU) information related to the state of links locally detected by routers. An LSA contains the router's local routing topology (e.g., detected links), an LSU is a packet that implements the flooding of one or more LSAs. Routers receiving these messages can deduce the resulting network graph, and use efficient algorithms (e.g., Dijkstra or Bellman-Ford) for calculating shortest paths within the topology and reachable nodes.

Path-vector routing protocols such as the Border Gateway Protocol (BGP, [13]) distribute pathvectors (sequences of intermediate nodes) towards reachable nodes. When path-vector routing protocols such as BGP are used, participating routers have no (direct) view on the topology of the network.

Because the Internet network is large and diverse, routing protocols are only responsible for specific network parts. Depending on their scope, the protocols are classified into two classes: the intra-AS routing level determined by *Interior Gateway protocols* (IGP) such as OSPF link-state protocol; and *Exterior Gateway Protocols* (EGP) for inter-AS routing as used in the Internet backbone network. BGP is the currently used EGP-protocol within the Internet.

Figure 1.9 illustrates the difference between IGP and EGP protocols. Whereas an IGP such as OSPF is used within the scope of an AS, BGP is used to distribute paths between the ASes. For example, BGP router D from AS z will announce the path towards its own AS z to BGP router C. BGP router C knows how to reach BGP router B relying on the local IGP protocol running in AS y. Using this connectivity BGP router B detects the path towards AS z and distributes the path (AS y, AS z) to BGP router A. Because BGP is a policy-based protocol, routers can filter received paths as desired before forwarding them towards their BGP adjacencies (neighboring BGP routers).

Although IP routing based on IGP and EGP routing protocols scales relatively well in dynamic networks, the resulting re-convergence times of these protocols

can be relatively slow, i.e. in the range between seconds and minutes. For this reason techniques such as IP-FastReroute (IPFRR) have been designed in the context of IGPs, as means to pro-actively calculate and install alternate forwarding entries. Upon local failure detection of the next hop, these can be quickly activated, ensuring quicker failure recovery. These techniques are studied into more detail in Chapter 4 and 5.

1.3.2 Multi-Protocol Label Switching

While connectionless IP routing in combination with IGPs has proven to scale relatively well (up to thousands of routers), the resulting routing is restricted to shortest paths, and the re-convergence times can be slow (seconds to minutes). This observation has lead to the design of *Multi-Protocol Label Switching* (MPLS, [14]), a packet-switched technology which allows connection-oriented networking[9]. MPLS relies on the set up of *Label Switched Paths* (LSPs), to provide connection-like connectivity in an IP-network. In this forwarding technology, additional labels are attached to IP packets. These labels are stored in the MPLS label header attached between the IP packet header and the layer 2 header (referred as the *shim header*, as indicated in Figure 1.7). MPLS-enabled routers or Label Switching Routers (LSRs) make forwarding decisions on the MPLS label, rather than on the IP destination address attached to the IP packet.

Three types of MPLS routers can be distinguished. The first and last LSR on the LSP are referred as the *ingress Label Edge Router* (LER) and the *egress LER*. The path and transformations followed by an IP packet which is sent through an LSP is illustrated in Figure 1.10. Ingress LER router A receives a packet for destination 200.23.18.12, it checks its FIB, finds a match with the prefix 200.23.18.0/23[10], and pushes the label 5 to the packet and forwards it towards LSR B. The latter solely checks the label (potentially in combination with the incoming interface[11]) in its FIB, swaps the label for 10, and forwards it to the Egress LSR, which pops the label and delivers it to the rest of the network. It is clear that this process allows any type of path in a network, as long as the intermediate forwarding tables are configured correctly.

The most important advantage of MPLS compared to connection-less IP forwarding, is its support for *traffic engineering* and *load balancing*. This refers to the possibility of MPLS to configure arbitrary paths through the IP network, rather than only relying on shortest paths. This principle is illustrated in Figure 1.13.

[9] Stricto sensu, MPLS is rather path-oriented, than connection-oriented. Some constructs such as path merging do not ensure that the input and output of the path is the same. Therefore the resulting paths are not really (isolated/guaranteed) connections.

[10] The class of all packets which are handled in the same way at a given ingress router are referred as the Forwarding Equivalence Class (FEC)

[11] interface-wide/link-local forwarding instead of platform-wide forwarding

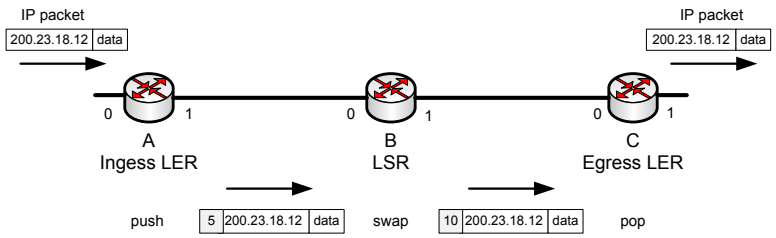


Figure 1.10: Label switching and swapping in MPLS networks

	IN interface	*IN label*	*Destination address prefix*	*OUT label*	*OUT interface*
Ingress LER	0	None	200.23.18.0/23	5	1
LSR	0	5	200.23.18.0/23	10	1
Egress LER	0	10	200.23.18.0/23	None	1

Table 1.3: MPLS forwarding tables for Figure 1.10

While shortest routing of two demands: (a,d) and (a,e) can result into a bottleneck link a-e, as indicated in the second network on the figure, path freedom as possible with MPLS can spread traffic such that the link a-e a-b and b-d are used for fulfilling these demands without forcing double bandwidth on any of the links.

The setup of LSPs requires protocols to distributed network-related information, and protocols for signaling the resulting paths. In this context the OSPF-protocol is often extended with Traffic Engineering (TE) facilities [15]. OSPF-TE not only distributes discovered adjacencies, but also load and/or label related information. This allows other OSPF-TE-enabled MPLS-routers in the network to calculate suitable paths taking into account the available resources. The resulting constraints that possibly can be taken into account lead to the term of *constraint-based routing* (CR). In case LER/LSRs do not have sufficient computational resources to calculate the routes themselves, a special-purpose, better-equipped (central) router can be introduced into the network, taking in charge the adequate calculate paths based on received information via OSPF-TE. The latter is often referred as a Path Computation Engine (PCE).

In order to effectively configure the calculated paths in an MPLS network,

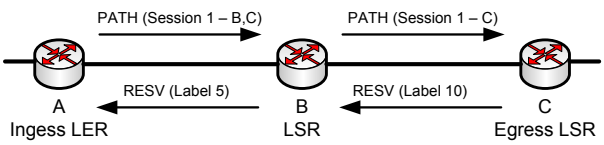


Figure 1.11: Using RSVP-TE for LSP setup

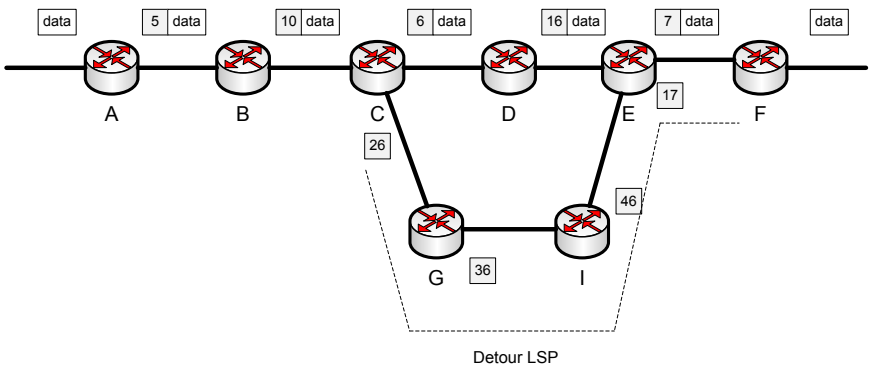


Figure 1.12: MPLS local detour

a signaling protocol is required such as the Resource reSerVation Protocol for Traffic Engineering (RSVP-TE, [16]). RSVP communicates with two basic types of messages, PATH and RESV messages. PATH messages indicate a request for an LSP setup initiated by the ingress LER (*source routing*), RESV messages indicate a reservation of a path or LSP in response of the request. Figure 1.11 illustrates the *downstream-on-demand path setup* process for the LSP of Figure 1.10. In this case the path request is initiated by the source node A which has calculated the route (A,B,C) towards C. The PATH-request is sent from A to C via B, and in reply, the reservation of according labels is distributed in the opposite direction (the label bound to a FEC is received from its downstream neighbor). This leads to the labels 10 and 5 as provided by node C and node B in the corresponding RESV messages. The path request can contain several attributes such as bandwidth requirements. RSVP-TE is a soft-state protocol, requiring the retransmission of refresh messages in order to maintain an LSP.

Using MPLS, network recovery can be provided within the range of tens of ms. LSPs can be recovered on almost any possible scope: an end-to-end base, but also using local recovery forms such as link or node recovery (see 1.2.1).

Figure 1.12 illustrates MPLS' support for local detours. The figure illustrates the label allocation for the primary LSP2 and detour LSP2 protecting LSP1 for a failure of the link c-d or LSR d. For example, when a failure of the node d occurs, as soon as the LSR C detects the failure (for which the detection time can be as fast as 10 ms, see Section 1.4), the fast-reroutable LSP1 is locally rerouted to follow the Detour LSP2. Once LSR C detects the failure of LSR D, the label is not any longer swapped from 10 to 6 and forwarded on the interface C-D, but from 10 to 26, and sent on the outgoing interface C-G. While both the Detour LSP and the primary LSP use the same link E-F, they use different labels. However, this is not strictly required because MPLS supports *LSP merging*. LSP merging allows

that two LSPs use the same label when they overlap on a link or segment. Other recovery mechanisms for MPLS networks can be found in [4].

1.3.3 Ethernet (VLAN-)switching

Connectivity between L3 IP routers or L2.5 MPLS switches needs to be provided by lower layer technologies of the TCP/IP model. The default packet-switched and connection-less layer 2 technology in many home and company networks is (VLAN-)bridged Ethernet as defined in the IEEE standard 802.1Q [17]. This simple, cost-efficient and almost plug & play technology allows the interconnection of a set of IP end-hosts through a network of switches.

Ethernet interfaces are identified by a 6 byte MAC address. The Ethernet frame header carries both a source MAC address and the destination MAC address, as indicated in Figure 1.7. The default forwarding behavior of Ethernet switches is to flood received frames over all non-incoming interfaces. To avoid forwarding loops during this flooding process, Ethernet switches limit the physical topology to a logical tree using the *Spanning Tree Protocol* (STP, RSTP or MSTP[12]). This is illustrated in the topmost network of Figure 1.13. The physical mesh topology of the switches is restricted to a logical tree topology by blocking the links b-e, b-d and a-e.

MAC learning is used to further limit the flooding. This process ensures that a switch remembers the link between a MAC address and the interface from which a frame using this MAC address as source arrived. The resulting relationship is stored into dynamic entries into the *Ethernet Filtering database*. These entries age after a pre-configured time of typically 300 s. Static entries could be entered through a management action. This process is less advanced than shortest-path routing protocols, however it requires significantly less control.

A *virtual local area network*, virtual LAN or VLAN, is a group of hosts with a common set of requirements, which communicate as if they were attached to the same broadcast domain, regardless of their physical location. A VLAN has the same attributes as a physical local area network (LAN), but it allows end stations to be grouped together even if not on the same network switch. Therefore, VLANs are a means to provide network segmentation and traffic isolation. VLANs can be attached to switch interfaces, leading to port-based VLANs, or Ethernet frames can be attached to a VLAN by attaching an additional label or tag. The resulting 12-bit tag is defined in IEEE 802.1q [17], leading to 4096 potential VLANs in an Ethernet network. Therefore, the VLAN-tag is also referred as the *Q-tag* or VID (VLAN-ID).

The principle of VLANs is illustrated in Figure 1.14. The situation depicts a simplified company LAN which interconnects offices of two company depart-

[12]defined in respectively IEEE 802.1d [18], IEEE 802.1w [19], and IEEE 802.1s [20] standards

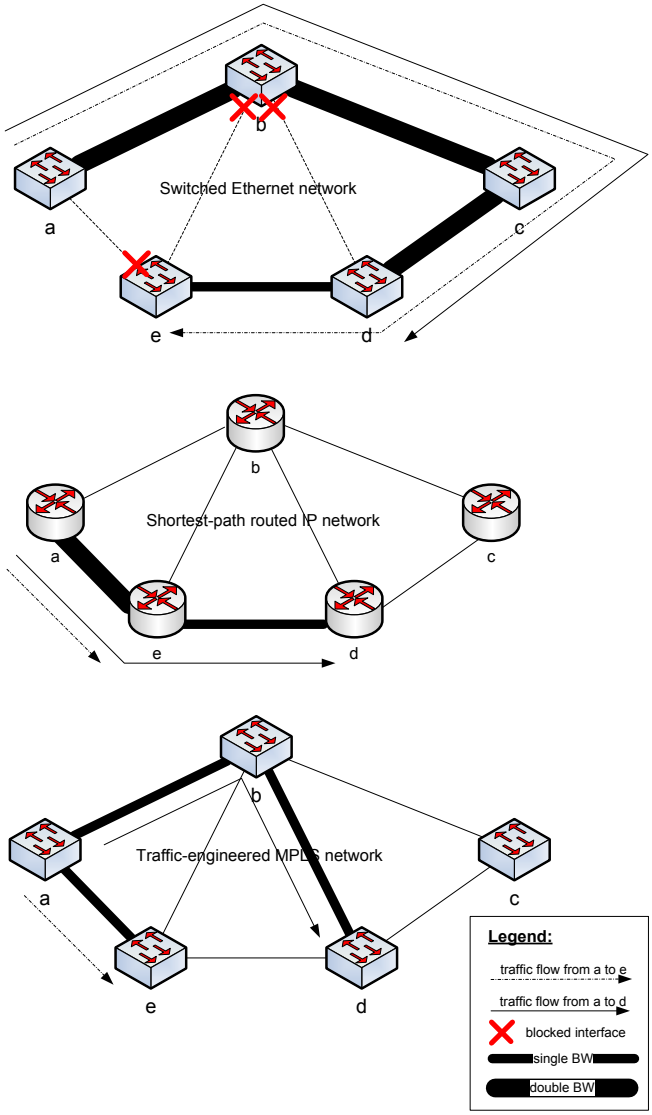


Figure 1.13: Bandwidth usage vs. technology

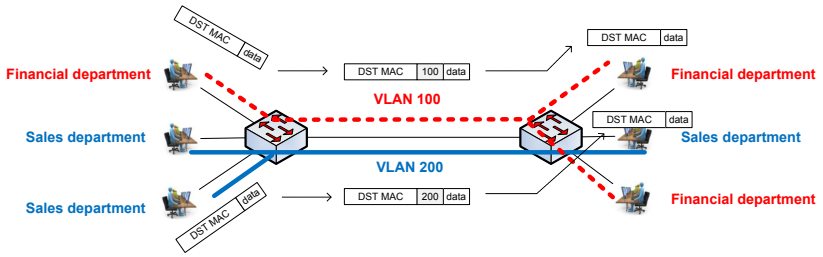


Figure 1.14: Traffic isolation using VLANs

ments: the financial department (FD) and the sales department (SD). To avoid that financial department-related frames are flooded over links and interfaces which are connecting to the sales departments, FD traffic is logically constrained by the use of VLAN 100, and SD traffic is constrained using VLAN 200. The switch interfaces connected to the end hosts are configured to tag the frames with the according Q-tags. Because of this, the link segment between the two switches (interconnected via so-called trunk ports), can distinguish and isolate the frames using Q-tags. The resulting MAC-learning can be made dependent on the VLAN, resulting in Shared-VLAN-learning (SVL), or independent, leading to Independent VLAN learning (IVL).

Connectivity recovery upon a failure in an Ethernet link or switch relies on xSTP protocol. If a failure disconnects the logical tree, connectivity is broken as long as the xSTP protocol hasn't re-converged. In the best cases, this can happen within ms, but some failures (for example root bridge failures) could require tens of seconds to recover [21]. Through the use of the Multiple Spanning Tree Protocol[13], (MSTP), different logical spanning trees can be used for different VLANs.

Bridged Ethernet has become so popular that it is commonly used as the convergence layer for packet-based network communication in transport networks. This has lead to a spectacular wealth and growth in Ethernet physical layer (PHY) specifications, from the original Fast Ethernet at 10 Mbps via 100 Mbps and Gigabit Ethernet over copper or fiber, towards the standardization of optical 100 GbE.

The definition and provisioning of Ethernet services in the context of (metro-) aggregation and core networks is being standardized by the Metro Ethernet Forum[14] (MEF). Whereas Ethernet service can be provided by lower layer technologies such as Synchronous Digital Hierarchy (SDH), Optical Transport Networks (OTN) or Wavelength Division Multiplexing (WDM) (see next section), Ethernet services can also be provided on top of connection-oriented MPLS networks. Point-to-point services are typically referred as Virtual Private Wire Service (VPWS), while multipoint services are referred as Virtual Private LAN ser-

[13] defined in IEEE standard 802.1s
[14] http://metroethernetforum.org/

vices (VPLS). These are supported on top of MPLS relying on pseudo-wires and layer-2 Virtual Private Networks (VPN), as defined in the IETF Layer 2 VPN (L2VPN) Working Group and the Pseudowire Emulation Edge-to-Edge (PWE3) Working Group.

Bandwidth usage with Ethernet switching, IP routing or IP/MPLS routing To illustrate the (in-)flexibility rising from different control planes in the introduced technologies, Figure 1.13 compares the potential difference in bandwidth usage for a simple 5-node network between: i) switched Ethernet network (L2), ii) a shortest-path routed IP network, and iii) a traffic-engineered MPLS network. The figure depicts the used bandwidth for two traffic flows requiring the same bandwidth: traffic from node a to e (dotted line), and traffic from a to d (continuous line). Clearly the Ethernet scenario is the least efficient in terms of bandwidth usage. This may be explained from the limitations from the Spanning Tree Protocol. In the considered scenario, switch c is elected as root bridge, and accordingly, the links a-e, b-e and b-d are disabled, as indicated by the red marks, and dotted link lines. As a consequence, both flows follow the same path up to switch d, resulting in double capacity needed along the used links. In the shortest-path routed network (e.g., relying on OSPF), both flow follow the same path up to node e, resulting into double capacity needed between a and e. The third scenario illustrates the potential effect of load balancing or traffic engineering in an MPLS-switched network. By setting up MPLS connections (LSPs), none of the links requires double capacity.

1.3.4 Optical network technology

Connectivity between layer 2 nodes, such as Ethernet switches, is provided via lower layer technologies. In larger networks this relies on optical network technology. Circuit-switched (and thus connection-oriented) L1 technologies such as Synchronous Optical Networking (SONET) / Synchronous Digital Hierarchy (SDH) or Optical Transport Networks (OTN) emerged from old phone network technology. These technologies are often used when larger bandwidth and throughput guarantees are required to interconnect nodes in aggregation, core or campus networks.

SONET/SDH networks rely on Time-Division Multiplexing (TDM) over optical fiber for data transmission. Different data streams can be multiplexed, by receiving different time slots. These technologies are restricted to ring topologies of up to tens of nodes, allowing network recovery in the order of tens of ms using ring protection or restoration [4]. Packet-based connectivity can be transported over SONET/SDH equipment, by relying on framing technologies for mapping packets over fiber-optical networks. Typical technologies are: Packet Over SONET/SDH techniques (POS), or Ethernet Over SONET/SDH (EoS). SONET/SDH optical

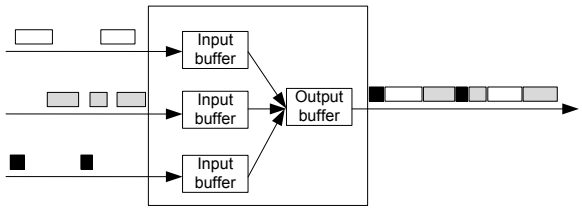


Figure 1.15: Statistical multiplexing as possible in packet-switching

technology can run on top of Wavelength Division Multiplexing (WDM) which allows to multiplex different wavelengths over the same fiber.

Optical Transport Networks (OTN) are mostly used in the backbone of the Internet, and are able to form networks of optical switches from tens to hundreds of nodes interconnected in arbitrary topologies. Connections following any desired path can be set up by management or relying on an IP-based control plane such as Generalized Multi-Protocol Switching (GMPLS) [22]. Every optical switch (cross connect), is in this case augmented with an IP controller which is able to interpret previously referred protocols such as OSPF-TE and RSVP-TE. In the context of GMPLS, these protocols were extended for supporting optical circuit-switching. The label concept is generalized such that it not only applies to packet header-based labels such in MPLS, but also to wavelengths (lambda's). OTN networks also allow a broad variety of protection and restoration mechanisms at link, node, segment- and path level allowing recovery within the order of tens of ms [4].

The advantage of these optical circuit technologies is that they are able to provide large bandwidths with high service level guarantees. However, many of these optical technologies have low support for fine and tunable bandwidth granularities. For example, the data-rate progression of SDH circuits follows the following sequence: $(51, 155, 622, 1244, 2488, 9953, 39813) \times 1$ Mbps. This step-wise function does not allow gradual increase of bandwidth usage, leading to over-provisioning or bandwidth limitation. Circuit-switched technologies such SDH or OTN, pre-allocate and reserve circuits regardless of the effective demand. If several circuits arrive at a common network node to use the same outgoing link (see Figure 1.15), no benefit can rise from sharing the common path. In order to guarantee that both circuits can follow the same path, circuit-switching requires that the sum of the incoming link capacities is reserved on the outgoing link. On the contrary, packet-switched technologies can share the same link by only using bandwidth if packets are effectively received (see Figure 1.15). This type of on-demand (instead of pre-allocate) sharing of resources is called *statistical multiplexing* of resources.

1.4 OAM and fault detection

Fault detection can be considered as subset of the broader set of Operations, Administration and Management (OAM) functionality. OAM tools allow the network operator to pro-actively manage networks and Service Level Agreements (SLAs). The resulting functionality includes fault detection, fault verification, and fault isolation. This provides proactive detection of service degradation, service performance monitoring, and SLA verification. The OAM toolbox usually consists of a set of commands which can be executed on demand by the network manager, such as ping functionality to check connectivity with specified points in the network, or traceroute tools to check the path/route taken between some points in the network. These and other OAM tools are mainly considered for troubleshooting purposes. Fault detection mechanisms are not only used for troubleshooting, but also for triggering network recovery schemes.

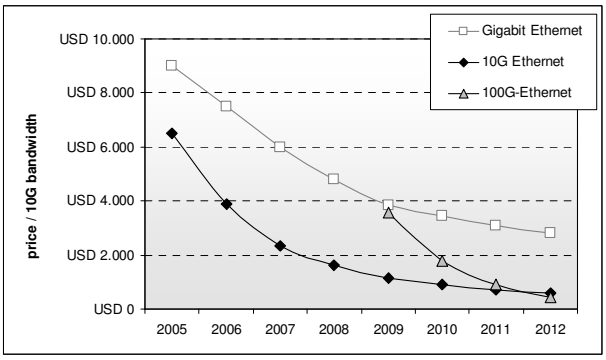
1.4.1 Hardware and circuit-based failure detection

Adequate fault detection is an essential component of any network technology in supporting fast recovery of connectivity. Optical technologies such as SONET/SDH allow line card hardware to detect failures in 20 ms, relying on Loss of Signal (LOS) detection, Loss Of Frame (LOF) or Alarm Indication Signal (AIS) [4, 23]. Similar detection times are possible using hardware failure detection in back-to-back Gigabit Ethernet [23]. The detected failure can now be recovered in these optical layers, or raised to higher layers such as Ethernet bridging, IP, or IP/MPLS, in order to allow these technologies to execute the necessary recover actions (previous sections describe the possible mechanisms).

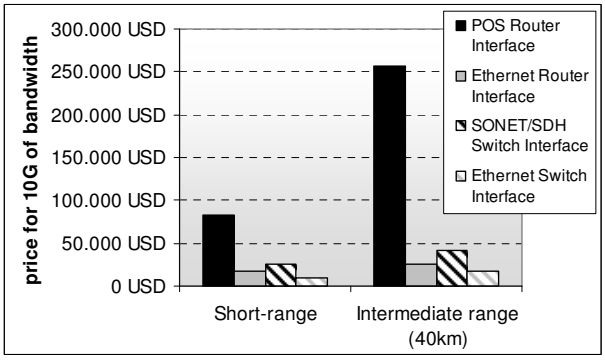
1.4.2 Failure detection using Hello-protocols

When failures cannot be detected in the physical layer, failure detection is possible through the use of Hello-based protocols in packet-switched technologies. IGPs such as OSPF have built-in timers to send Hello messages (refreshes) to adjacent routers at regular time intervals. However, because of the resulting load on the router CPU, the resulting time intervals are typically at least every second. The default value in, e.g., OSPF is 10 seconds.

Bi-Directional Forwarding Detection (BFD [24]) is a light-weight Hello-based failure detection protocol which is independent of the used routing protocol or forwarding plane. Highly frequent Hello messages can be sent through the forwarding path to achieve detection times in the order of 15 ms. A downside of this protocol is the extra traffic that is generated resulting from the regularly transmitted Hello's. BFD can be used on connectionless IP in single and multi-hop settings and on IP/MPLS for verifying LSP-based connectivity.



(a) Extrapolated price evolution of Ethernet bandwidth for different Ethernet PHY standards



(b) Price comparison of router and switch interfaces

Figure 1.16: Cost of Ethernet equipment, from [25]

1.5 Cost observations

Ethernet technology is often considered as a cost-efficient technology. However, it is difficult to prove this on a systematic basis. The authors of [25] quantified this claim by performing a comparative cost study of different technologies in the context of core networks. The study takes into account the Capital and Operational Expenditures (CAPEX and OPEX) of the different technologies applied on a German backbone reference topology over three years, taking into account growing network traffic and equipment price evolutions.

While we won't go into detail on the different technologies and neither the details of the cost study, the outcome of the CAPEX and OPEX analysis demonstrates a considerable cost advantage of 100G-Ethernet in comparison to SDH-based solutions. The superior CAPEX performance is explained from the huge cost advantage of Ethernet devices and their fast price decline over time. The

reduced switch and line card count in 100G-Ethernet networks (because higher bandwidth is available per port, less ports and less switches are needed, resulting into a lower failure probability) and the efficient economics of Ethernet services are the arguments for a superior OPEX performance.

The fast price decline on which the study relies, is based on an extrapolation of equipment prices of 7 equipment vendors. The resulting curves in Figure 1.16a depict the used price evolution 10 Gbps bandwidth for three Ethernet PHY standards: Gigabit Ethernet, 10 Gbps Ethernet and 100 Gbps Ethernet. The resulting price for 10 Gbps of bandwidth is compared with other switch technologies in Figure 1.16. One may observe on the figure that the prices of Ethernet interfaces as used in routers and switches are considerably less compared to their counterparts in POS routers or SDH-based switches.

1.6 Problem statement and research questions

Traditionally, circuit-switched technologies have been the preferred transport technology in aggregation and core networks. Contrary to the past, where telephone service was circuit-based, almost all current network traffic originates from an IP-enabled end-host and is packet-based. While the latter gives opportunities for statistical multiplexing, most of the current transport infrastructure still relies on these circuit-based technologies. For this reason, optical circuit-switched technology is preferably only used for highly aggregated network traffic, such that the loss due to static multiplexing is minimal. Last years, operators of aggregation networks and parts of the core network are increasingly considering using packet-switched technologies directly for this purpose. However, there are several technologies in this space with different cost and complexity, and the separation of concerns of L2 and L3 technologies is becoming less explitit. Ethernet technology has low cost, is easy to deploy and supports high PHY speeds. On the other hand, the use of traditional VLANs and RSTP is not very scalable, and the support for OAM and failure detection is not available, as it originally was not designed for this purpose. Connectionless IP routing is very scalable, but lacks control on the routing. For this reason, relatively expensive IP/MPLS routing is often the packet-switched technology of choice in the core. However, the idea of slightly changing the Ethernet control plane, could potentially be a better cost-efficient solution, as suggested in Section 1.5. The evaluation of the resulting Carrier Ethernet technologies is one of the points which will be further investigated in this dissertation.

No matter how fast or how smart the recovery process could be made, first a network failure must be detected. The most deployed network situation consists of IP routers which are interconnected by bridged Ethernet segments, or multi-access broadcast capable networks. For this reason, this dissertation also investigates the value of failure detection techniques such as BFD in exploiting the multipoint

features of an Ethernet network.

A third observation is that still many routers within the core of the Internet infrastructure run IGP routing protocols. Despite the superior support for traffic engineering by MPLS routers, connectionless IP routing driven by IGPs are still widely used. IGP-driven IP networks scale well and require low management effort in case the network traffic dynamically changes[15]. However, when a network failure occurs, these have to recalculate their routing tables, and subsequently update tens of thousands of routing entries. Existing research has focused mostly on reducing the global recovery time of all affected network traffic. Nevertheless, there is an aspect related to the quality of this update process itself: if many traffic flows are affected, existing techniques do not take into account the cost from updating a specific traffic flow sooner or later than another. This observation raises another research question: could network routing and recovery be made adaptive to the observed network traffic, and could the resulting ”quality” of the recovery process be improved from this perspective?

At last, to improve the global recovery time of IGP protocols, IPFRR techniques have been proposed. However these again introduce additional management and configuration effort, while still not assuring the recoverability of all potential link failures. Could these processes be improved such that they result into a lower network management effort and still provide decent network recovery? Ideally the resulting control loop could be closed, such that no operator any longer would be needed to configure alternate entries.

1.7 Outline and contributions

Fast network recovery and efficient routing in switched and routed network architecture are the central research topics of this dissertation. These topics are studied from two network levels: connection-oriented Ethernet on one level, and connection-less IP routing at the other. The first direction introduces a cost-efficient and flexible alternative for IP/MPLS, the second allows for an increased independence of the network with respect to the network operator, while still providing sufficient adaptivity and carrier-grade performance.

In Chapter 2, the potential of Carrier Ethernet alternatives is analyzed, and the design of Ethernet Label Switching (ELS) is proposed. The resulting design allows label switching at the Ethernet layer. The proposed technology is implemented in an emulation platform based on the Click Modular Router platform for executing the forwarding functionality, and the Dragon GMPLS suite for control. The latter control protocol implementation was extended for enabling control of the

[15]Comparison between MPLS-driven and IGP-driven IP networks, and combinations is made in [26, 27].

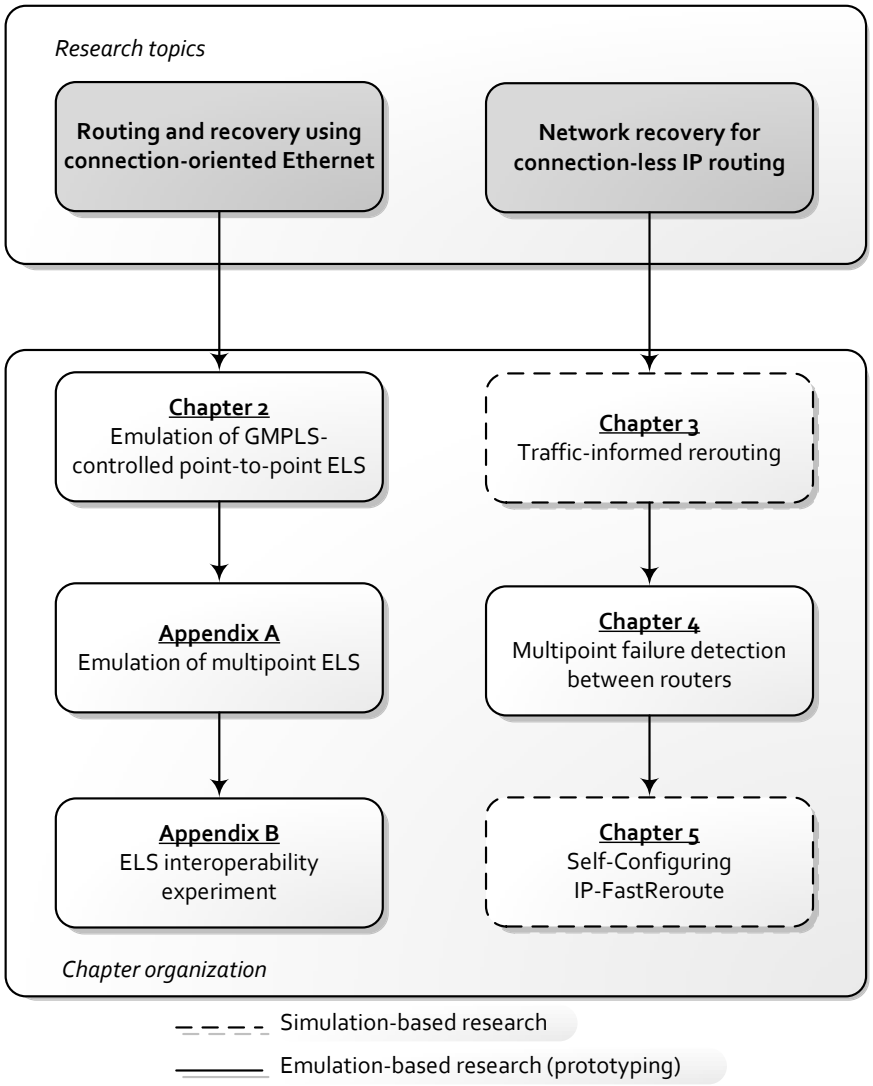


Figure 1.17: Topics and structure of the dissertation

resulting ELS switches. This allowed benchmarking a realistic prototype in different settings. The resulting evaluation validates that carrier-grade recovery, high throughput and traffic engineering are possible for point to point label switched paths (LSPs). In Appendix A, the platform is further extended for supporting point-to-multipoint connections, as well as recovery of these multipoint connections. The implemented functionality has been demonstrated on various European events[16], and the related TIGER-project has been awarded with the European CELTIC Excellence award in 2009. Appendix B reports on an interoperability experiment between two Carrier Ethernet implementations: the ELS-implementation developed in this thesis, and label-swapped Ethernet as implemented at Keio University in Japan. The paper [28] resulting of the cooperation received the IEICE Communications Society Excellent Paper Award.

Chapter 3 investigates the potential of IGP-based IP routers to involve network traffic information to improve the recovery process during the event of updating forwarding entries upon a network failure. The routing table update process is modeled such that the resulting packet loss as a result of the process can be characterized. A heuristic is proposed in [29], and the resulting performance is measured in simulation, modeling the network traffic according to a bounded Pareto distribution. In [30] the model and heuristic are further extended, and evaluated based on real network traces. The results illustrate that real network traffic fluctuates in very short time periods. For this reason, network traffic models are taken into account to evaluate the potential gain of network traffic prediction. The resulting paper received the Best Paper Award at the 2th RNDM workshop in 2010 in Moscow.

Chapter 4, a multipoint failure detection mechanism is proposed based on BFD for use between IP routers interconnected via Ethernet-based multipoint networks. The resulting scheme is prototyped in an emulation environment using the Click Modular Router framework. In combination with IPFRR, the proposed scheme is compared with OSPF-based recovery. The resulting tests illustrate that very fast recovery can be obtained. In addition, time performance and resource consumption of the implementation are evaluated.

At last, the research reported in Chapter 5 intends to reduce the required management effort, as well as the potential recoverability of IGP-driven IP routers relying on IPFRR techniques. In addition, a self-configuring process is proposed, avoiding the need for configuration by the network operator. This solution enables fast recovery in connection-less IP networks for a wide range of potential link failures for little management effort. The related paper [31] received the Best Paper Award at the 3rd RNDM workshop in 2011, in Budapest.

[16] European Celtic Event in Helsinki on 27-28 February 2008, European Celtic event on 11-12 March 2009 in Paris, and the final audit of the IWT TIGER project in Ghent

1.8 Publications

The research results obtained during this PhD research have been published in scientific journals and presented at a series of international conferences. The following list provides an overview of the publications during my PhD research.

1.8.1 Publications in international journals (listed in the Science Citation Index [17])

1. S. Shimizu, **W. Tavernier**, K. Kikuta, M. Nishida, D. Ishii, S. Okamoto, D. Colle, M. Pickavet, P. Demeester, and N. Yamanaka. *Interoperability experiment of VLAN tag swapped Ethernet and transmitting high definition video through the layer-2 LSP between Japan and Belgium*. IEICE transactions on communications, 93(3):736, 2010. IEICE Communications Society Excellent Paper Award

2. **W. Tavernier**, D. Papadimitriou, D. Colle, M. Pickavet, and P. Demeester. *Packet loss reduction during rerouting using network traffic analysis*. Telecommunication Systems, pages 1 - 19, 2011

3. **W. Tavernier**, D. Papadimitriou, D. Colle, M. Pickavet, and P. Demeester. *Packet Loss Reduction During Rerouting*. Communications Letters, IEEE, (99):1 - 3, 2011

4. **W. Tavernier**, D. Papadimitriou, D. Colle, M. Pickavet, and P. Demeester. *Self-configuring Loop-Free Alternates with high link-failure coverage*. Telecommunication Systems, 2012. Accepted

5. S. Sahhaf, **W. Tavernier**, D. Papadimitriou, D. Colle, M. Pickavet, and P. Demeester. *Link Failure Recovery Technique for Greedy Routing in the Hyperbolic Plane*. Computer Communications, 2011. Accepted

1.8.2 Publications in international conferences (listed in the Science Citation Index [18])

6. **W. Tavernier**, D. Colle, M. Pickavet, and P. Demeester. *GMPLS-controlled Ethernet segment protection*. In Optical Communication, 2008. ECOC

[17] The publications listed are recognized as 'A1 publications', according to the following definition used by Ghent University: A1 publications are articles listed in the Science Citation Index, the Social Science Citation Index or the Arts and Humanities Citation Index of the ISI Web of Science, restricted to contributions listed as article, review, letter, note or proceedings paper.

[18] The publications listed are recognized as 'P1 publications', according to the following definition used by Ghent University: P1 publications are proceedings listed in the Conference Proceedings Citation Index - Science or Conference Proceedings Citation Index - Social Science and Humanities of the ISI Web of Science, restricted to contributions listed as article, review, letter, note or proceedings paper, except for publications that are classified as A1.

2008. 34th European Conference on, pages 1 - 2. IEEE, 2008

7. **W. Tavernier**, D. Papadimitriou, D. Colle, M. Pickavet, and P. Demeester. *Emulation of GMPLS-controlled Ethernet label switching*. In Testbeds and Research Infrastructures for the Development of Networks & Communities and Workshops, 2009. TridentCom 2009. 5th International Conference on, pages 1 - 9. IEEE, 2009

8. **W. Tavernier**, D. Papadimitriou, D. Colle, M. Pickavet, and P. Demeester. *Optimizing the IP router update process with traffic-driven updates*. In Design of Reliable Communication Networks, 2009. DRCN 2009. 7th International Workshop on, pages 115 - 122. IEEE, 2009

9. K. Casier, **W. Tavernier**, D. Colle, M. Pickavet, D. Papadimitriou, and P. Demeester. *Forecasting Cost Trends for Carrier Ethernet*. In GLOBECOM Workshops, 2009 IEEE, pages 1 - 6. IEEE, 2009

10. **W. Tavernier**, D. Papadimitriou, B. Puype, D. Colle, M. Pickavet, and P. Demeester. *Fast failure detection in multipoint networks*. IP Operations and Management, pages 51 - 64, 2009

1.8.3 Publications in other international conferences

11. L. Caviglia and et al. *TIGER: Optimizing IP and Ethernet adaptation for the Metro Ethernet Market*. In Proceedings of 12th European Conference on Networks and Optical Communications (NOC), 2007

12. G. Das, D. Papadimitriou, **W. Tavernier**, D. Colle, T. Dhaene, M. Pickavet, and P. Demeester. *Link State Protocol data mining for shared risk link group detection*. In Computer Communications and Networks (ICCCN), 2010 Proceedings of 19th International Conference on, pages 1 - 8. IEEE, 2010

13. B. Puype, G. Verbanck, J. Michielsen, M. Moeskops, **W. Tavernier**, D. Colle, M. Pickavet, and P. Demeester. *Impact of topology on layer 2 switched QoS sensitive services*. In Transparent Optical Networks (ICTON), 2010 12th International Conference on, pages 1 - 4. IEEE, 2010

14. **W. Tavernier**, D. Papadimitriou, D. Colle, M. Pickavet, and P. Demeester. *Using AR(I)MA-GARCH models for improving the IP routing table update*. In Ultra Modern Telecommunications and Control Systems and Workshops (ICUMT), 2010 International Congress on, pages 628 - 634. IEEE, 2010. Best Paper Award

15. **W. Tavernier**, D. Papadimitriou, D. Colle, M. Pickavet, and P. Demeester. *Automated learning of loop-free alternate paths for fast re-routing*. In Ultra Modern Telecommunications and Control Systems and Workshops (ICUMT), 2011 3rd International Congress on, pages 1 - 7. IEEE, 2011. Best Paper Award.

16. **W. Tavernier**, D. Papadimitriou, D. Colle, M. Pickavet, and P. Demeester. *Point-to-multipoint connectivity and recovery in Ethernet Label Switching*. In L. Fàbrega and R. Fabregat, editors, IX workshop in G/MPLS networks, pages 125 - 136. Documenta Universitaria, 2011

1.8.4 Publications in national conferences

17. **W. Tavernier**, D. Papadimitriou, D. Colle, M. Pickavet, and P. Demeester. *GMPLS-controlled Ethernet segment protection*. In Proceedings of 2nd Gent University and KEIO University Global COE Joint workshop, pages 3 - 7,2008

18. **W. Tavernier**, D. Papadimitriou, D. Colle, M. Pickavet, and P. Demeester. *Emulation of GMPLS-controlled Ethernet label switching*. In 3rd Gent University and KEIO University Global COE Workshop Proceedings, 2009

19. **W. Tavernier**, D. Colle, M. Pickavet, and P. Demeester. *Carrier Grade Ethernet*. In 9th FTW PhD Symposium, 12 2008

20. **W. Tavernier**, D., D. Colle, and D. T. *A traffic-driven IP router update process using machine learning techniques*. In Bell Labs Open days 2009,10 2009

21. **W. Tavernier**. *A traffic-driven IP router update process using machine learning techniques*. In FEA PhD symposium, 12th. Ghent University. Faculty of Engineering and Architecture, 2011.

References

[1] D. Maltz, G. Xie, J. Zhan, H. Zhang, G. Hjálmtỳsson, and A. Greenberg. *Routing design in operational networks: A look from the inside*. In ACM SIGCOMM Computer Communication Review, volume 34, pages 27–40. ACM, 2004.

[2] A. Shaikh, C. Isett, A. Greenberg, M. Roughan, and J. Gottlieb. *A case study of OSPF behavior in a large enterprise network*. In Proceedings of the 2nd ACM SIGCOMM Workshop on Internet measurment, pages 217–230. ACM, 2002.

[3] M. Al-Fares, A. Loukissas, and A. Vahdat. *A scalable, commodity data center network architecture*. In ACM SIGCOMM Computer Communication Review, volume 38, pages 63–74. ACM, 2008.

[4] J. Vasseur, M. Pickavet, and P. Demeester. *Network recovery: Protection and Restoration of Optical, SONET-SDH, IP, and MPLS*. Morgan Kaufmann Publishers, 2004.

[5] G. Scheer and D. Dolezilek. *Comparing the reliability of Ethernet network topologies in substation control and monitoring networks*. In Western Power Delivery Automation Conference, Spokane, Washington, 2000.

[6] A. Markopoulou, G. Iannaccone, S. Bhattacharyya, C. Chuah, Y. Ganjali, and C. Diot. *Characterization of failures in an operational IP backbone network*. IEEE/ACM Transactions on Networking (TON), 16(4):749–762, 2008.

[7] M. Huynh, S. Goose, and P. Mohapatra. *Resilience technologies in Ethernet*. Computer Networks, 54(1):57–78, 2010.

[8] J. Postel. *Internet Protocol*. RFC 791 (Standard), September 1981. Updated by RFC 1349. Available from: http://www.ietf.org/rfc/rfc791.txt.

[9] S. Deering and R. Hinden. *Internet Protocol, Version 6 (IPv6) Specification*. RFC 2460 (Draft Standard), December 1998. Updated by RFCs 5095, 5722, 5871, 6437, 6564. Available from: http://www.ietf.org/rfc/rfc2460.txt.

[10] V. Fuller and T. Li. *Classless Inter-domain Routing (CIDR): The Internet Address Assignment and Aggregation Plan*. RFC 4632 (Best Current Practice), August 2006. Available from: http://www.ietf.org/rfc/rfc4632.txt.

[11] J. Moy. *OSPF Version 2*. RFC 2328, Internet Engineering Task Force, April 1998.

[12] D. Oran. *OSI IS-IS Intra-domain Routing Protocol*. RFC 1142 (Informational), February 1990. Available from: http://www.ietf.org/rfc/rfc1142.txt.

[13] Y. Rekhter, T. Li, and S. Hares. *A Border Gateway Protocol 4 (BGP-4)*. RFC 4271 (Draft Standard), January 2006. Available from: http://www.ietf.org/rfc/rfc4271.txt.

[14] E. C. Rosen, A. Viswanathan, and R. Callon. *Multiprotocol Label Switching Architecture*. RFC 3031, IETF, January 2001. Status: STANDARDS TRACK.

[15] D. Katz, K. Kompella, and D. Yeung. *Traffic engineering (TE) extensions to OSPF version 2*. Technical report, RFC 3630, September, 2003.

[16] D. Awduche, L. Berger, D. Gan, T. Li, V. Srinivasan, and G. Swallow. *RSVP-TE: Extensions to RSVP for LSP Tunnels*. RFC 3209 (Proposed Standard), December 2001. Updated by RFCs 3936, 4420, 4874, 5151, 5420, 5711. Available from: http://www.ietf.org/rfc/rfc3209.txt.

[17] *IEEE 802.1Q - Virtual LANs*. IEEE. Available from: http://www.ieee802.org/1/pages/802.1Q.html.

[18] *IEEE 802.1D - MAC bridges*. IEEE. Available from: http://www.ieee802.org/1/pages/802.1D.html.

[19] *IEEE 802.1w - Rapid Reconfiguration of Spanning Tree*. IEEE. Available from: http://www.ieee802.org/1/pages/802.1w.html.

[20] *IEEE 802.1s - Multiple Spanning Trees*. IEEE. Available from: http://www.ieee802.org/1/pages/802.1s.html.

[21] K. Elmeleegy, A. Cox, and T. Ng. *On count-to-infinity induced forwarding loops in ethernet networks*. In Proc. IEEE Infocom, pages 1–13, 2006.

[22] E. Mannie. *Generalized Multi-Protocol Label Switching (GMPLS) Architecture*. RFC 3945 (Proposed Standard), October 2004. Updated by RFC 6002. Available from: http://www.ietf.org/rfc/rfc3945.txt.

[23] P. Francois, C. Filsfils, J. Evans, and O. Bonaventure. *Achieving sub-second IGP convergence in large IP networks*. ACM SIGCOMM Computer Communication Review, 35(3):35–44, 2005.

[24] D. Katz and D. Ward. *Bidirectional Forwarding Detection (BFD)*. RFC 5880 (Proposed Standard), June 2010. Available from: http://www.ietf.org/rfc/rfc5880.txt.

[25] A. Schmid-Egger and A. Kirstadter. *Ethernet in core networks: A technical and economical analysis*. In High Performance Switching and Routing, 2006 Workshop on, pages 6–pp. IEEE, 2006.

[26] A. Bagula. *Hybrid routing in next generation IP networks*. Computer communications, 29(7):879–892, 2006.

[27] S. Uhlig and O. Bonaventure. *On the Cost of Using MPLS for Interdomain Trafic*. In Quality of Future Internet Services, pages 141–152. Springer, 2000.

[28] S. Shimizu, W. Tavernier, K. Kikuta, M. Nishida, D. Ishii, S. Okamoto, D. Colle, M. Pickavet, P. Demeester, and N. Yamanaka. *Interoperability experiment of VLAN tag swapped ethernet and transmitting high definition video through the layer-2 LSP between Japan and Belgium*. IEICE transactions on communications, 93(3):736, 2010.

[29] W. Tavernier, D. Papadimitriou, D. Colle, M. Pickavet, and P. Demeester. *Optimizing the IP router update process with traffic-driven updates*. In Design of Reliable Communication Networks, 2009. DRCN 2009. 7th International Workshop on, pages 115–122. IEEE, 2009.

[30] W. Tavernier, D. Papadimitriou, D. Colle, M. Pickavet, and P. Demeester. *Using AR(I)MA-GARCH models for improving the IP routing table update*. In Ultra Modern Telecommunications and Control Systems and Workshops (ICUMT), 2010 International Congress on, pages 628–634. IEEE, 2010. Best Paper Award.

[31] W. Tavernier, D. Papadimitriou, D. Colle, M. Pickavet, and P. Demeester. *Self-configuring Loop-Free Alternates with high link-failure coverage*. Telecommunication Systems, 2012. accepted.

2

Emulation of GMPLS-controlled Ethernet Label Switching

This chapter proposes Ethernet Label Switching (ELS) as an extension on traditional Ethernet bridging. The resulting connection-oriented technology enables traffic engineering and fast recovery, similar as possible in MPLS, but at lower cost and complexity. A software prototype of the proposed technology is developed, and its performance is evaluated for point-to-point paths on a testbed on the iLab.t virtual wall[1] *at IBBT. In Appendix A the approach is extended and evaluated for multipoint paths.*

⋆⋆⋆

W. Tavernier, D. Papadimitriou, D. Colle, M. Pickavet, and P. Demeester.

Presented at the 5th International Conference on Testbeds and Research Infrastructures for the Development of Networks & Communities and Workshops, 2009. TridentCom 2009.

[1]The IBBT virtual wall facility is a generic test environment for advanced network, distributed software and service evaluation, and supports scalability research. The virtual wall facilities consist of 100 nodes (dual processor, dual core servers, 6x1 Gb/s interfaces per node) interconnected via a non-blocking 1.5 Tb/s VLAN Ethernet switch, and a display wall (20 monitors) for experiment visualization. The facility allows to interconnect the nodes according to any logical topology.

Abstract Last decades bridged Ethernet has been the standard layer 2 technology for LAN environments. This is mainly because it is easy to deploy, it has low cost and is relatively robust. These attractive properties, together with highly increased PHY speeds, result in several initiatives of research projects and standardization bodies to extend Ethernet for using the technology in carrier environments. However, these Carrier Ethernet solutions are difficult to test with given simulation tools. This paper presents an emulation platform based on open-source software that allows to evaluate a Carrier Ethernet-variant called Ethernet Label Switching (ELS). The platform allows various topology setups and has the possibility to perform time and performance benchmarking and traffic generation using standard Linux software. The architecture of the emulation framework is presented, and promising results are obtained in several setups.

2.1 Introduction

For decades Ethernet is dominating the LAN environment. Ethernet bridging has become a synonym for cheap, plug-and-play and highly compatible network technology. The high Ethernet usage in companies together with economical globalization trends result in an ever-increasing demand of Ethernet service from these companies to service providers. A market research survey of Infonetics over 29 service providers shows that carriers report that there is an increase of 90 to 100 percent of Ethernet traffic (see [1]). These Ethernet services enable the seamless interconnection of the Ethernet networks of geographically spread parts of a company as if they were directly connected within the same Ethernet LAN (see VPWS and VPLS services in next sections).

The shift towards more packet-oriented services has also its consequences on the transport technology that is being envisioned by operators. Why would they still use more expensive circuit-based optical equipment, involving several conversion layers, e.g., Generic Framing Procedure (GFP), Virtual Concatenation (VCAT), etc., if the majority of services is becoming more and more packet-based (elastic and streaming traffic). This reasoning will become even more dominant, given the highly increasing Ethernet PHY speeds, going from 10 Gbps towards 40 and 100 Gbps. Therefore using Ethernet directly as a transport technology in access-, (metro-)aggregation or even core networks, becomes more and more attractive.

On the other side, control mechanisms within Ethernet technology were not designed for being used in transport networks. The base principles of Ethernet bridging of learning and flooding (see Section 2.2) within a restricted virtual tree topology are far too restrictive for controlling transport networks. This resulted in a spectrum of new Ethernet technology designs that were outlined both by IEEE, ITU-T and IETF. Recent efforts of these standardization organizations intend to

use GMPLS for controlling Ethernet networks. However, these proposals rely on a different forwarding model, as will become clear in the next sections. The following section will shortly discuss the shortcomings of existing bridged Ethernet and will give an overview of resulting efforts to overcome these. In the rest of the paper, the focus will be on Ethernet Label Switching (ELS).

Whereas there is a plethora of environments to simulate and emulate Ethernet bridging, this is not the case for these new Carrier Ethernet technologies which involve several changes to the Ethernet control and forwarding plane. Nevertheless, benchmarking the performance and scalability of these technologies is an important next step to make these technologies realistic alternatives for existing optical transport technology. Section 2.3 will give a short overview of available simulation techniques, their respective shortcomings and will given an overview on existing evaluation environments. In Section 2.4, an emulation environment based on extended Click Modular Router and Dragon open-source software will be presented to enable benchmarking of ELS. The last part of the paper will give time and performance benchmarking results of ELS that were obtained in the constructed emulation platform. Finally, the paper concludes with future work and some summarizing remarks.

2.2 Evolving Ethernet

2.2.1 Bridged (VLAN) Ethernet

Ethernet bridges and switches do not require configuration of their forwarding tables or any control protocol to enable network operation. This plug-and-play character mainly results from the fact that bridges use MAC address learning, flooding, and learned MAC forwarding. This means that initially MAC bridges have empty forwarding tables and broadcast any incoming frames to all non-incoming ports (split-horizon flooding).

Meanwhile a forwarding entry is created by watching the relationship of the incoming interface and the source MAC address of the incoming frame in each node (learning). Once this relationship is stored in the forwarding table, a new incoming frame destined to a learned MAC address will only be forwarded to the learned interface. This forwarding mechanism works on a local, independent per-node basis, meaning that no state is stored in nodes concerning end-to-end connectivity (connectionless forwarding). Because Ethernet does not contain a TTL field, flooding frames (for those frames for which no forwarding entry can be found in the forwarding table) can result into frames being endlessly looped through the bridged network, and so congesting the network. Therefore a control protocol, called the (Rapid) Spanning Tree Protocol (RSTP, [2]) is used so as to restrict the physical topology into a logical tree topology such that it doesn't contain cycles.

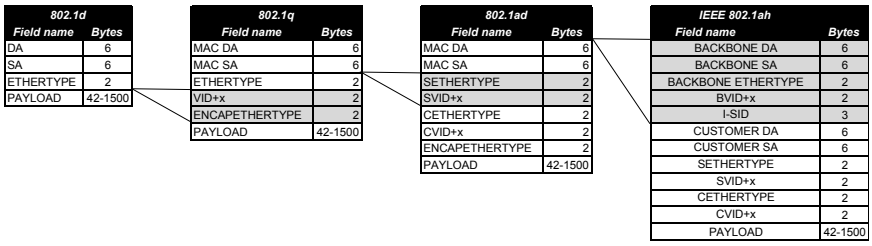


Figure 2.1: Hierarchy of Ethernet frame headers in the IEEE standards

To enable logical separation within a LAN network, the concept of the Virtual LAN was created. VLANs allow segmentation of a broadcast domain (associated to the physical topology) to multiple logical broadcast domains (each associated to a given VLAN) such as to restrict and separate the flooding domain and learning scope of MAC frames. The VLAN to which a frame belongs is identified by an additional 12-bit C-tag in the Ethernet frame header (see Figure 2.1[2]). More details can be found in the related IEEE standard 802.1Q.

2.2.2 Carrier Ethernet

Whereas the protocol as sketched in the previous subsection works perfectly well for local environments (such as campus and enterprise networks), it lacks features which are desired in provider networks:

- Isolation of several traffic streams is not possible because all traffic is handled in the same way
- Scalability of bridged Ethernet networks is limited because forwarding entries cannot be grouped, every MAC address needs to be learned individually because the MAC address space is non-hierarchic (in contrary to the IP address space)
- Not all network paths allowed by the physical topology can be taken because of the RSTP-induced restriction (some ports are in blocking state to prevent loops).
- In case of network failures, re-convergence towards a new network state can become very slow as a result of the RSTP protocol.

[2]Some fields include the notation '+x' to indicate that the VLAN ID (VID) is actually contained in a Tag Control Information (TCI) field which mainly consists of the VID, but also includes a Protocol Code Point (PCP) of 3 bits, and a Drop Eligible (DE) bit to indicate frames eligible to be dropped in the presence of congestion.

	Connectionless Ethernet	Connection-oriented Ethernet
Classic Ethernet	IEEE 802.1D MAC Bridge - MAC address-base forwarding - Learning & Flooding - RSTP-constrained topology IEEE 802.1Q VLAN Bridge - Restrict broadcast by VID	
Carrier Ethernet	→ *Evolved forwarding plane with more scalability, but maintain RSTP* IEEE 802.1AD Provider Bridge - Separation C-VID vs. S-VID (Q-in-Q) IEEE 802.1AH Provider Backbone Bridge - MAC-in-MAC tunneling	→ *Evolved control (eg. GMPLS) and forwarding plane (eg. label switching)* Ethernet Label Switching - Link local S-VID label IEEE 802.1Qay Provider Backbone Bridge-TE - Domain-wide B-DA + B-VID label

Figure 2.2: Overview of Ethernet technologies

2.2.2.1 Connectionless (CL) Ethernet

A multitude of extensions have been developed to overcome the given limitations. A first category of solutions borrows from the main nature of Ethernet bridging, being a connectionless forwarding protocol. The given extensions have lead to several extensions to the Ethernet frame header format to be used, these are shown in Figure 2.1. The Multiple Spanning Tree standard (IEEE standard 802.1S) allows to configure a different tree topology per VLAN. In turn, this capability allows to make better use of the physical topology by restricting them into complementary logical trees.

The Q-in-Q standard (IEEE standard 802.1ad Provider Bridging) allows two levels of VLANs: Customer-VLANs and Service-VLANs, which can be nested thanks to an additional frame header field of 12-bit, being the placeholder for the Service-VLAN.

Although both extensions allow for a reasonable degree of traffic stream isolation in combination with traffic relative prioritization using IEEE 802.1p priority codepoints, they do not solve the issue of all network nodes having to learn all MAC addresses (thus needing one entry per address) in the network, leading to the bad scaling behavior of the forwarding tables as shown more in detail in [3].

Real scalability improvements are met with the MAC-in-MAC standard , defined in IEEE 802.1ah Provider Backbone Bridging (PBB). This allows to separate the provider MAC address space from the customer MAC address space using additional fields in the extended Ethernet frame header (see Figure 2.1). Mapping customer MAC addresses to Provider Backbone MAC addresses at the edge of the provider backbone network solves the forwarding table scalability, as no customer MAC addresses need to be learned in the core of the Provider Backbone Network. In addition, the standard adds, as part of the adaptation of the customer MAC frame (performed at the network edges), a service tag (the I-tag) that allows for additional traffic stream isolation, for up to 2^24 services streams.

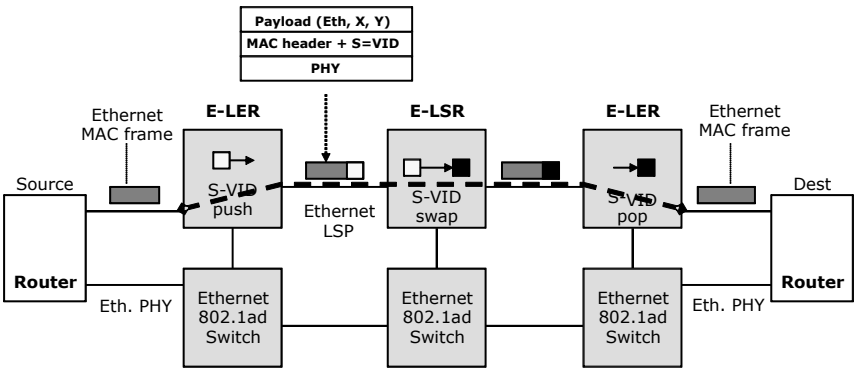


Figure 2.3: Forwarding in an ELS network

2.2.2.2 Connection-oriented (CO) Ethernet

Whereas the given connectionless extensions mainly solve the first two or three limitations, they still inherit on the characteristics of the RSTP-protocol, resulting in relatively bad network usage and bad re-convergence behavior (see [4]). Therefore several protocols have been developed which depart from (by disabling) the learning, the flooding behavior and the distance-vector protocol RSTP.

Routing freedom can now be obtained by maintaining state per logical data path (connection-oriented behavior) and control protocols to discover the topology and control protocols to explicitly signal logical connections across the Ethernet network. These design objectives typically result into the re-use of advanced control protocol suites such as GMPLS (RFC 3945) making use of link-state routing protocols such as OSPF(-TE) and RSVP(-TE) for signaling (i.e. for data path provisioning).

One of the technology proposals resulting from these objectives is PBB - Traffic Engineering (PBB-TE, [5]) which builds further PBB logic, using its frame header and forwarding behavior. PBB-TE can create logical connections and forward frames based on a combination of the B-VID (Backbone Vlan-ID) and the B-DA (Backbone Destination Address). These fields are invariant along its path towards the destination (domain-wide unique label). Because in this paper we focus on Ethernet Label Switching, being the CO technology to which we dedicated the following subsection, we won't further go into detail of PBB-TE, for this we refer to [5].

2.2.3 Ethernet Label Switching (ELS)

ELS is a CO-like scheme that intends to use GMPLS as its control suite with OSPF-TE for routing and RSVP-TE for signaling logical data paths over an Ethernet Network. ELS uses the Provider Bridges (802.1ad) standard to perform label switching in a similar way as is done in MPLS (RFC 3031). It encodes the label in the S-VID tag field of the related frame header (see Figure 2.1). The Ethernet S-VID label space has link local scope and local significance: thus providing 4096 (12 bits) values per interface and allowing intermediate ELS switches to translate the S-VID value resulting logically into a label swapping operation as it is the case in MPLS networks.

The logical data paths established using ELS are called Ethernet label switched paths (E-LSP). Intermediate nodes are called Ethernet Label Switching Router (E-LSR). Ingress/egress ELSR where a LSP starts and ends, provide for a Ethernet Label Edge router (E-LER) functionality. Figure 2.3 describes the label operations along an Ethernet LSP.

When a native Ethernet frame arrives to the ingress LSR, its E-LER function based on the information of the frame header, pushes the corresponding label (i.e. adding an S-TAG with the appropriate S-VID value). Then, the Ethernet VLAN-labeled frame is forwarded along the Ethernet LSP. For each E-LSR, the label is swapped (i.e. that the incoming S-VID is translated into an outgoing S-VID as defined in IEEE 802.1ad). When the frame reaches the egress LSR, its E-LER function pops the label (the S-TAG and so the S-VID are removed). Finally, the frame is sent to its destination as a native Ethernet frame.

It is important to underline that ELS maintains a control state per logical data path but keeps the forwarding paradigm of existing Ethernet switches unchanged except for the fact that forwarding entries are defined per port. In a sense, signaling is used to restrict the incoming and outgoing S-VID per port. The rest of the forwarding process is as per IEEE 802.1ad.

2.3 Carrier Ethernet emulation and simulation

2.3.1 Challenges

Evaluating (Carrier) Ethernet technologies is not an obvious task. Several reasons can be found for this, but the most prominent one is that except for Ethernet bridging and VLAN bridging no simulation environment supports out-of-the-box the discussed Ethernet technology enhancements (see overview in Figure 2.4). This is not surprising, given the fact that these Ethernet flavors are very recent.

In addition, evaluating Carrier Ethernet technologies sets other requirements than bridged Ethernet: time and resource performance requirements are more severe, scalability needs to be a lot higher and manageability is more important.

	NS2	Omnet++	Click	Linux
Type	simulation	simulation	emulation	emulation
Ethernet support				
Bridging (802.1D)	x	x	x	x
VLAN Bridging (802.1Q)	x		O	x
RSTP (802.1W)			O	x
MSTP (802.1S)				
PB (802.1AD)			O	
PBB (802.1AH)				
GMPLS support				
OSPF-TE	x			x (Dragon)
RSVP-TE	x	x		x (Dragon)

Figure 2.4: Ethernet components in simulators and emulators

2.3.2 Network simulation

Network simulation is a technique where the properties of an existing, planned and/or non-ideal network are modeled in a software environment in order to assess performance, predict the impact of change, or otherwise optimize technology decision-making. Simulation thus involves some level of abstraction such as to calculate the interaction between the different network entities (hosts/routers, data links, packets, etc) possibly using mathematical formulas. Typically simulations are executed on a single computation system.

Evaluating scalability or deducing performance trends of technologies is a task for which simulation is well-suited. Most network simulators use discrete event simulation, in which a list of pending ”events” is stored, and those events are processed in order, with some events triggering future events – such as the event of the arrival of a packet at one node triggering the event of the arrival of that packet at a downstream node.

A direct constraint of a network simulator is the dependency on the accuracy of the model that is used. Because there is not necessarily a one-to-one mapping between resources in the simulation and resources of an actual environment, simulation environments are often better for detecting certain trends than for having an accurate and sensitive platform for measuring time and system performance. The authors of [6] illustrate for example that simulated models typically do not account for low level entities such as cpu type, busses, network devices, etc.

The most popular free simulation tools are NS2 [7] and Omnet++ [8]. Figure 2.4 illustrates the carrier Ethernet functionality that they support. In [9] a Carrier Ethernet simulation study has been done using the TOTEM-simulator [10] which was extended for modeling GMPLS-enabled Ethernet Label Switching (GELS) and the BridgeSim [11] simulator for modeling RSTP behavior. This software was used for characterizing LSP acceptance rates, bandwidth placement, link utiliza-

tion and convergence time of both technology classes. The tools in the study were an excellent tool to acknowledge the trends of GELS being more efficient in most of these areas. However the fact that assumptions were made for parameters such as signaling, reservation or switching delay, and the way they are linked, indicates that the resulting accuracy of time convergence numbers are relative and depending on these. For an accurate representation of measures which are also depending on complex processes at different levels, such as cpu load, packet loss, delay or other time related numbers, more sophisticated tools are needed.

2.3.3 Network emulation

Network emulation is a technique where the properties of an existing, planned and/or non-ideal network are imitated in order to assess performance, predict the impact of change, or optimize technology decision-making. As emulation actually mimics the technology under benchmarking, one network device part is typically imitated by a computation system running custom software, e.g., a server blade or PC. Therefore, to emulate a whole network setup, typically a set of server blades or PCs is needed.

Emulation needs more detailed development work than modeling in simulation. The largest gain of emulation regarding simulation is that various performance measures can be made more representative and detailed as its sensitivity is close to the intended end design, and interoperability can be accurately tested. At the other hand, scalability tests require a large number of executing nodes.

Linux is a platform which can be used relatively easy for network emulation. Because of its open-source nature, the built-in protocol stack can be adapted such as to imitate specific router- or switch-alike behavior. The Click Modular Router [12] is a software package which builds on this opportunity by putting a set of frequently used packet processing software components at the disposal of the network engineer developing emulation equipment. Built-in components in support for Ethernet emulation are also shown in Figure 2.4.

2.4 Emulation architecture

2.4.1 Emulation network architecture

For benchmarking ELS technology, an execution environment was set up using the Click Modular Router to emulate the forwarding plane, and Dragon Virtual Label Switch Router (VLSR) GMPLS software to emulate the control plane [12] (Figure 2.5). Both were modified such as to allow ELS forwarding and control as described in earlier sections.

This means that an emulated ELS network consists of a set of Linux PC's running these software packages. To allow arbitrary topologies with ELS emulation

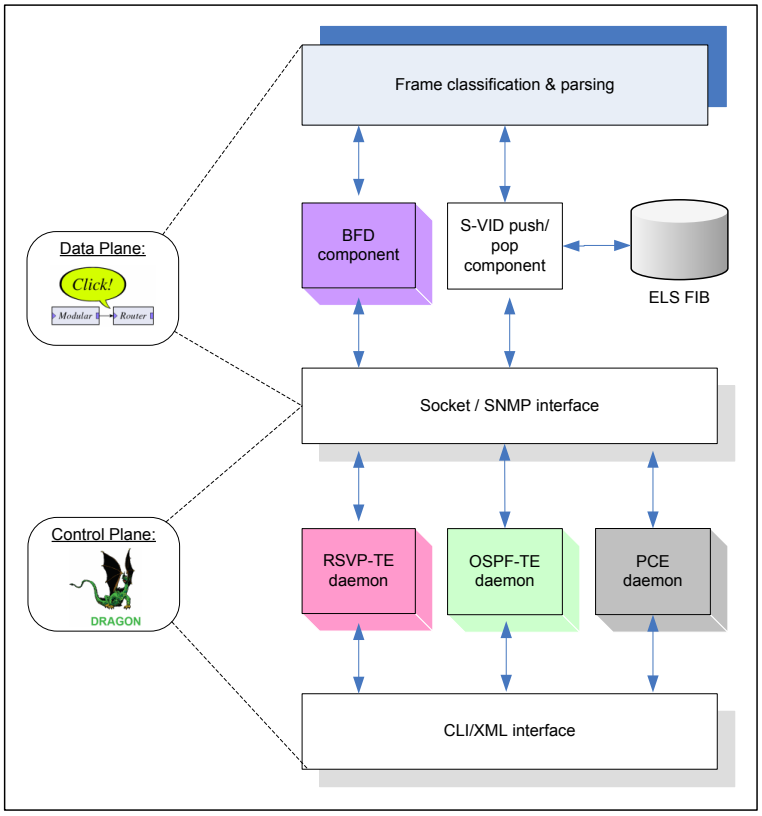


Figure 2.5: Emulated ELS switch architecture

software, Emulab software was used on our local virtual wall at IBBT. Using a NS2 look-alike configuration script, Emulab software allows to define which software image needs to be installed on which PC part of the execution environment, and how several PC's need to be interconnected to each other, defining thus the topology.

2.4.2 Emulated forwarding plane

As shown in Figure 2.4, standard components in Click are available for frame handling, parsing of untagged Ethernet frames as defined in IEEE 802.1D, basic MAC frame learning, forwarding and flooding. For the ELS experimentation, our development tasks consisted first of implementing (by building further on the existing components) single VLAN tagged IEEE 802.1Q and double VLAN tagged IEEE 802.1ad frame handling and parsing. The double tagged Ethernet MAC frame header allowed us to experiment with VLAN translation capability as defined in

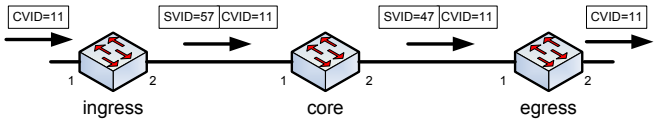


Figure 2.6: Emulated ELS forwarding diagram

		ingress switch	*core switch*	*egress switch*
FTN table	*Incoming interface:*	1		
	CVID:	11		
	NHLFE index:	2		
ILM table	*Incoming interface:*		1	1
	Incoming SVID:		57	47
	NHLFE index:		2	3
NHLFE table	*NHLFE index:*	2	2	3
	Outgoing SVID:	57	47	UNTAG
	Outgoing interface:	2	2	2

Table 2.1: Emulated ELS forwarding table usage

IEEE 802.1ad, forming the base of the label swapping functionality of the ELS switch.

The latter borrows from existing terminology of MPLS forwarding as defined in RFC 3031. In practice, this means that three main tables were developed such as to allow the setup of end-to-end E-LSPs. Instead of using IP-prefixes to match incoming frames in the head-end E-LER to the configured E-LSP, the incoming port index, possibly in combination with the incoming (customer) VLAN-tag, is determining the E-LSP to be used, i.e., outgoing frames are tagged with the S-VID associated to that LSP on the corresponding outgoing link). As illustrated in Figure 5.4, the following tables and associated actions were implemented :

- FTN (FEC-TO-NHLFE) The FTN table is used at the ingress (or head-end) ELS switch to match or classify incoming frames (based on their incoming interface and VLAN-tag) to entries in the NHLFE table.

- ILM (Incoming Lable Map) The ILM table is used at intermediate and egress (or tail-end) ELS switches to match or classify incoming ELS frames based on their incoming interface and S-VID label to entries in the NHLFE table. Together with this matching operation, a POP action is performed on the incoming S-VID label, meaning that the S-VID label is taken away from the frame header, such as to make PUSH actions subsequently possible.

- NHLFE (Next Hop Label Forwarding Entry) The NHLFE table is used at

ingress, intermediate and egress ELS switches in order to forward them into the ELS network having a S-VID label on which can be further switched. This implies that a PUSH action , meaning that an S-VID label is attached to the frame header.

By implementing above functionality, port- and C-VID-based LSPs between ELS edge switches can be set up. This is similar to an Ethernet Virtual Private Wire Service (VPWS) as defined in RFC 4664, being the basic Ethernet service with two attachment points that providers can offer.

The Click ELS component can communicate with external parties by three mechanisms: the built-in handler functionality, an SNMP-interface supporting a newly defined MIB based on NET-SNMP software and a Berkeley socket-based interface. The last technique was developed because the SNMP-interface in our implementation was not optimized enough to reach responsivity below 10 ms.

2.4.3 Emulated control plane

Standard Dragon VLSR GMPLS software consists of SNMP functionality (RFC 2674) which is able of configuring VLANs on a restricted set of commercial Ethernet switches. This allows Dragon to use RSVP-TE signaling to trigger the creation of continuous end-to-end (point-to-point) VLANs in a VLAN bridged Ethernet network.

To enable creation of ELS LSPs, triggered by Dragon GMPLS software, we re-used and extended the above SNMP framework such as to enable interaction based on a customized MIB for the defined ELS forwarding plane. The defined MIB not only allows to configure the required S-VID translation, but also allows to request what S-VID labels are still available on a specific link/interface. This is functionality is needed, because ELS uses link local S-VID labels instead of domain-wide unique VLANs as is the case in the default Dragon implementation using OSPF-TE to discover available VLANs on each link.

Using one of the control interfaces of the Dragon control plane (CLI, XML-driven or hard-coded RSVP-TE functions) one can setup LSPs between edge nodes in the ELS network. As indicated earlier, the SNMP-interface in Click proved to behave rather slowly (compared to the expected behavior). Therefore, additional components were made in Dragon such as to enable Berkeley socket-based configuration of the ELS forwarding plane.

2.5 Experimental results

The framework as described in previous sections has been used to perform a set of base time and resource performance studies of ELS forwarding and control

functionality. This section describes a methodology for these experiments and discusses obtained results that are representative of the capabilities provided of our emulation environment.

2.5.1 Methodology

To ease the roll-out of an emulation experiment a methodology was set up as shown in Figure 2.7. To roll out an emulation experiment on a certain topology, one needs to install the same software on every node (Dragon, Click and dependencies), with each node having a different set of configuration files for both control (Dragon) and forwarding software components (Click).

To facilitate the creation of configuration scripts, a software tool was developed in JAVA. The tool expects as input an XML-file defining the desired emulation topology and a template for the configuration files of Dragon and Click which need to be generated for every node separately. As output it creates a set of files supporting the process of setting up the emulation experiment, being:

- An NS2-like configuration script is generated in order to trigger the roll-out of an Emulab experiment, involving the setup of network links and installation of software images on network nodes. The configuration script also specifies the needed IP-addresses for both out-of-band control (which is used in Dragon) and data plane links.
- Configuration scripts are generated for every node for both Click forwarding and Dragon control.
- Shell scripts to create GRE tunnels in support of the out-of-band control channels between emulated controllers.
- Shell scripts to enable easy centralized control of the different network nodes, including central start- and stop functionality of control and data plane functionality and templates to set up E-LSP's.

Using the information from above enables setting up a running emulation network with active forwarding and control plane. Using the pre-generated shell scripts of the Java-tool, additional scripts can now be manually made such as to trigger the creation of a set of LSPs and traffic generators such as Iperf [13] or D-ITG [14] in the network and accompanying monitoring tools.

Once the above shell scripts have been created, they can be started such as to run the actual experiment. This will trigger output including log files and trace dumps which can be post-processed such as to present them in graphs and scientific results.Measuring performance of the ELS network can be done at end-to-end level using standard Linux tools such as TCPdump [15], and resource usage at

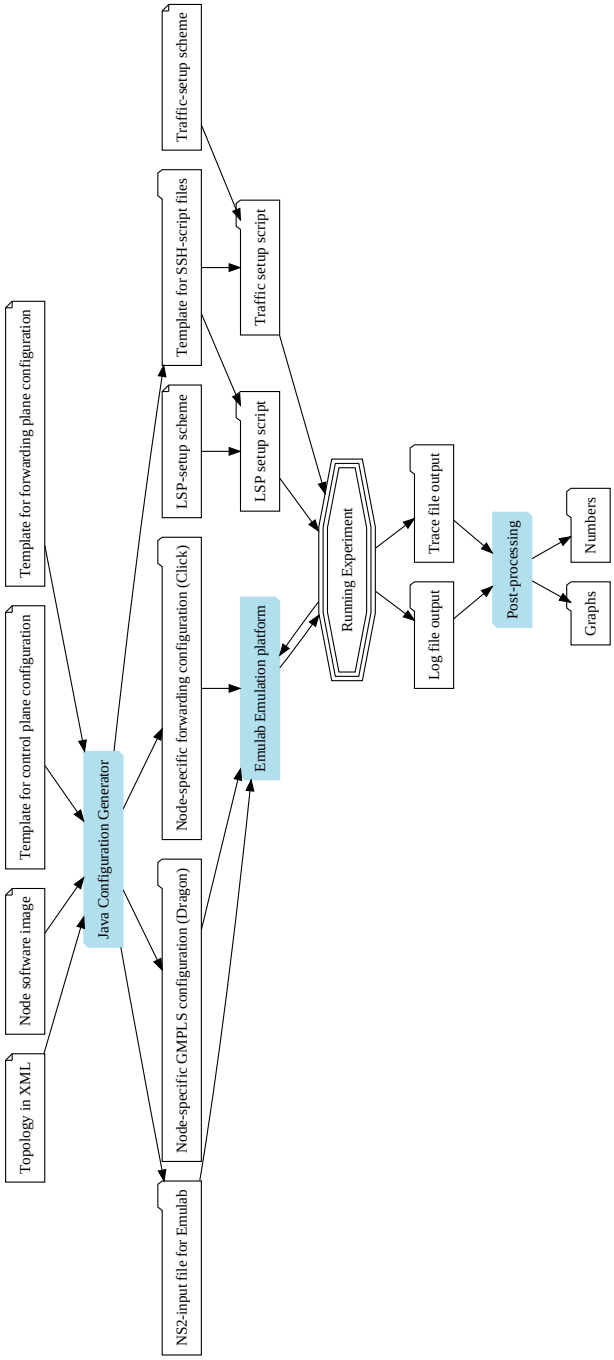


Figure 2.7: Detailed methodology flow chart

link or node level can be done using logging and counter functionality which was incorporated into Click or Dragon software components.

2.5.2 Node performance

To measure the performance of the (Click-based) forwarding plane of an emulated ELS node, we have set up a 2-hop LSP on which we sent traffic with speeds varying from 100 Mbps to 1 Gbps.

The graph in Figure 2.8 illustrates that the throughput performance of an emulated ELS node is good. Even at high bit rates, the emulated switch is capable of keeping track with high input rates up to 1 Gbps. More specific, the emulated ELS switch forwarded with zero loss at all loads below 1 Gbps, having relatively stable delay. At 1 Gbps throughput some packet loss is visible (36 percent). This is probably due to the fact that the CPU is going into overload, a condition which would not appear when performing switching in hardware.

The effect of the varying traffic loads on the emulated ELS node regarding CPU usage is shown in Figure 2.9. A trend of linear increasing CPU usage can be seen, up to the point of a load of 1 Gbps. At that point a significant increase in CPU usage was encountered (almost doubling), resulting in instability of the emulated node (that is Linux PC).

The time that is needed within the Dragon GMPLS software in order to process ELS-related RSVP-TE messages is in the order of tens to hundreds of ms, depending on the type of node (ingress, core or egress), based on average numbers of a 3-node LSP setup as shown in Figure 2.10. Configuring the Click-forwarding plane triggered from Dragon-GMPLS software only takes 3 ms on average.

2.5.3 Network and LSP performance

In order to measure the performance of LSPs, a 16-node ring network was used such as to set up LSPs having different lengths, varying from hop lengths of 1 until 15.

The resulting setup time for the shortest possible LSP fell in a range of 150-200ms. Using the average numbers that were found in the 3-node LSP (Figure 2.10), increasing the LSP length would result in adding an additional (core) node to the LSP, boiling down to adding 130 ms of additional provisioning time. These results were experimentally verified, as can be seen in Figure 2.11. This results in an overall linear increase regarding provisioning time for LSPs with increasing lengths.

Besides the provisioning time of the LSPs, the quality of LSPs was measured for different lengths. Again LSPs were set up having 1 to 15 hops. Once set up they were loaded with traffic streams of 500 Mbps, and resulting loss, delay and jitter was measured. Figure 2.12 shows that on average no loss was measured over

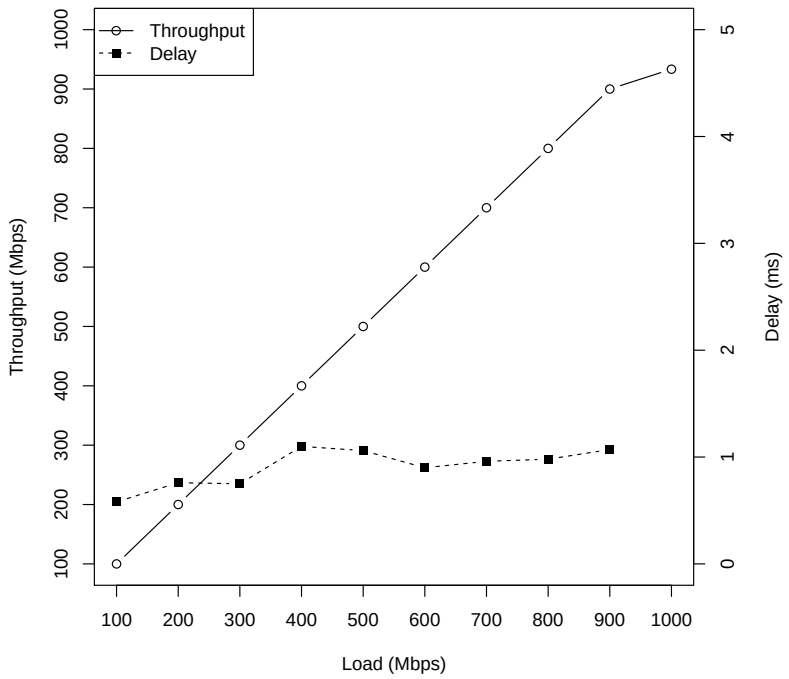


Figure 2.8: ELS throughput and delay in emulation

the LSPs being set up with any hop length. Give the very small scale on which jitter is shown, we can say that jitter remained more or less stable on the LSPs having different hop lengths.

Given the stable results in node, network and LSP performance within the emulated ELS network, we were able to validate both the ELS technology to be used as a carrier Ethernet technology, and the emulation environment as a test platform.

2.5.4 BFD failure detection and ELS segment protection

Having a GMPLS control plane, ELS technology can be enriched with several highly advanced features that this protocol suite allows. Whereas bridged Ethernet typically depends on RSTP with or without hardware failure detection to recover from node or link failures, ELS can now make use of techniques such as segment protection as defined in RFC 4873 such as to enable fast switch-over in failure cases.

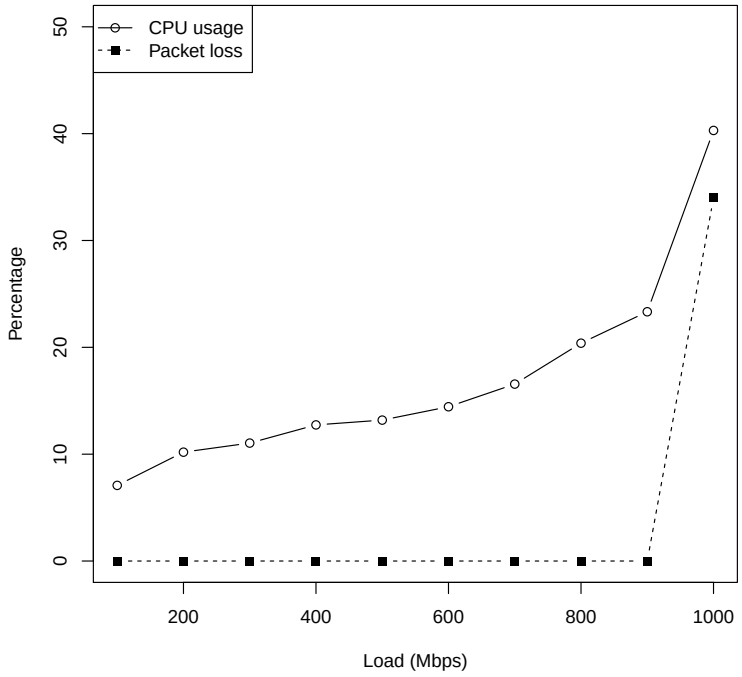


Figure 2.9: CPU usage and packet loss of ELS forwarding in emulation

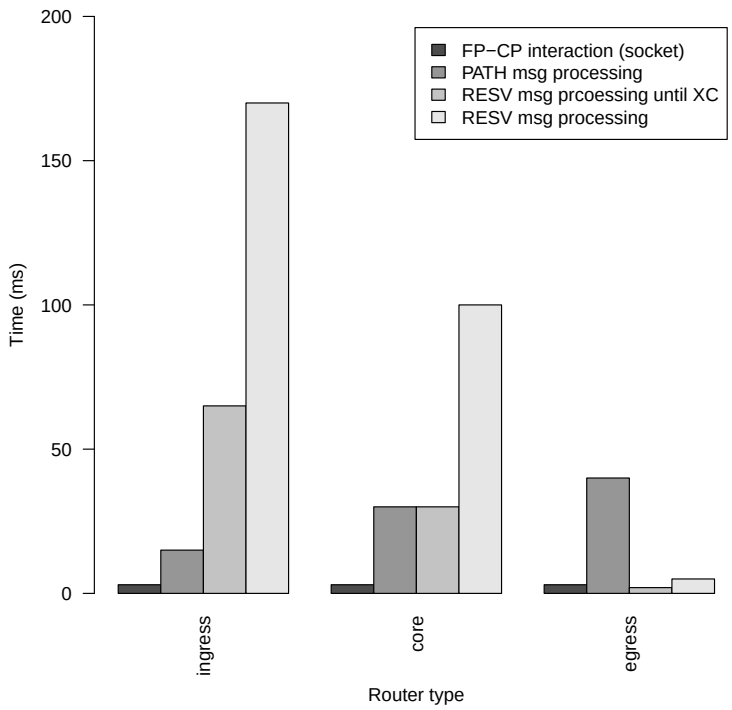


Figure 2.10: Times to configure forwarding (Click) from GMPLS (Dragon)

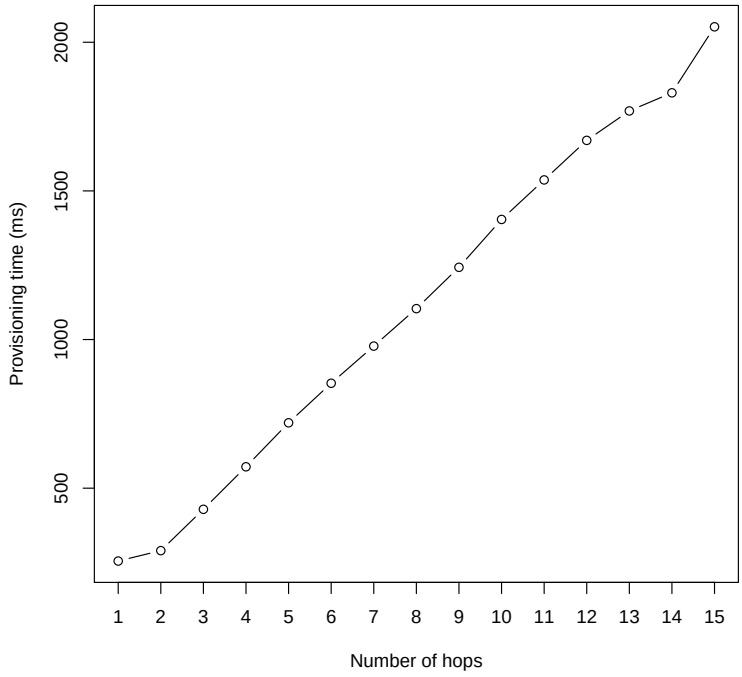


Figure 2.11: Control plane provisioning time (Dragon-based)

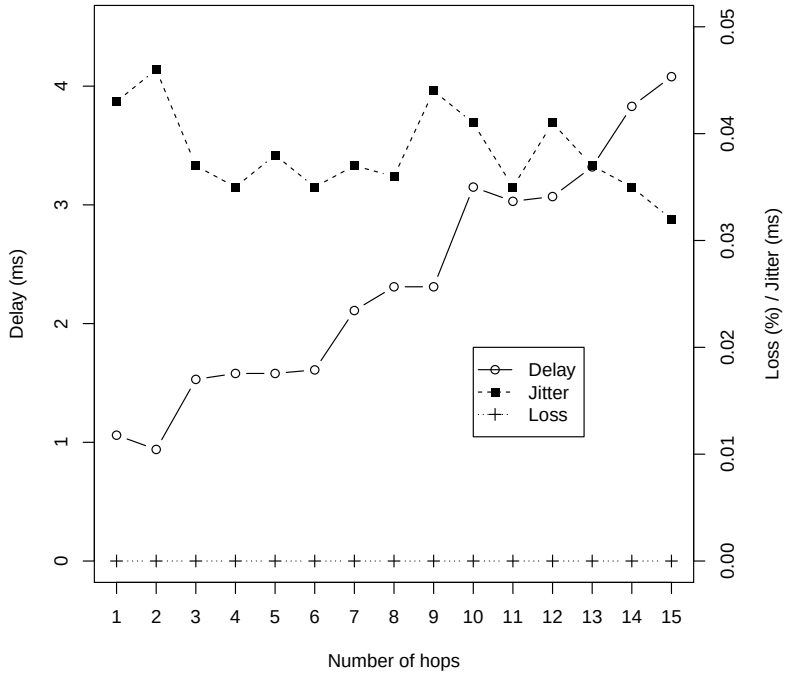


Figure 2.12: Emulated ELS LSP quality (500 Mbps LSP)

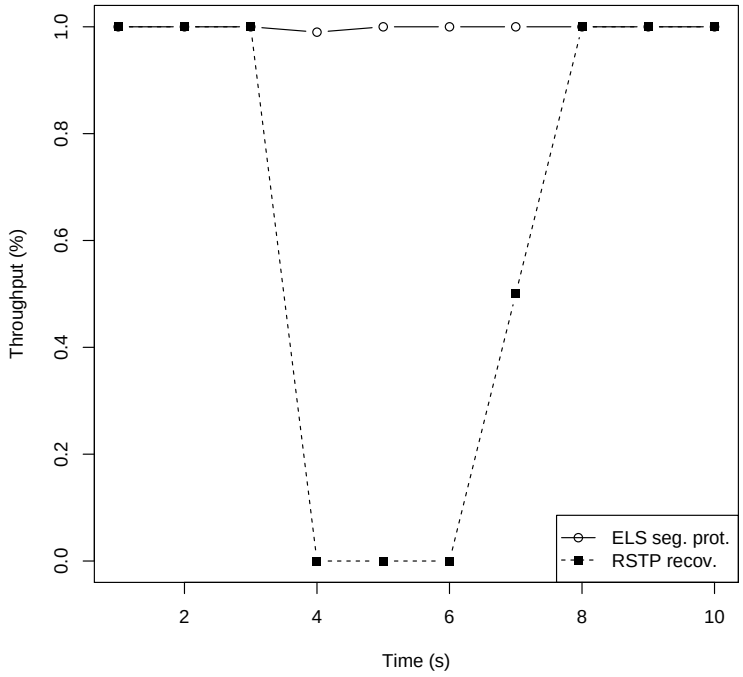


Figure 2.13: ELS segment protection vs. RSTP-recovery in throughput on failure

As discussed in more detail in [16], we also implemented Bi-directional Forwarding Detection (BFD) running directly over Ethernet within the emulation platform. BFD is a highly advanced failure detection mechanism, which allowed us to reach failure detection times below 20 ms.

Having implemented a base form of segment recovery in the Dragon control software in addition to BFD failure detection, we were able to also validate ELS' recovery performance in emulation network, compared to RSTP-based recovery. Figure 2.13 shows the difference in percentage of throughput during the event of failure between ELS segment protection and RSTP-based recovery using hello-timers for failure detection . More details can be found in [16].

2.6 Conclusion

An emulation environment using open-source technology has been developed enabling to test prototypes of Carrier Ethernet technologies. Furthermore, the developed emulation environment provides the flexibility to experiment with these technologies on arbitrary topologies with accuracy which is impossible in simulation environments.

We have implemented Ethernet Label Switching (ELS) and were able to successfully benchmark time and resource consumption of the technology. The framework was extended to allow basic forwarding plane failure detection and recovery mechanisms. Future plans exist to further extend it such as to allow also VPLS services.

2.7 Future work

Whereas basic VPWS-functionality was implemented and presented in this paper, future work will consist of implementing Virtual Private LAN Services (RFC 4664) having multiple attachment points in our framework. This will allow to emulate realistic Layer 2 VPN scenario's.

In addition to the existing support for point-to-point LSPs, the support for point-to-multipoint LSPs will be considered as future work.

As our implementation was mainly intended for prototyping, the entire protocol setup using both modified Dragon source code and home made Click components did not always perform with the expected stability. In case real stress tests are envisioned over longer periods, more stability in the emulation environment would be desired, and future work would consist of optimizing the developed prototype code.

Acknowledgment

This research is partly funded by The Institute for the Promotion of Innovation through Science and Technology in Flanders (IWT-Vlaanderen) through the TIGER project [17] in the European CELTIC framework and the FP-7 project BONE.

References

[1] *Service Provider Plans for IP/MPLS*, January 2008. Infonetics.

[2] *IEEE 802.1W. IEEE Standard for Local and Metropolitan Area Networks - Rapid Spanning Tree Protocol*. IEEE.

[3] D. Colle, W. Tavernier, and A. Gladisch. *Ethernet: beyond the LAN?* DRCN2007, La Rochelle, France, pages 07–10, 2007.

[4] K. Elmeleegy, A. Cox, and T. Ng. *On count-to-infinity induced forwarding loops in ethernet networks*. In Proc. IEEE Infocom, pages 1–13, 2006.

[5] *IEEE 802.1Qay. IEEE Standard for Local and Metropolitan Area NetworksVirtual Bridged Local Area Networks - Amendment: provider Backbone Bridge Traffic Engineering*. IEEE.

[6] R. Chertov, S. Fahmy, and N. Shroff. *Emulation versus simulation: A case study of TCP-targeted denial of service attacks*. In Testbeds and Research Infrastructures for the Development of Networks and Communities, 2006. TRIDENTCOM 2006. 2nd International Conference on, pages 10–pp. IEEE, 2006.

[7] *The Network Simulator NS-2*. http://www.isi.edu/nsnam/ns/.

[8] A. Varga et al. *The OMNeT++ discrete event simulation system*. In Proceedings of the European Simulation Multiconference (ESM2001), volume 9, 2001.

[9] S. Ilyas, A. Nazir, F. Bokhari, Z. Uzmi, A. Farrel, and F. Dogar. *A simulation study of GELS for Ethernet over WAN*. In Global Telecommunications Conference, 2007. GLOBECOM'07. IEEE, pages 2617–2622. IEEE, 2007.

[10] S. Balon, J. Lepropre, O. Delcourt, F. Skivee, and G. Leduc. *Traffic Engineering an Operational Network with the TOTEM Toolbox*. Network and Service Management, IEEE Transactions on, 4(1):51 –61, 2007. doi:http://dx.doi.org/10.1109/TNSM.2007.030105.

[11] *BridgeSim*. http://www.cs.cmu.edu/~acm/bridgesim/index.html.

[12] E. Kohler, R. Morris, B. Chen, J. Jannotti, and M. Kaashoek. *The Click modular router*. ACM Transactions on Computer Systems (TOCS), 18(3):263–297, 2000.

[13] A. Tirumala, F. Qin, J. Dugan, J. Ferguson, and K. Gibbs. *Iperf: The TCP/UDP bandwidth measurement tool*, 2005.

[14] S. Avallone, S. Guadagno, D. Emma, A. Pescapè, and G. Ventre. *D-ITG distributed internet traffic generator*. In Quantitative Evaluation of Systems, 2004. QEST 2004. Proceedings. First International Conference on the, pages 316–317. IEEE, 2004.

[15] V. Jacobson, C. Leres, S. McCanne, et al. *Tcpdump*, 1989.

[16] W. Tavernier, D. Colle, M. Pickavet, and P. Demeester. *GMPLS-controlled Ethernet segment protection*. In Optical Communication, 2008. ECOC 2008. 34th European Conference on, pages 1–2. IEEE, 2008.

[17] L. Ciavaglia et al. *Tiger: Optimizing ip & ethernet adaptation for the metro ethernet market*. In Proc. European Conference in Networks and Optical Communications (NOC 2007), 2007.

3

Packet loss reduction during rerouting using network traffic analysis

In this chapter, the recovery process of an IP router as steered by IGP routing protocols is modeled. We characterize the packet loss as a result of the update order of forwarding entries, and propose a heuristic to reduce the packet loss in combination with a traffic model.

⋆⋆⋆

W. Tavernier, D. Papadimitriou, D. Colle, M. Pickavet, and P. Demeester.

Published in Telecommunication Systems, Springer, 2011

Abstract Upon certain network events, such as node or link failures, IP routers need to update their affected routing table entries. During the period between the failure occurrence and the installation of the updated entries (on the line cards), the network traffic is lost when forwarded by routers that are still using old entries. Indeed, current IP routers do not involve network traffic information during this unordered update process. The consequence is more packet losses compared to a process that would order these entries based on local traffic information. In this paper, we model and predict network traffic passing through an IP router and define two dynamic heuristics in order to reduce the packet loss resulting from routing

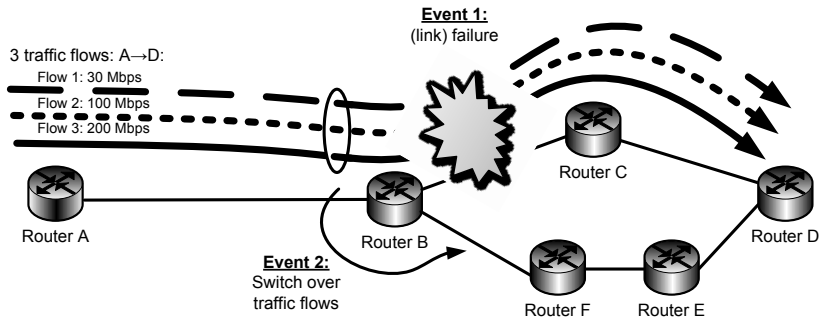


Figure 3.1: IP backbone router about to update routing table entries corresponding to three traffic flows

table updates. AutoRegressive Integrated Moving Average (ARIMA) - Generalized AutoRegressive Conditional Heteroskedasticity (GARCH) traffic models are used in combination with heuristics that dynamically sort the routing entries and improve the low-level routing table update process. In a realistic simulation environment, we show that this setup can result into a decrease of packet loss, depending on: i) the network traffic model, ii) the applied heuristic, and iii) the network traffic aggregation level.

3.1 Introduction

Current IP networks run dynamic routing protocols to automatically compute their routing tables. For this purpose, link-state routing protocols are commonly used in today's IP networks, and referred to as Interior Gateway Protocols (IGP). The (re-)convergence time of routing protocols can take (tens of) seconds upon topological change(s) resulting from, e.g., link or node failures. Henceforth, many techniques have been developed to help IP networks reducing the convergence time of their routing tables upon occurrence of these events. Their common target is to make sub-second (re-) convergence possible [1] in order to prevent the disruption of the traffic flows affected by the routing table updates resulting from these events.

Fast failure detection techniques ensure detection times in the order of milliseconds, for example, using hardware-based loss-of-signal detection or software-based Bidirectional Failure Detection (BFD, [2]). Recovery techniques such as plain IP Fast ReRoute (FRR, [3]) aim to minimize the duration of traffic delivery disruption caused by link or node failure(s) until the network re-converges on the new topology. Common to all these techniques is their focus on recovery time and availability of a (loop free) backup path.

While the backup path can be known (i.e. pre-calculated) at the moment of

failure detection, the low-level update of routing entries at the IP level can still take more than 1 second in IP backbone networks [1]. Therefore, updating one forwarding entry before another can be of high value in order to reduce the transient losses of packets. This can be illustrated by the scenario in Figure 3.1. The figure presents the simplified situation of an IP backbone router B needing to forward 3 traffic flows from router A towards router D. Whereas the shortest path for these traffic flows is via router C, a failure between router B and C forces router B to update its routing table. As will be discussed in more detail in the next section, the default IP rerouting process foresees no specific order for the update of the routing entries corresponding to the three traffic flows. For example, updating the flows in order (1,2,3) would be perfectly viable. Because packet loss occurs as long as the entries are not updated (traffic is sent towards a black hole), 200 Mb could be lost if the routing entry corresponding to the third flow would only be updated as last in the batch after 1 second. Clearly it would be beneficial if the update order would be (3,2,1). While in this example only 3 routing table entries need to be updated, in a realistic scenario of an IP backbone network with 10 Gbps or 40 Gbps links where routers need to update thousands of routing entries, packet loss reduction techniques during rerouting events can make a big difference.

The study conducted in [4] was the first to show that, under the simplified assumption that network traffic remains stable during the routing table update, a decrease of packet loss could be obtained when changing the update order of routing table entries (see Section 3.3). In this paper, we show that realistic network traffic in a IP backbone network does fluctuate during sub-second periods. Therefore, more accurate network traffic pattern knowledge is needed to model and predict network traffic during the process of updating routing table entries. Given more advanced traffic models, we show that the routing table update process can be further improved on a lower process level using heuristics that dynamically calculate: i) the update order of the routing entries, and ii) the quantum time of low-level router process in order to decrease the resulting packet loss.

The paper is structured as follows. In Section 3.2 we describe how routed IP networks deal with network failures and investigate in more detail the default local rerouting process. The concept of traffic-informed rerouting is introduced in Section 3.3, containing a state-of-the-art overview and outlining the approach followed in this paper. Traffic monitoring as explained in Section 3.4 is the first step of the outlined process in the previous section. In Section 3.5 we analyze realistic network traffic traces of IP backbone routers and investigate their sub-second dynamics. The same section then gives an overview of a set of simple and more advanced state-of-the-art models for network traffic characterization and prediction. Given the input of network traffic models, Section 3.6 formalizes the goal of packet loss reduction and describes two packet loss minimization heuristics usable during the traffic-informed routing table update. The resulting set of techniques is

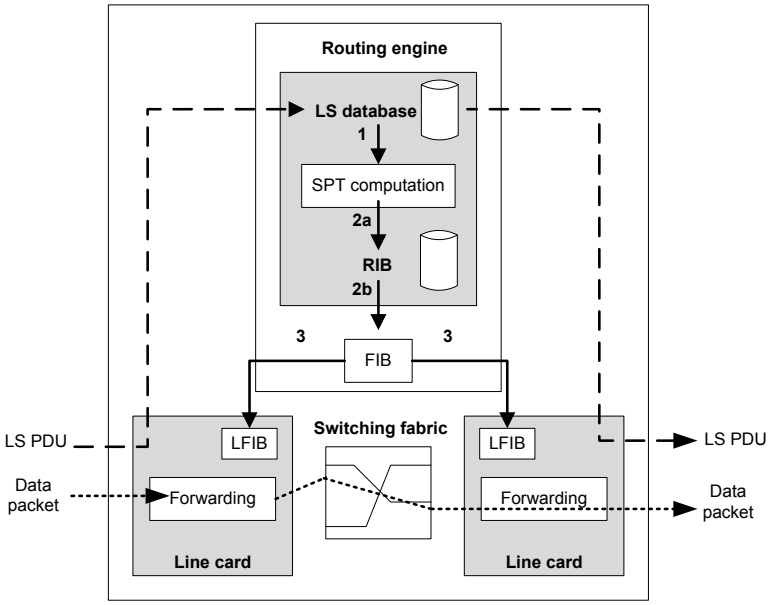


Figure 3.2: The router update process

benchmarked in Section 3.7, quantifying the accuracy of traffic models, and measuring the resulting packet loss and recovery time consequences with respect to the used traffic aggregation levels. Section 3.8 recapitulates the resulting computational complexity, and a conclusion is formulated in Section 3.9.

3.2 The traffic-agnostic IP routing table update

In order to determine which processes can be improved to reduce the packet loss resulting from rerouting events, this section will describe the IP routing table mechanics in more detail. The usual IP router consists of two components: a forwarding engine and a routing engine. When a data packet arrives at an incoming interface of an IP router, the forwarding engine looks for an entry in its Forwarding Information Base (FIB) and performs a longest prefix match on the destination IP address of the incoming packet. It then forwards the traffic towards its next hop according to the best matching entry. The FIB can either be manually configured or can be populated by routing table entries computed by a routing protocol.

Routing protocols ensure that topology information is distributed over the network, or the Autonomous System (AS), such that individual routers can maintain a consistent and up-to-date full view of network topology. We focus on IP backbone routers operating Link-State (LS) Interior Gateway routing Protocols (IGP) such

as the Open Shortest Path First protocol (OSPF, [5]). These flood LS protocol data units (PDUs) over the network, containing information about the local links and MA (multi-access) networks a router is connected to. All received LS PDUs are collected into a database (*the LS database*) which allows a router to have a complete view on the network link topology and to calculate shortest paths towards different destinations (IP addresses) or network parts (IP network prefixes). The LS database is updated by either detecting a local connectivity change (e.g. failing link or interface), or by receiving an LS PDU from a peering router.

A router detecting a link failure originates a LS Update (LSU) message. This message, containing LS Advertisement(s) (LSA) which describe the topological link state change(s), is reliably disseminated in the network (flooding). At every router receiving the LSU, the following three-step process executes (see Figure 3.2):

1. Re-computation of the *shortest path tree* (1) using the topological information stored in the updated LS DataBase (LSDB);

2. Update of the central *Routing Information Base (RIB)* and the central *Forwarding Information Base (FIB)* based on the Shortest Path Tree (SPT) computation (2a and 2b).

3. *Distribution of central FIB* towards the local FIB (LFIB) on line cards (3).

The SPT is usually re-computed in its entirety and takes about 30 to 50 μs per IGP destination prefix. Time optimizations can be obtained using incremental SPF (iSPF) algorithms [6, 7]. The second step consists of updating the central RIB and FIB, using the calculated shortest paths. This uses about 50 to 100 μs per destination prefix [1]. Typically, this step happens in (pseudo-)parallel with step 3, which is about distributing the central FIB entries towards the line cards' LFIB. Running step 2 and 3 in (pseudo-)parallel, means that they both use the central CPU in interleaved time slots, swapping between both processes for updating and distribution. This process can be compared to the usual process scheduling in time-sharing Operating Systems (OS) such as Linux, whereas commercial routers make use of a hard real time OS. The consecutive time the central CPU is spending to perform central RIB/FIB entries update(s) or distribution of FIB entries towards the line card(s) is determined by the quantum of the swapping process. The quantum time can typically be configured between 10 and 800 ms. Similar quantum time values were found in [1]. In practice, the local re-convergence results into a series of update-distribution batches, where during a first time quantum a fixed set of prefixes are updated towards the central RIB/FIB, followed by a time quantum during which the same set of prefixes is distributed towards the LFIBs.

By default, the update of RIB/FIB entries (i.e., IGP prefixes) and their distribution are not ordered. Moreover, the size of the batches is determined by the

quantum time of the router process, usually a preconfigured constant number over the entire update process. In this context, the proposed technique aims to sort the routing entries in decreasing bit rate order at update time while taking into account the size of the update-distribution batches.

3.3 Traffic-informed rerouting

3.3.1 State-of-the-art

The previous section illustrated that traditional IP routers are currently not designed to efficiently handle the scenario depicted in Figure 3.1. However, at first sight, several relatively simple approaches may be followed. For example, one might assign priorities to routing table entries corresponding to IP destination prefixes, as implemented by some vendors [8]. The resulting priorities then lead to a predefined update order of the routing table entries. While this may be feasible for small networks, given that the network operators have sufficient knowledge about the spatio-temporal properties of the traffic transported by their networks, it is definitely not practical or scalable for larger IP backbone networks, as it requires manual intervention.

An alternative approach to the problem was formulated in [4]. The main idea of this work relies on ensuring that routing entries corresponding to higher bit rate traffic should be updated and distributed (from the RIB towards the local FIB) earlier than those corresponding to lower bit rate traffic. However, the study was performed using generated network traffic from Pareto distributions with the assumption that network traffic behaves persistently (stable or constant traffic) during the routing table update process (which can take more than 1 second). As we will show in the next section, this assumption does not hold for real network traffic, leading to the need for more sophisticated traffic models and packet loss reduction heuristics.

3.3.2 Suggested approach

In order to thoroughly tackle the problem of traffic-informed routing table updates, we suggest the inclusion of the following router components (Figure 3.3):

- a *monitoring component* to maintain traffic volume and rate statistics associated to different IP prefixes (routing entries)
- a *traffic modeling component* to characterize and estimate the traffic volume and rate associated to IP prefixes (routing entries)
- a *packet loss reduction component* to configure the resulting order and size of update batches upon recalculation (see Section 3.2).

Each of components will be discussed in larger detail in the following sections.

3.4 Traffic monitoring

Given the evolution of routers' line cards in terms of high bit rate interfaces (including transmitters/receivers, framers, etc.) and fast packet processors (on-board high frequency CPUs), real-time network traffic monitoring per IP prefix (corresponding to a routing entry) has become feasible [9–11]. As previously motivated, obtaining statistics of the traffic properties can be of high value during the event of router updates. The following methods exist to gather data such as to estimate the traffic volume associated to an IP destination prefix at failure detection time:

- *Online statistical counters* measure aspects of transiting traffic in a router using counters, for example the number of packets per destination prefix or used packet size distribution curves (for example [12]).
- *Traffic sampling* by means of samplers. Instead of counting certain traffic characteristics, unmodified traffic is captured for some time interval. This sample is then used to derive certain characteristics, for example done in Netflow-alike settings (e.g. [13]).

As we will not focus on the metrology problem, we redirect the reader to the above-mentioned references for more detailed information. However we assume the following running conditions. We assume the monitoring system to be *passive* in the sense that no additional traffic is inserted into the network for the sake of monitoring[1]. In fact, active monitoring wouldn't be of any specific help, because we don't measure the liveness of the forwarding path, or the available bandwidth on the forwarding path.

We expect the monitoring system to be *independent* and *distributed*, meaning that we do not assume any form of centralized monitoring entity. Next, it is our assumption that the monitoring system is based on an *open loop*, meaning that the counts or estimations do not lead to actions modifying the corresponding values.

3.5 Network traffic analysis

Using network monitoring techniques, the volume of traffic per IP destination prefix (corresponding to the routing table entry) can be measured for a given time

[1] *Active monitoring* techniques wouldn't be of any help with respect to the characterization of the consumed bandwidth per destination prefix, as they typically check the available bandwidth or the liveness of an individual forwarding path. However, for the specific goal of detecting link failures (for triggering the rerouting process), detection techniques such as BFD can be considered as active measurements, as they introduce *Hello messages* to check the liveness of a link.

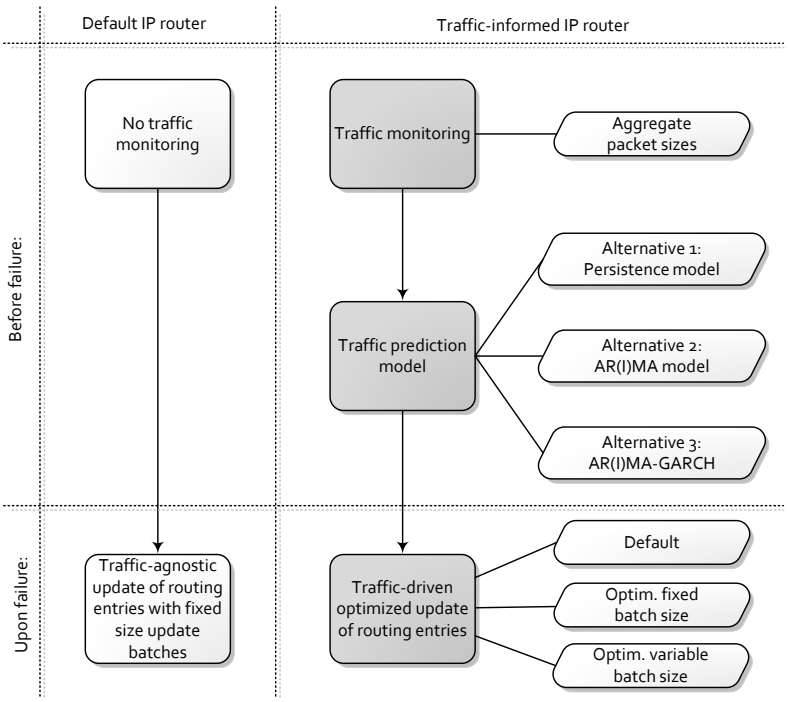


Figure 3.3: Default IP router vs. a traffic-informed IP router

interval. Aggregating traffic arriving within a given time interval or time bin is referred to as *temporal traffic aggregation*. Larger bin sizes lead to more aggregation and typically smoother traffic curves, as may be observed in Figure 3.4a. Another level of traffic aggregation can be applied at the subnet level, by taking packets whose destination IP addresses belong to the same subnet together (smaller subnets result into more aggregation). This is referred as *spatial aggregation*.

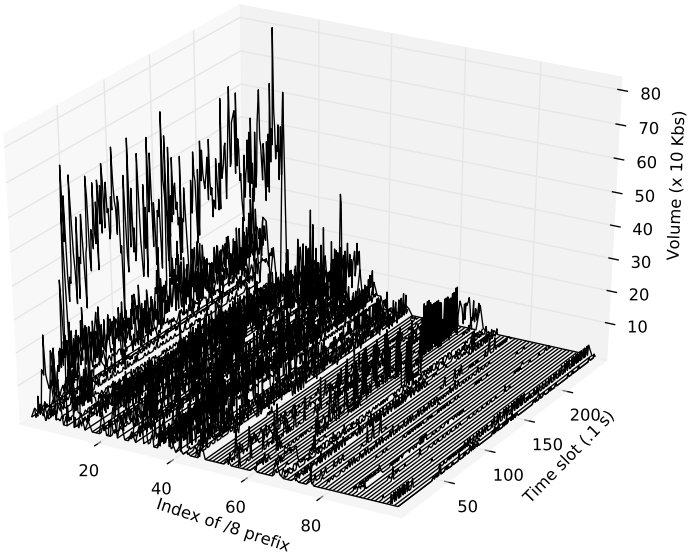
3.5.1 Network traffic persistence

One might assume, as in previous work (see Section 3.5.2.1), that network traffic in an IP backbone network remains stable (persistent) during short (sub-second) time periods. To investigate the validity of this persistence assumption, we analyzed a set of trace files obtained from the Samplepoint F monitoring point deployed in WIDE backbone network (related to the MAWI project [14]). Samplepoint F monitors traffic on a trans-Pacific line (150 Mbps link) that is in operation since July 1st, 2006. We analyzed trace files at several aggregation levels. We inspected the files at the level of /8, /16 or /24 IPv4 subnet prefixes (spatial aggregation), using time intervals of either 100, 500 and 1000 ms time bins (temporal aggregation). This results into 9 aggregation level combinations.

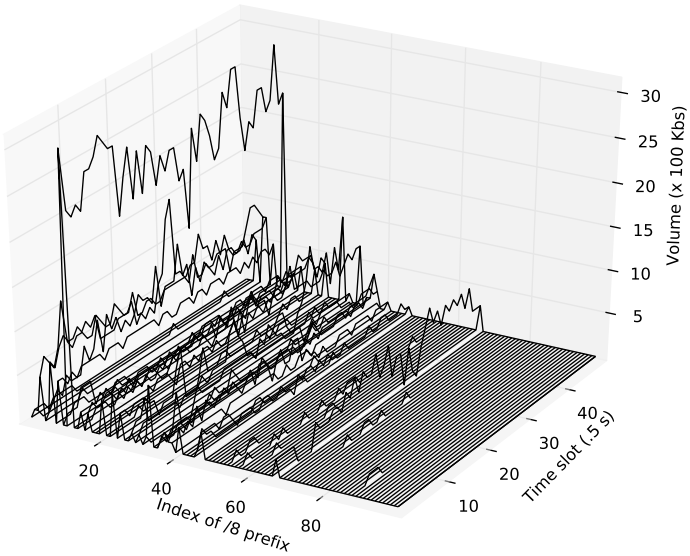
The resulting analysis sets were generated by a preprocessing module generating a two-dimensional table which groups all accumulated packet sizes per subnet for each of the above time intervals. One cell in the table corresponds to a group of packets for the given destination prefix within a given time bin. This resulted into: on average 21K /24 subnets, 8K /16 subnets, and 0.16K /8 subnets for a 15 minute trace containing on average 30 million packets (also see Figure 3.5).

The persistence analysis using time bins of 0.5 s and 1 s, as shown in Figure 3.6, shows two important aspects: i) the of number of active flows, and ii) the number of persistent flows. A flow is considered as active, if a non-zero volume is associated to it in the respective time bin. The number of persistent flows refers to the number of active flows that remain active in the next time bin. These results illustrate that the global activity and persistence level, i.e., the number of flows, is more or less stable over the duration of the traces. However, depending on the aggregation granularity, the persistence level is lower than the activity level. At coarser aggregation levels (using for example /8 subnet prefixes), the persistence-level is close to the activity-level. However, the persistence level for /16 or /24 subnet prefixes is significantly lower (only about 50 to 60 percent of the active flows stays active the next time interval). This high churn rate gives an indication on the high variability of network traffic on a sub-second timescale.

The above observations show that persistence at small timescale may not be assumed, implying that simply using a *persistence traffic model* (as done in [4]) is probably too optimistic for the goal we want to achieve. This is the direct trigger



(a) Aggregating traffic using 100 ms time bins



(b) Aggregating traffic using 500 ms time bins

Figure 3.4: Aggregation

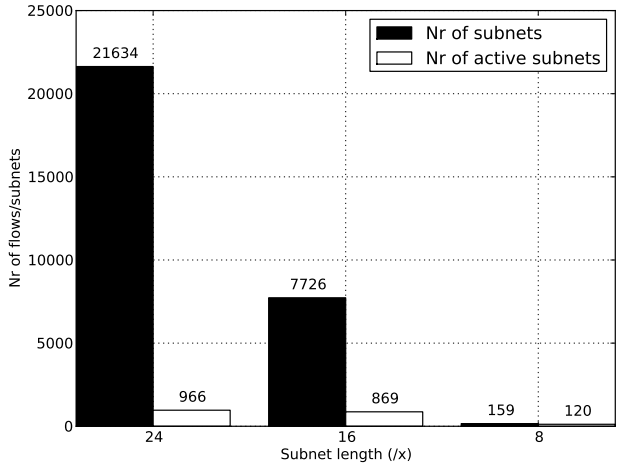


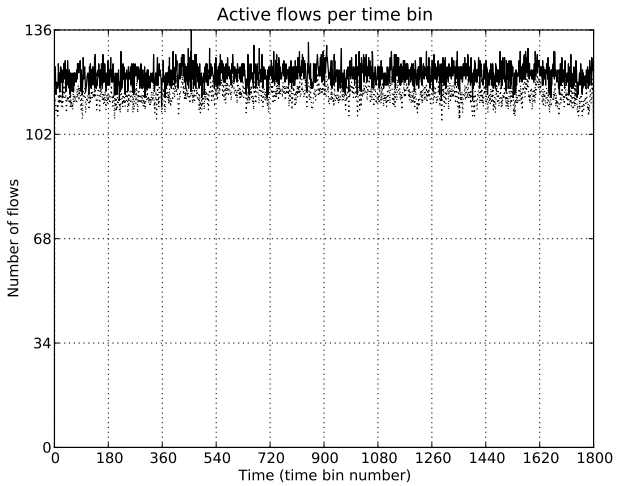
Figure 3.5: Number of flows at different spatial aggregation levels

to evaluate more advanced traffic models in the coming sections. This observation does not seem to hold at larger time scales. Several studies such as [15] have shown that large flows (elephants) and small flows (mice) seem to behave more predictable and persistent over larger time scales (larger than one second).

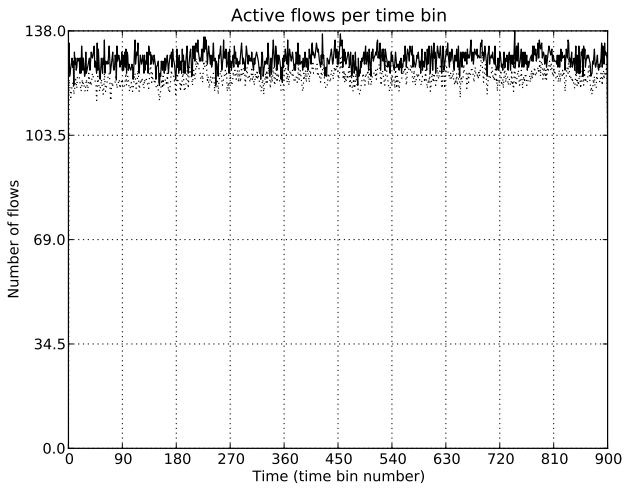
3.5.2 Network traffic modeling

An accurate network traffic model is expected to capture the prominent traffic properties including short- and long range dependence, self-similarity in large time scale, and multi-fractality in short time scale. On the other hand, it has been observed that Internet traffic also exhibits non-stationary and non-linear properties. Therefore, in order to benchmark the given Routing Update Process (RUP) heuristics, one needs adequate traffic models fit and predict realistic IP backbone network traffic. This section summarizes the state-of-the-art of the network traffic models as will be used for experimental evaluation in Section 3.7.

Network traffic analysis studies in the last decades have uncovered the subtle pattern of *self-similarity* in network traffic time series. Stochastic self-similarity describes a statistical property of the traffic and manifests itself in several equivalent fashions: slowly decaying variance, long range dependence (LRD), non-degenerate autocorrelation, or Hurst effect. Intuitively, a process is said to be self-similar if its statistical behavior is independent of time-scale. Formally, a time series $Y = \{y_t | t \in T\}$ is self-similar if its autocorrelation function decays only hyperbolically instead of exponentially [16]. For self-similar processes, the

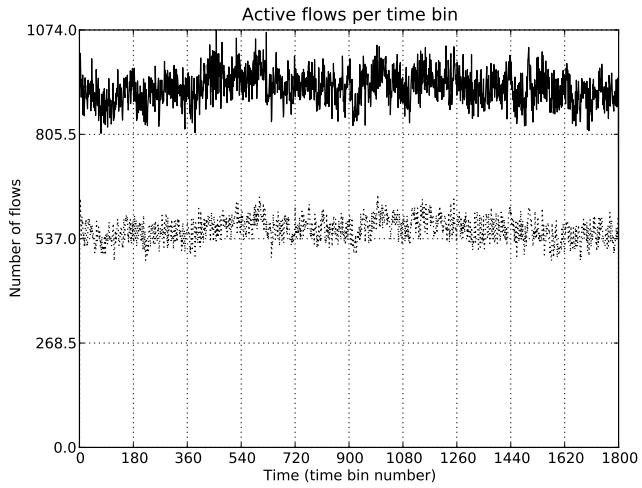


(a) /8 subnet-500 ms

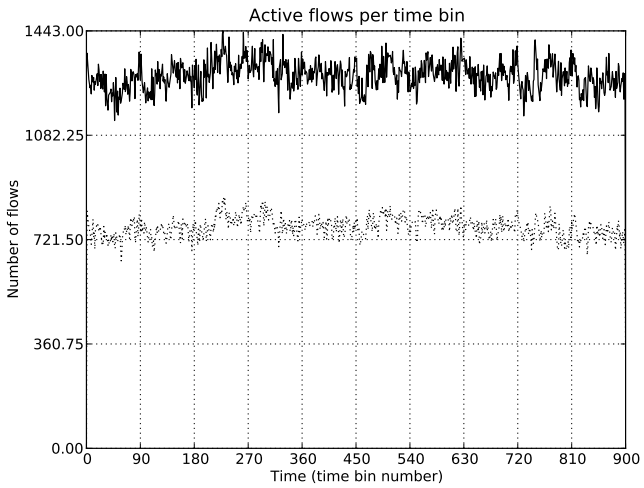


(b) /8 subnet-1000 ms

Figure 3.6: Flow activity and persistence per aggregation level

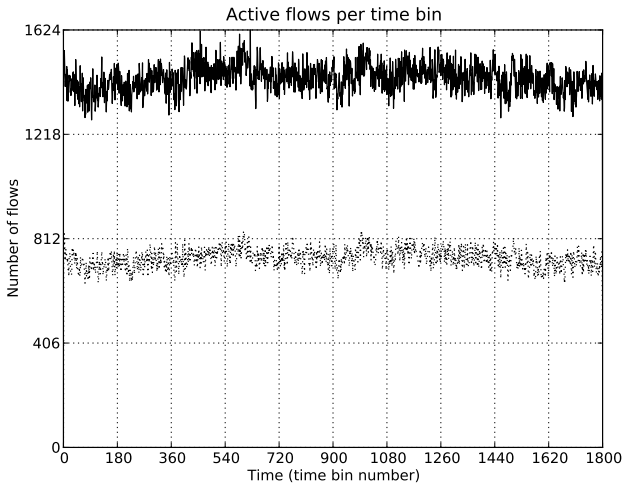


(c) /16 subnet-500 ms

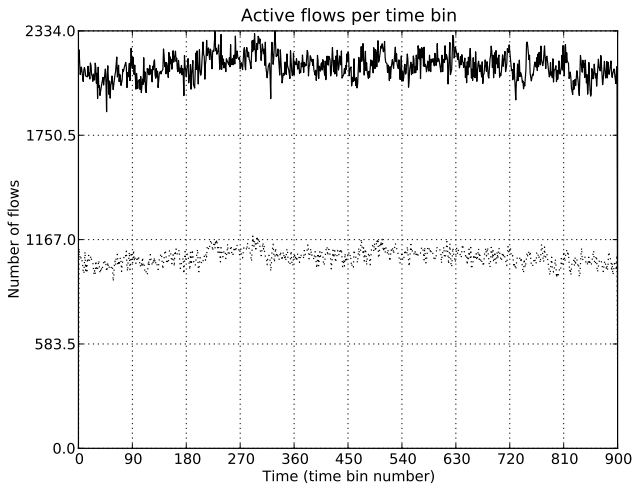


(d) /16 subnet-1000 ms

Figure 3.6: Flow activity and persistence per aggregation level



(e) /24 subnet-500 ms



(f) /24 subnet-1000 ms

Figure 3.6: Flow activity and persistence per aggregation level

autocorrelation function drops hyperbolically (not exponentially) toward zero but may never reach zero (non-summable auto-correlation function). The 'degree' of self-similarity is often denoted by the Hurst parameter, which is a measure of the persistence of a statistical phenomenon, denoting the length of the *long-range dependence* of a stochastic process.

After the seminal work reported in [17] confirmed mathematically in [16], it has been commonly accepted that Internet traffic behaves statistically self-similar [18, 19] and that aggregating streams of such traffic typically intensifies the self-similarity ("burstiness") instead of smoothing it. A more general introduction to the multi-scale nature of network traffic, self-similarity and the Hurst effect can be found in [20].

3.5.2.1 State-of-the-art

Many studies have been conducted to predict the behavior of network traffic on time scales larger than a second. In this section, we give a short overview of these studies some of which also considered traffic prediction on a smaller time scale.

Fractional AutoRegressive Integrated Moving Average (FARIMA) models were successfully used in [21] to predict video, and Internet traffic on a timescale of one second or larger. The self-similar character of network traffic was shown to be adequately captured. However, at smaller time scales, depending on the specific traffic trace, it was also shown that the signal-to-noise (SNR) significantly decreases. As the FARIMA model cannot capture the multifractality which has been observed in the network traffic at short-time scale, [22] introduces the *AutoRegressive Integrated Moving Average* (*ARIMA*) - *Generalized Auto Regressive Conditional Heteroscedasticity (GARCH)* model. The resulting *ARIMA-GARCH* model is a non-linear time series model which combines the linear ARIMA model (with conditional mean but constant variance) with a conditional variance GARCH model (as will be further detailed in the next section). It captures both SRD and LRD traffic properties and observes self-similarity and multifractality. Results obtained using the ARIMA-GARCH model show that the predictive performance of the ARIMA-GARCH model outperforms traditional FARIMA model. Nevertheless, even if the prediction results can capture the actual traffic very well, some prediction errors can not be avoided because the real trace dynamic variance is out of the expectation of the GARCH model variance prediction.

In [23] wavelet neural networks proved to fit the self-similar structure of network traffic quite well too. Again, the authors concluded that predictions are dependent on the timescale – sub-second prediction performing consistently worse – and on the specific type of the network trace. The work presented in [24] analyzes the predictability of network traffic bit rates using *AutoRegressive Moving Average* (*ARMA*) and *Markov-Modulated Poisson Process* (*MMPP*) models. Under the assumption that those models are appropriate, authors of [24] developed analytic

expressions to describe bounded predictability. Traffic aggregation and smoothing proved to monotonically increase the predictability of the network traffic. At last, the authors of [25] came to the same conclusion after performing a predictability analysis of large set of traffic traces. The work also shows that a decreasing bin size in aggregating traffic increases the variance of the resulting signal. This implies that short-term traffic evolution (which implies small bin sizes by definition) has a higher variability than longer-term traffic.

Based on the results of prior studies on short-term network traffic modeling, we will further focus on the ARIMA-GARCH model for our purposes. To ease the theoretical introduction, as well as to enable comparison with simpler models, in the next subsections we will introduce the ARMA model, the ARIMA model and the GARCH model separately.

3.5.2.2 ARIMA-models

The *AutoRegressive Moving Average* $ARMA(p, q)$ model is defined as follows:

$$y_t = \sum_{i=1}^{p} \alpha_i y_{t-i} + \sum_{j=1}^{q} \beta_j w_{t-j} + w_t \tag{3.1}$$

This formula combines two techniques: i) autoregression, which reflects that a prediction is based on the signal itself (using p previous values) referring to the first term, and ii) moving averages, reflected by q terms of the white noise series w_t (with $E(w_t) = 0$ and $Var(w_t) = \sigma^2$) which is put through a linear non-recursive filter determined by the coefficients α_i (weighted average). The autoregressive part directly reflects the *Short Range Dependence* (SRD) of a time series. Whereas these models have been successfully applied to model network traffic, e.g. [26], they are also known to be unable of modeling non-stationary time series. Stationarity implies that the probability distribution characterizing the time series mean and variance remains constant over time.

Some non-stationary time series can be made stationary by one or more levels of differencing[2]. Once the resulting differenced time series is stationary, an *ARMA*(p, q) model can subsequently be fit and predictions can be made by integrating the predictions back. The resulting model of this approach is called the *AutoRegressive Integrated Moving Average* (ARIMA) model. The resulting model including the lag operator L[3] for *ARIMA*(p, d, q) is defined as follows (d referring to the number of levels of differencing):

[2]A differenced time series y_t generates a new time series of differences $z_t = y_t - y_{t-1}$

[3]applying lag operator L on y_t generates y_{t-1}, and a power i of L defines a similar recursive process of the order i

$$(1 - \sum_{i=1}^{p} \alpha_i L^i)(1 - L)^d y_t = (1 + \sum_{j=1}^{q} \beta_j L^j) w_t \tag{3.2}$$

3.5.2.3 GARCH-models

The ARIMA model cannot capture the multifractality which has been observed in some Internet traffic traces. For this reason, the Multifractal Wavelet model (MWM, [27]) has been introduced. However, while the MWM model can capture short timescale multifractality, it cannot predict traffic. For this purpose, the work of [28] introduces the ARIMA - Generalized AutoRegressive Conditional Heteroscedasticity (GARCH) model, a non-linear time series model which combines the linear ARIMA model (with constant variance) with conditional variance GARCH model [29]. The term "conditional" implies an explicit dependence of the variance of the noise/innovation series w_t from the ARIMA model on a past sequence of observations. The *GARCH*(r, s) model for the conditional variance σ^2 of the innovation series w_t follows an *ARMA*$(p = r, q = s)$ model of the following form:

$$\sigma_t^2 = \sum_{i=1}^{r} \gamma_i \sigma_{t-i}^2 + \sum_{j=1}^{s} \omega_j w_{t-j}^2 + w_t \tag{3.3}$$

where, r and s are positive integers.

3.6 Packet loss reduction

Once a network traffic model is determined, which is able to adequately capture and predict network traffic behavior during the update of the routing table entries, we return to our initial target of reducing the resulting packet loss. For this purpose, we will formalize the concepts of "recovery time" of an updated routing table entry and the resulting "packet loss" during its update in this section.

3.6.1 Recovery time of an updated routing table entry

Given the batch structure of the router update process (see Section 3.2), all affected traffic flows directed to a given IP destination address are recovered when the following conditions are met: i) the IGP routing (table) entry for the prefix including that IP destination address is updated and stored in the central RIB, ii) the corresponding FIB entry is updated and stored in the central FIB, and iii) the batch number b_i that comprises this updated entry is distributed towards the LFIBs on the line cards. To simplify the description, we will consider that after its recomputation, the RIB and FIB updates are performed altogether by summing their

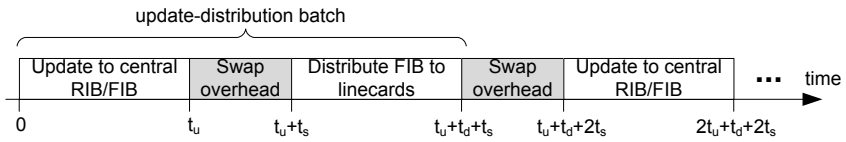


Figure 3.7: Timeline of the router update process

update time (and thus refer to a RIB/FIB entry update). Thus, assuming a fixed batch size of x_u (number) of updated RIB/FIB entries and a given order of these entries, the recovery time of a flow f_i is characterized by the following formula:

$$r(f_i) = b_i x_u (t_u + t_d) + (2b_i - 1)t_s \tag{3.4}$$

Here, t_u refers to the update time for a single RIB/FIB entry, t_d to the distribution time for a single entry, and t_s to the swapping time interval between an update and a batch distribution operation. This process is represented in Figure 3.7.

The concept of recovery time can be illustrated by the following example. Assume that a network failure results into the disruption of 5000 traffic flows (F_{5000}) and that a single RIB/FIB entry is associated to each of their destination. If we use the following values $t_u = 100$ μs, $t_d = 100$ μs, $x_u = 100$, $t_s = 5000$ μs, this configuration results into 50 update-distribution batches, each taking $100 \times (100 + 100) + 5000 = 25000\mu$ s. Taking into account the additional intermediate swapping times between the batches, this results into $50 \times 25000 + 49 \times 5000 = 1495000$ μs, or 1.495 s to recover all flows.

The *Earliest Recovery Time (ERT)* possible for a given flow f_j, j referring to the waiting position (in the batch) of the updated RIB/FIB entry associated to this flow at time t, is defined as follows:

$$ERT(f_j, t) = t + j(t_u + t_d) + t_s \tag{3.5}$$

The ERT is the time interval of the smallest batch including the updated RIB/-FIB entry associated to the considered flow together with the RIB/FIB entries updated before it. Indeed, in this situation only one swapping time interval is needed (separating the update and distribution batch) together with the time needed to update and distribute all prefixes comprised within the batch.

3.6.1.1 Packet loss induced by an updated routing entry

Packet loss occurs for network traffic flows as long as their corresponding RIB/FIB entries are not updated and distributed on line cards. For example, if two network traffic flows are recovered from updates part of the same batch, their recovery time is the same, and packet loss occurs for both flows up to the moment of recovery.

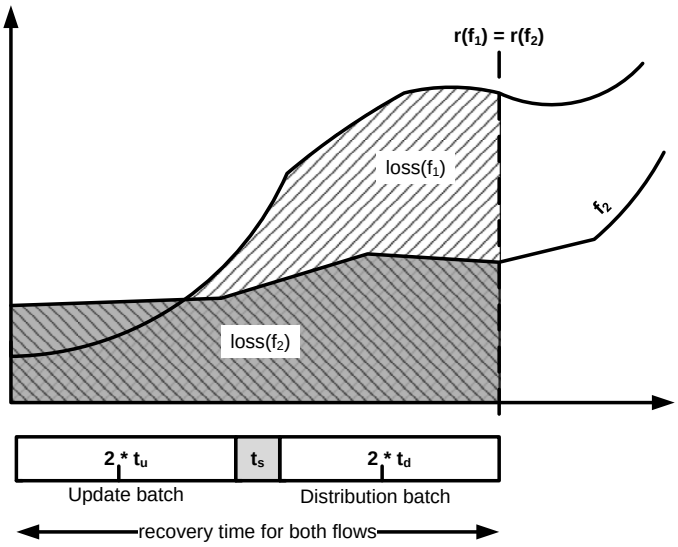


Figure 3.8: Packet loss under dynamic traffic conditions

This is represented in Figure 3.8. The resulting packet loss can be interpreted as the integral over time from the moment the failure happens (time 0 in the figure), up to the moment of recovery ($r(f_1) = r(f_2)$ in the figure). The loss resulting from the recovery of flow f_i relates to its bit rate, $br(f_i)$, through the following formula:

$$loss(f_i) = \int_0^{r(f_i)} br(f_i)dt \tag{3.6}$$

Packet loss calculation can be illustrated for the following simple example. Assume we have a small set F_4 consisting of four flows with associated constant bit rates $br(f_1) = 5, br(f_2) = 2, br(f_3) = 4, br(f_4) = 1$ (in Mbps), and $t_u = 100$ μs, $t_d = 90$ μs, $x_u = 2$, $t_s = 70$ μs, this results into a packet loss of $260 \times 10 + 590 \times 5 = 5550$ Kb.

The above formula illustrates that the order in which RIB/FIB entries are updated, can heavily influence the resulting packet loss. Randomly updating RIB/-FIB entries, as it is usually the case, can result into the situation where recovery of higher bit rate flows is delayed because RIB/FIB entries associated to lower bit rate flows are updated earlier. This delay results in high cumulative packet loss (for example f_3 is proportionally bigger than f_2, but however needs to wait until $f_1 \ldots f_2$ are updated).

3.6.2 Packet loss reduction heuristics

Assuming that we know the time series $br(f_i, t)$ of all flows (using traffic modeling techniques), in this section we formulate heuristics or packet loss reduction functions. Reduction of packet losses can be achieved by: i) updating the RIB/-FIB entries associated to a (set of) flows in a different order, and/or ii) bundling the resulting updated RIB/FIB entries in well-specified consecutive batches. As such, an optimization function is defined as a function which maps a given traffic model (set of well-defined $br(f_i, t)$-functions) at a certain time t to a tuple $(flowordering, batching)$. The first part of the tuple denotes a permutation of the flows, the second part refers to a decomposition of the flows into ordered batches.

We evaluate two variants: a first heuristic which assumes fixed batch size, and thus only reduces packet loss by RIB/FIB entries reordering, and a second heuristic which works on both dimensions, i.e., reducing packet loss by changing both the RIB/FIB entries order and the set of batch sizes.

3.6.2.1 Fixed batch size

Given a fixed batch size of n entries and an adequate traffic model, we can exactly calculate the resulting recovery times for all batches. Once we know in which batch a routing entry is contained, we can calculate the associated packet loss. Therefore, the following approach can be applied.

Consecutively consider all batches to be processed, starting from the first batch, and apply the following iterative process:

1. Calculate the *recovery time for the current batch* of RIB/FIB entries
2. For *every routing table entry* corresponding to flow f_j that is not yet updated, calculate the *resulting packet loss* if its corresponding RIB/FIB entry would be contained in the *current batch* of RIB/FIB entries
3. *Sort* all resulting packet losses in decreasing order
4. *Select* the n entries with the highest associated packet loss and their associated RIB/FIB entry
5. *Terminate the current batch*, and remove the recovered flows from the working set
6. Continue the process from step 1 for the next batch of RIB/FIB entries and the remaining flows.

3.6.2.2 Variable batch size

Building further on the idea of the heuristic described in [4], we can formulate a dynamic variant which takes into account both the flow ordering (thus the order of RIB/FIB entries), and the batch size so as to minimize the packet losses. For this purpose, consider the following scenario. We denote by $F_n = f_1, \ldots, f_n$ the set of traffic flows affected by the failure, and by $b_{current} = (f_i, \ldots, f_{i+s})$ the set of flows for which the corresponding entries still need to be updated on routers' LFIB[4]. In this context, we have two possible options when in the middle of the ordered process of updating the router's LFIB entries (associated to the set of flows F_n) within our current batch of RIB/FIB entries $b_{current}$:

1. *Extension*: extend the current batch with the RIB/FIB entry associated to the next flow f_{i+s+1};

2. *Splitting*: terminate the current batch and put the RIB/FIB entry associated to the next flow into a new update-distribution batch.

We can now compare the additional cost of extension (ec) versus the additional cost of finishing (fin) the update-distribution batch to guide us into the decision above. The idea of the trade-off is illustrated in the simple example with flows to be updated as shown in Figure 3.9. Given a current batch (starting from the first one), the heuristic makes a trade-off between terminating the current batch before the next RIB/FIB entry is inserted into the current batch, or extending the current batch with the next RIB/FIB entry. For example, let's assume our current batch contains the RIB/FIB entries for the flows f_1, f_2 and f_3. According to the iterative character of the strategy, this means that extending the batch with the entries for the flows f_2 and f_3 had lower cost than splitting. In a next step, we can now either again extend the current batch with the RIB/FIB entry for the flow f_4 or split this batch and insert the entry for f_4 in a new update-distribution batch. Figure 3.9 shows the difference between both decisions: an extension delays the recovery time for f_1, f_2 and f_3, while a split delays the recovery time for f_4 (and all flows following f_4). Table 3.1 quantitatively shows the recovery times for all flows in both scenario's and compares them with the earliest recovery time (ERT) possible per flow. The decision $ec < fin$ will compare the difference in recovery time multiplied with the associated bandwidth to compare packet loss of both options.

We can now further formalize the trade-off using the following definition for the extension cost $ec(b_{current})$ for this batch, with $t_{b_{current}}$ as the starting time of the current batch $b_{current}$:

[4]The RIB/FIB entries for the prefixes corresponding to the flows prior to f_i have already been updated in an ordered manner (from f_1 to $f_{(i-1)}$)

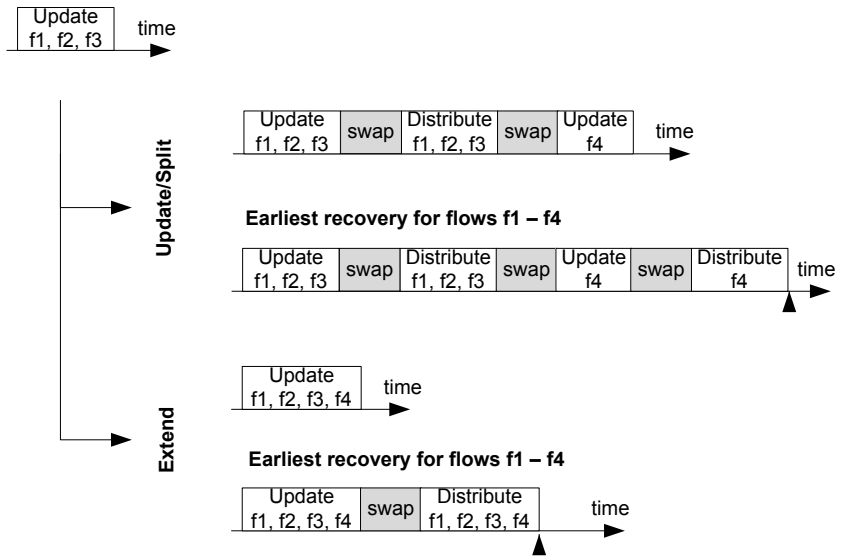


Figure 3.9: Compare extension vs. split-scenario for F_4

		f_1	f_2	f_3	f_4
Earliest recovery time (ERT)		$t_u + t_d + t_s$	$2(t_u + t_d) + t_s$	$3(t_u + t_d) + t_s$	$4(t_u + t_d) + t_s$
Split	Recovery time	$3t_u + 3t_d + t_s$			$4t_u + 4t_d + t_s$
	Diff. with ERT	$2(t_u + t_d)$	$(t_u + t_d)$	0	$2t_s$
Extension	Recovery time	$4t_u + 4t_d + t_s$			
	Diff. with ERT	$3(t_u + t_d)$	$2(t_u + t_d)$	$(t_u + t_d)$	0

Table 3.1: Recovery time comparison of F_4-scenario: split vs. extension

$$ec(b_{current}) = ec_{is} = \sum_{j=i}^{i+s} \int_a^b br(f_j, t)dt \tag{3.7}$$

with

$$a = ERT(f_j, t_{b_{current}}) \tag{3.8}$$

and

$$b = ERT(f_{i+k}, t_{b_{current}}) + (i + s - j + 1)(t_u + t_d) \tag{3.9}$$

The formula expresses the fact that, by extending the current batch, the recovery time of every RIB/FIB entry comprised in the current batch will result into an additional delay compared to the minimal delay it can experience (given the position of that entry in the current batch). The minimal delay an entry can experience is determined by the ERT, i.e., the time elapsing for an entry positioned as the last one in an update quantum having no earlier update quanta. This additional delay, when multiplied with the associated bit rate, allows deducing the additional loss caused by the extension of the update-distribution batch. For example, if the RIB/-FIB entry associated to the first flow f_i was already delayed by s update times t_u (as this entry was not directly distributed but put in the same batch as the s next entries ranging from i to $i + s$), extending the current batch by one element (to reach $i + s + 1$ elements) further delays the recovery time of the entry i. On the contrary, the recovery of the last entry $i + s$ of the current batch will only be delayed by one update time t_u in case of extension of the current batch.

On the other hand, terminating the current batch has also an associated cost, as it will introduce additional delay for the recovery of coming flows, resulting from the additional swapping cost. This termination condition can be formulated as follows:

$$fin_{b_{current}} = \sum_{f_j \notin b_{current}} \int_{ERT(f_j, t_{b_{current}})}^{ERT(f_j, t_{b_{current}}) + 2t_s.} br(f_j, t)dt \tag{3.10}$$

Our configuration strategy now consists in identifying the action with the least associated cost. The overall algorithm can be expressed as follows:

1. *Add all affected routing table entries* to the working set
2. *Sort* all entries in the working set in decreasing order of their current cumulative packet loss $loss(f_j, t_{current})$
3. *Compute both extension and splitting cost:*

- If no current batch exists (RIB/FIB entry associated to the first flow), then *create a batch* and add this entry into this newly created batch.
- Otherwise:
  - If the extension cost is smaller than the splitting cost, then *add* the corresponding RIB/FIB entry in the sorted list to the *current batch*;
  - Otherwise, *create a new batch* and add the corresponding RIB/FIB entry into this newly created batch.

4. *Remove* the added flow from the working set
5. *Repeat* the procedure from step 2 until the working set is empty

3.7 Experimental results

In the previous section theoretical foundations have been developed to model the IP router update process and the network traffic through the router. Both aspects were used to formulate two heuristics enabling packet loss reduction during the update of the LFIB entries (process realized via the update of the corresponding RIB/FIB entries). In this section, real network traces will be used to evaluate the performance gain (i.e., packet loss decrease) obtained by means of the proposed heuristics. First, we detail the overall methodology and experimental environment, then we benchmark the discussed network models of Section 3.5.2. At last, we measure the overall gain of the suggested approach in terms of resulting packet loss reduction.

3.7.1 Environment and methodology

As indicated in Section 3.5, a set of PCAP traces obtained in December 2009 was taken from the MAWI project [14]. These traces were preprocessed using Python scripting at three time interval levels (100 ms, 500 ms and 1000 ms), referred to as time bins, and at three spatial levels (/8, /16 and /24 subnetworks) with respect to their aggregated traffic volume, i.e., aggregated packet sizes per time-bin per prefix. The choice for this set of spatial aggregation levels allows us to set an upper and lower bound on the possible gain that could be achieved, because the typical aggregation level in usual routing operations (especially when distributed to inter-domain environments) is uncommon to be above /8 or below /24. Next, the resulting traffic time series were analyzed and fit to one of the described traffic

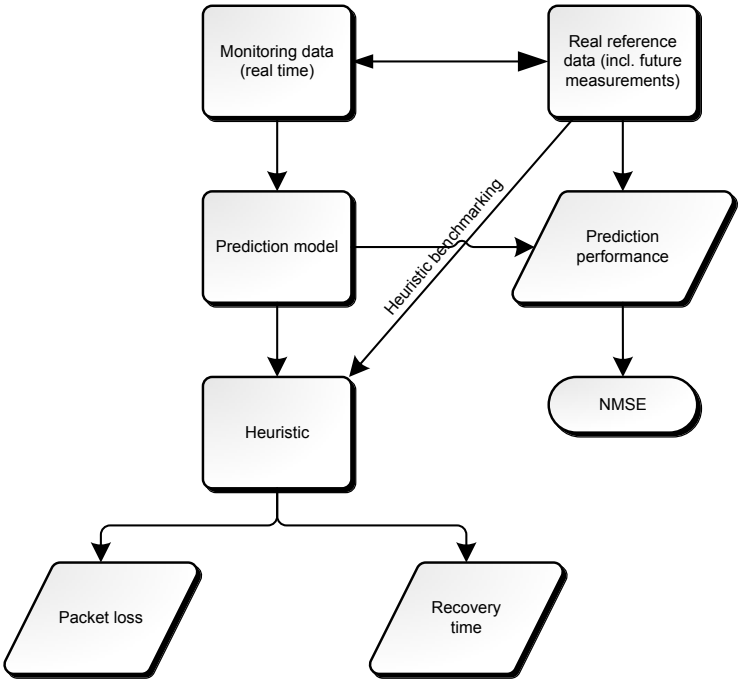


Figure 3.10: Benchmarking process

models using R software [30]. The resulting modeled data was used as input to the heuristic implemented in C++/Python. All the experiments ran on a regular desktop computer equipped with an AMD Athlon 2.7 GHz CPU and 4 GB RAM. The entire process is shown in Figure 3.10. To obtain representative results, all experiments were repeated 1000 times, and the resulting values were averaged.

3.7.2 Network traffic fitting and prediction

The goal of the first experiment is to validate fitting abilities of network traffic modeling techniques introduced in Section 3.5.2 with respect to the given set of MAWI trace files. We evaluated the following variants: i) `PERSIST`, ii) `ARMA`, iii) `ARIMA`, iv) `ARMA-GARCH`, and v) `ARIMA-GARCH`. The first model represents the naive assumption that network traffic for a given time bin will behave exactly the same as the previous time bin (see Section 3.5.1, as in [4]). The latter models refer to those described in Section 3.5.2, and have configurable parameters and different orders determined by their parameters p, q, d and r, s.

The main task in automatic ARIMA forecasting is to select an appropriate model order. To automate the process of optimal parameter selection, we used the order selection strategy of [31] for ARMA and ARIMA models restricted to

maximum values of 10. The strategy selects those parameters which minimize the Akaike's Information Criterium (AIC) among all fitting models. Once the best parameters are selected, Maximum Likelihood Estimators (MLE) are computed to deduce the best fitting coefficients for the ARMA and ARIMA models [31]. The lowest value for the d parameter of the ARIMA model, resulting into a stationary time series, according to the Kwiatkowski-Phillips-Schmidt-Shin (KPSS) test for stationarity[5], is then selected [31].

For the GARCH model, the values of parameters $r = 1$ and $s = 1$ were chosen to reduce the resulting computational complexity, as well as to avoid non-optimizable parameter values due to the lack of sufficient information[6]. The coefficients of GARCH$(1, 1)$ were estimated using the Quasi-Maximum Likelihood Estimator (QMLE) [32].

For one MAWI trace, the fitting procedure at all aggregation levels takes about 20 hours on the referred computer. However, in the context of the router update, this process can be executed in background. Figure 3.11 shows the average Normalized Mean Square Errors (NMSE) of fitting the models to the set of MAWI traces. The NMSE is defined as follows[7].

$$NMSE = \frac{\sum_n (\widehat{y}(x) - y(x))^2}{\sum_n (y(x) - \mu_T)^2} \tag{3.11}$$

The NMSE allows to measure how much of the variance of a signal can be accommodated by a model. Smaller NMSE values correspond to better fitting models. Figure 3.11 illustrates the obtained average[8] fitting errors obtained for the set of referred MAWI traces.

Figure 3.11a shows the error for the models at three different bin sizes (100, 500 and 1000 ms). Figure 3.11b depicts the error with respect to different spatial aggregation levels (i.e. subnet lengths): /24, /16 and /8 subnets.

As might be expected, the persistence model has a significantly larger fitting error than the four other models. In general the error is at typically twice as high. Not unexpected either, the ARIMA-GARCH$(1, 1)$ model gives the best performance on average on all aggregation levels, followed by – in decreasing order of performance – ARMA-GARCH$(1, 1)$, ARIMA, and ARMA model. However, the fitting performance of all more advanced strategies (2 to 5) is rather similar, and differences are small. This result may be understood from the fact that sub-second dynamics of network traffic in IP backbones is very high, resulting into

[5]The KPSS test assesses the null hypothesis that a univariate time series trend is stationary against the alternative that it is a non-stationary unit-root process

[6]Not all time series corresponding to the volumes of a given prefix are dense enough. Some series have more zero values than others, resulting into non-optimizable parameters due to singular Hessian matrices.

[7]$\widehat{y}(x)$ referring to the predictions, $y(x)$ referring to the actual values

[8]over all flows in all trace files

hard fitting and prediction problems. This result also corroborates those obtained by other studies (see state-of-the-art in Section 3.5.2). More aggregation, either by using larger time bins or by using larger subnetworks (smaller network masks or netmasks), leads to better fitting models; however, the relative gain between the techniques does not change drastically. Clearly, more aggregated traffic resulting into smoother and more predictable network traffic (also see Figure 3.4), which explains that the fitting error using /8 subnet levels is significantly less compared to the ones obtained using /16 or /24 subnets for spatial aggregation.

3.7.3 Packet loss evaluation

Given the traffic models of the previous subsection, the goal of the experiments below is to measure the combined gain in resulting packet loss between a set of combinations of traffic models and heuristics described in Section 3.6.2. The following parameters were fixed: the update time t_u needed for one RIB/FIB entry: 100μs, the distribution time t_d towards the LFIB's: 100μs and the swapping time $t_s = 5000\mu$s. For this experiment we assumed that a failure affected 5000 randomly chosen prefixes (if available) from a given MAWI trace. From the resulting routing table update event, packet loss and recovery time was calculated. This procedure was repeated 1000 times to obtain representative average results.

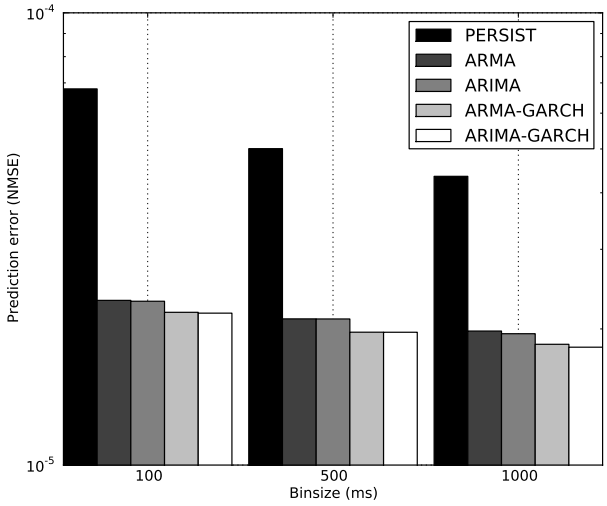
Average results obtained by the evaluation of the resulting decrease in packet loss, is depicted in Figure 3.12. The left part of the figure (Figure 3.12a) shows the resulting packet loss vs. the length of subnets using 100 ms time bins, while the right part (Figure) depicts the packet loss vs. several subnet lengths using 500 ms time bins. The following combinations were evaluated:

1. The default traffic-agnostic router update as described in Section 3.2 with a typical fixed batch size of 100 entries (`default_rup`).

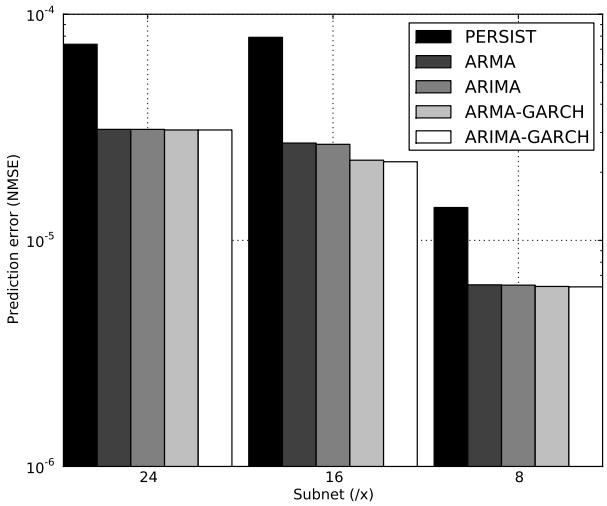
2. The routing update process as defined in [4], assuming persistent network traffic, sorting the resulting routing entries and dynamically updating the batches (`persist_ sorted`).

3. The heuristic using fixed batch sizes from Section 3.6.2.1 in combination with: i) the ARMA models (`arma_fix_ batch`), ii) the ARIMA models (`arima_fix_batch`), iii) the ARMA-GARCH models (`arma_garch_ fix_batch`) or the iv) ARIMA-GARCH models (`arima_garch_fix_ batch`) as fit in Section 3.7.2. The fixed batch size was also set to 100 prefixes.

4. The heuristic enabling variable batch sizes from Section 3.6.2.2 in combination with: i) the ARMA models (`arma_var_batch`), ii) the ARIMA models (`arima_var _batch`), the iii) ARMA-GARCH models (`arma_garch`



(a) Fitting error of traffic model vs. bin size



(b) Fitting error of traffic model vs. subnet length

Figure 3.11: Fitting error of different traffic models

_var_batch) or the iv) ARIMA-GARCH models (arima_garch_var _batch) as fit in Section 3.7.2.

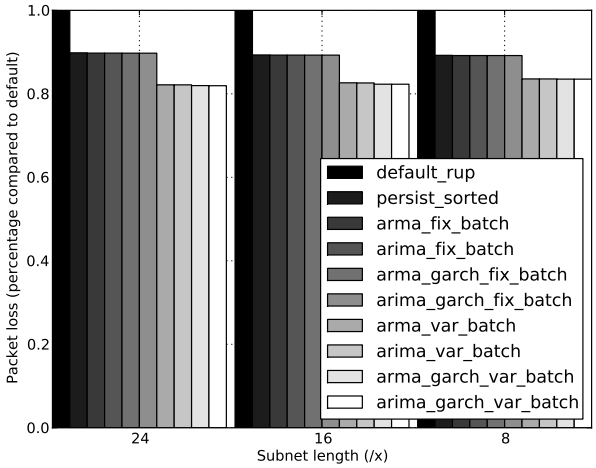
Combinations making use of fixed batch sizes (combination 1, and 3 to 6) were evaluated with 100 entries in a batch ($x_u = 100$). This is a representative choice based on [1] (also see Section 3.2). Smaller batch sizes typically result into relatively too large swapping times (discouraging multiple batches and swapping), and bigger batch sizes result into lower optimization potential as illustrated in the results of [4].

Figure 3.12a illustrates that: i) on average, a decrease of 18 percent packet loss can be obtained using the variable batch size heuristic with any of the proposed prediction techniques compared to the default random routing table update, and ii) about 8 percent packet loss compared to the more obvious optimization technique persist_sorted based on [4] (for the case of using /24 subnets with 100 ms time bins). These results give a more realistic view on the decrease in packet loss one could obtain during routing table updates compared to those in [4] (which reports up to 80 percent decreases). This validates that the high dynamics of network traffic in small time scales makes the problem of traffic-driven routing table updates drastically more difficult than one could imagine at first sight.

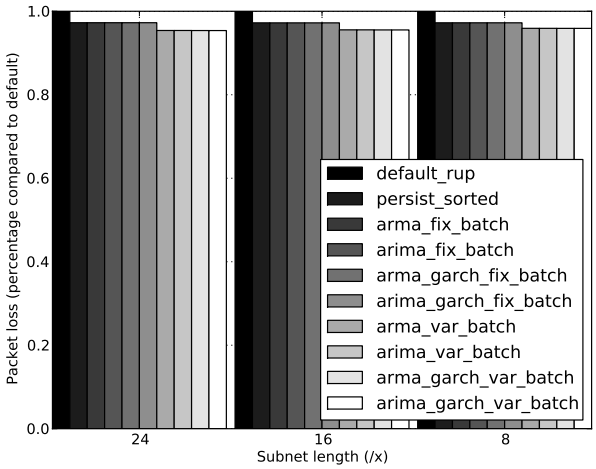
Whereas it is obvious that the default behavior of updating the RIB/FIB entries for IGP prefixes in random order is constant, independently of the traffic prediction model, it is surprising that the specific choice of the prediction model –in combination with the optimization heuristic– has relatively low influence on the decrease in packet loss. This observation may be explained from the fact that the difference between NMSE of the prediction models is rather small (except for the persistence model, see Section 3.7.2), due to the fact that sub-second network traffic modeling is a hard problem.

Figures 3.12a and 3.12b clarify the influence of traffic aggregation on the obtainable decrease in packet loss. Whereas larger aggregation levels improve the correctness of network traffic models (smoother and more equal traffic improves predictability, see Section 3.7.2), it also decreases the gain potential in packet loss compared to: i) default_rup) and ii) (persist_sorted). The more similar traffic behaves, the less difference RIB/FIB reordering makes. The following may be observed: smaller aggregation levels result into smaller packet loss decreases. Whereas in the best case, average reductions of about 18 percent are possible for /24 subnets in combination with 100 ms time bins, only reductions of about 5 percent decrease in packet loss were possible in our simulation environment when using 500 ms time bins. The average gain even becomes even smaller for 1000 ms time bins as indicated in Figure 3.13.

Our results show that on average, significant decreases in packet loss can be obtained. It is nevertheless important to determine the worst case: is it possible that the default random update shows in some case(s) lower packet loss than



(a) Packet loss decrease vs. default process using 100 ms time bins



(b) Packet loss decrease vs. default process using 500 ms time bins

Figure 3.12: Average packet loss decrease vs. default routing table update process at different traffic aggregation levels

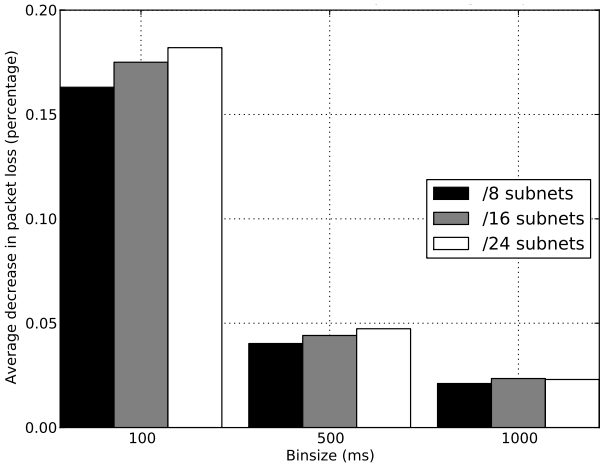


Figure 3.13: Average packet loss decrease vs. default process over all aggregation levels

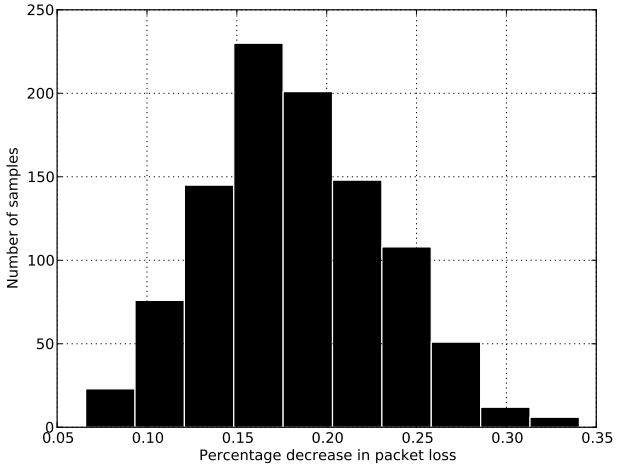


Figure 3.14: Distribution of decrease in packet loss (/24 subnet prefixes and bin size 100 ms)

the combination of prediction and the optimization heuristic. This question is answered by the results obtained in Figure 3.14. This figure illustrates that a normal distribution was obtained around the mentioned averages in packet loss, with minimal and maximal improvements of respectively 5 and 35 percent in decrease of packet loss compared to the default routing table update. We found no executions where the default behavior was better than the suggestion solution. Similar distributions were found at other temporal and spatial aggregation levels for which the average decrease of packet loss is as depicted in Figure 3.13. The results illustrate that using a prediction model in combination with the described heuristic makes most sense at low aggregation levels, where up to 18 percent (average) decrease in packet loss can be achieved.

3.7.4 Recovery time evaluation

The packet loss reduction heuristics from Section 3.6 influence the resulting global recovery time of all affected traffic flows (the duration of the entire routing table update). This is because packet loss decreases are achieved by reordering RIB/FIB entries and by online adapting update-distribution batch sizes. The latter results into more or less batches and swapping events, affecting the resulting global recovery time. In general, smaller resulting packet losses do not necessarily imply that the total recovery time will also be decreased. Depending on the specific traffic trace, splitting batches can be beneficial with respect to the resulting packet loss (lower splitting cost vs. extension cost, see Section 3.6.2.2). More batches lead to more swapping times, resulting into larger recovery times.

Figure 3.15 shows the obtained average global recovery times for the different combinations of traffic models and packet loss reduction heuristics. The figure depicts the average recovery times using modeled traffic at bin sizes of 100 ms (left) and 500 ms (right) at three spatial aggregation levels (/24, /16 and /8 subnets). Strategies 1 and 3 to 6 use fixed batch sizes, and thus result into the same fixed recovery time (only the first is depicted). If traffic is aggregated into /8 subnets, there are not as many aggregated subnet prefixes (max 256). Therefore, the resulting average recovery time is of a lower order than the ones using /16 or /24 subnets. This figure further illustrates that higher traffic aggregation levels typically result into recovery times that are closer to the ones obtained by fixed batch strategies because there remains less opportunity to minimize packet loss by reordering RIB/FIB entries or dynamically resizing batches (see previous section). From Figure 3.15a one might observe that recovery times of packet loss reduction strategies on average result into shorter recovery times ranging from 30 percent faster, to similar recovery times as update strategies using fixed batch sizes. Figure 3.15b illustrates that using packet loss reduction heuristics with less fine-grained traffic models (i.e. using 500 ms bin sizes) further reduce the difference with fixed

batch update strategies, resulting into average recovery times between 10 percent less, to similar recovery times as fixed batch size strategies.

3.8 Computational cost of the procedure

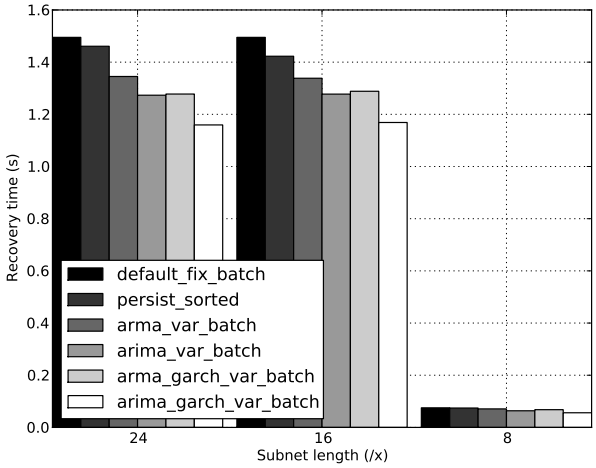
The execution of the proposed procedure can be classified as relatively intensive from a computational perspective. However, many parts can run offline or in background. Fitting ARMA and ARIMA models to time series samples have a computational complexity which is bounded by $O(m^3T)$, where T is the length of the time series sample, and $m = max(p, q + 1)$. Fitting GARCH models reduces to hard non-linear optimization problems. To the knowledge of the authors there is no clear bound to the computational complexity of fitting these models for arbitrary (r, s) values.

The computation of the described optimization heuristics involves operations which are bounded by *O(n log n)* during every step split/extension-step, caused by the need for real-time sorting. While this results in a heavy procedure, it can be drastically reduced by using hardware-implemented sorting algorithms which reduce to constant time sorting for a given number of values [33].

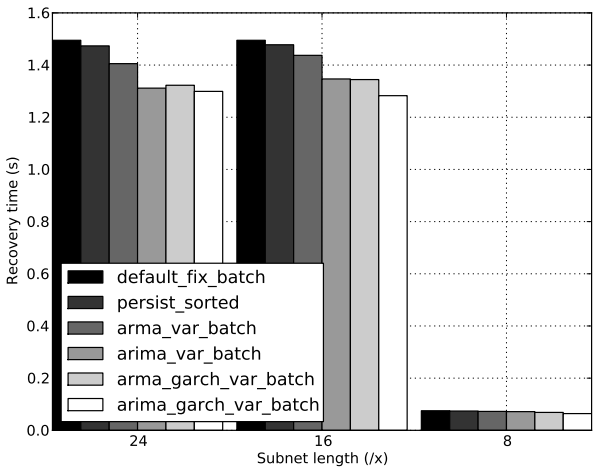
3.9 Conclusion

Link-state routing protocols used within current routing domains can have re-convergence times in the order of seconds when topology changes occur. During these periods of network recovery, affected network traffic is lost as long as their corresponding routing entries are not updated. We showed that first updating routing entries corresponding to high bit rate network traffic is not straightforward because network traffic can highly fluctuate during these periods of re-convergence. In this paper, we proposed a local router mechanism to reduce packet loss during re-convergence periods. This mechanism works independently of other routers and can be applied upon local failure detection as well as upon reception of topology change messages (link-state updates) received from non-adjacent routers.

Our approach used AR(I)MA-GARCH network traffic models to capture sub-second traffic fluctuations significantly better than a persistence traffic model assuming stable traffic during re-convergence. As a result, we showed that we can reduce a clear amount of packet loss resulting from rerouting events, depending on the used traffic model, the used heuristic and the aggregation level at which the traffic is modeled. We obtained packet loss reduction results varying from 18 percent on /24 subnetworks using 100 ms time bins to 2 percent using 1000 ms time bins. Figure 3.14 provided evidence that in some cases even larger gains could be obtained. The application of the used techniques also showed that the result-



(a) Average recovery time vs. default process using 100 ms time bins



(b) Average recovery time vs. default process using 500 ms time bins

Figure 3.15: Average recovery time of dynamic vs. fixed batching strategies

ing recovery times over all affected routing entries are not higher than when using standard router update mechanisms.

Acknowledgment

This work is supported by the European Commission (EC) Seventh Framework Programme (FP7) ECODE project (Grant nr. 223936).

References

[1] P. Francois, C. Filsfils, J. Evans, and O. Bonaventure. *Achieving sub-second IGP convergence in large IP networks*. ACM SIGCOMM Computer Communication Review, 35(3):33–44, July 2005.

[2] D. Katz and D. Ward. *Bidirectional Forwarding Detection*. Internet-Draft draft-ietf-bfd-base-08, Internet Engineering Task Force, March 2008. Work in progress.

[3] M. Shand. *IP Fast Reroute Framework*. Internet-Draft draft-ietf-rtgwg-ipfrr-framework-08, Internet Engineering Task Force, February 2008. Work in progress.

[4] W. Tavernier, D. Papadimitriou, D. Colle, M. Pickavet, and P. Demeester. *Optimizing the IP router update process with traffic-driven updates*. In DRCN 2009, Washington D.C., 2009.

[5] J. Moy. *OSPF Version 2*. RFC 2328, Internet Engineering Task Force, April 1998.

[6] J. M. McQuillan, I. Richer, and E. C. Rosen. *An overview of the new routing algorithm for the ARPANET*. SIGCOMM Comput. Commun. Rev., 25(1):54–60, 1995. doi:http://doi.acm.org/10.1145/205447.205453.

[7] P. Narváez, K.-Y. Siu, and H.-Y. Tzeng. *New dynamic algorithms for shortest path tree computation*. IEEE/ACM Trans. Netw., 8(6):734–746, 2000. doi:http://dx.doi.org/10.1109/90.893870.

[8] *IS-IS Support for Priority-Driven IP Prefix RIB Installation*. http://www.cisco.com/en/US/docs/ios/12_0s/feature/guide/fslocrib.html.

[9] *Alcatel 7750 SR OS router configuration guide*. http://www.juniper.net/techpubs/software/erx/junose82/swconfig-ip-services/html/ip-jflow-stats-config5.html.

[10] *Statistics Service Commands on Cisco IOS-XR Software*. http://www.cisco.com/en/US/docs/ios_xr_sw/iosxr_r2.0/system_management/command/reference/3yrstats.html.

[11] *Interface Statistics on JUNOS 9.2*. http://www.juniper.net/techpubs/software/junos/junos91/swcmdref-basics-services/monitor-interface.html.

[12] *Passive Measurement and Analysis by National Laboratory for Applied Network Research Project (NLANR)*. http://www.nlanr.net/PMA/.

[13] C. Estan, K. Keys, D. Moore, and G. Varghese. *Building a better NetFlow*. SIGCOMM Comput. Commun. Rev., 34(4):245–256, 2004. doi:http://doi.acm.org/10.1145/1030194.1015495.

[14] K. Cho, K. Mitsuya, and A. Kato. *Traffic data repository at the WIDE project*. In ATEC '00: Proceedings of the annual conference on USENIX Annual Technical Conference, pages 51–51, Berkeley, CA, USA, 2000. USENIX Association.

[15] J. Wallerich, H. Dreger, A. Feldmann, B. Krishnamurthy, and W. Willinger. *A methodology for studying persistency aspects of internet flows*. SIGCOMM Comput. Commun. Rev., 35(2):23–36, 2005. doi:http://doi.acm.org/10.1145/1064413.1064417.

[16] M. S. Taqqu, W. Willinger, and R. Sherman. *Proof of a fundamental result in self-similar traffic modeling*. SIGCOMM Comput. Commun. Rev., 27(2):5–23, 1997. doi:http://doi.acm.org/10.1145/263876.263879.

[17] W. E. Leland, W. Willinger, M. S. Taqqu, and D. V. Wilson. *On the self-similar nature of Ethernet traffic*. SIGCOMM Comput. Commun. Rev., 25(1):202–213, 1995. doi:http://doi.acm.org/10.1145/205447.205464.

[18] V. Paxson and S. Floyd. *Wide area traffic: the failure of Poisson modeling*. Networking, IEEE/ACM Transactions on, 3(3):226–244, Jun 1995. doi:10.1109/90.392383.

[19] M. E. Crovella and A. Bestavros. *Self-similarity in World Wide Web traffic: evidence and possible causes*. IEEE/ACM Trans. Netw., 5(6):835–846, 1997. doi:http://dx.doi.org/10.1109/90.650143.

[20] P. Abry, R. Baraniuk, P. Flandrin, R. Riedi, and D. Veitch. *Multiscale nature of network traffic*. Signal Processing Magazine, IEEE, 19(3):28–46, 2002.

[21] N. Sadek, A. Khotanzad, and T. Chen. *ATM dynamic bandwidth allocation using F-ARIMA prediction model*. In Computer Communications and Networks, 2003. ICCCN 2003. Proceedings. The 12th International Conference on, pages 359–363, Oct. 2003. doi:10.1109/ICCCN.2003.1284194.

[22] I. Marian, V. Dadarlat, and B. Iancu. *A comparative study of the statistical methods suitable for network traffic estimation*. In ICCOM: Proceedings of the 13th WSEAS international conference on Communications, pages 99–104, Stevens Point, Wisconsin, USA, 2009. World Scientific and Engineering Academy and Society (WSEAS).

[23] C. Di, F. Hai-Hang, L. Qing-jia, and C. Chun-xiao. *Multi-scale Internet Traffic Prediction Using Wavelet Neural Network Combined Model*. In Communications and Networking in China, 2006. ChinaCom '06. First International Conference on, pages 1–5, Oct. 2006. doi:10.1109/CHINACOM.2006.344786.

[24] A. Sang and S. qi Li. *A predictability analysis of network traffic*. In INFOCOM 2000. Nineteenth Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings. IEEE, volume 1, pages 342–351 vol.1, 2000. doi:10.1109/INFCOM.2000.832204.

[25] Y. Qiao, J. Skicewicz, and P. Dinda. *An Empirical Study of the Multiscale Predictability of Network Traffic*. In HPDC '04: Proceedings of the 13th IEEE International Symposium on High Performance Distributed Computing, pages 66–76, Washington, DC, USA, 2004. IEEE Computer Society. doi:http://dx.doi.org/10.1109/HPDC.2004.3.

[26] S. Basu, A. Mukherjee, and S. Klivansky. *Time series models for internet traffic*. In INFOCOM '96. Fifteenth Annual Joint Conference of the IEEE Computer Societies. Networking the Next Generation. Proceedings IEEE, volume 2, pages 611 –620 vol.2, 24-28 1996. doi:10.1109/INFCOM.1996.493355.

[27] R. Riedi, M. Crouse, V. Ribeiro, and R. Baraniuk. *A multifractal wavelet model with application to network traffic*. Information Theory, IEEE Transactions on, 45(3):992 –1018, apr 1999. doi:10.1109/18.761337.

[28] B. Zhou, D. He, and Z. Sun. *Network Traffic Modeling and Prediction with ARIMA/GARCH*, 2005.

[29] T. Bollerslev. *Generalized autoregressive conditional heteroskedasticity*. Journal of Econometrics, 31(3):307–327, April 1986.

[30] R Development Core Team. *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria, 2009. ISBN 3-900051-07-0.

[31] R. J. Hyndman and Y. Khandakar. *Automatic time series forecasting: the forecast package for R*. Monash Econometrics Working Papers 6/07, Monash University, Department of Econometrics, June 2007.

[32] T. Bollerslev and J. Wooldridge. *Quasi-maximum likelihood estimation and inference in dynamic models with time-varying covariances*. Econometric Reviews, 11(2):143–172, 1992.

[33] Y.-H. Tseng and J.-L. Wu. *On a constant-time, low-complexity winner-take-all neural network*. Computers, IEEE Transactions on, 44(4):601 –604, apr 1995. doi:10.1109/12.376175.

4

Fast failure detection in multipoint networks

In this chapter we design and evaluate a failure detection mechanism which can be used for triggering the use of alternate forwarding entries as configured by IP-FastReroute (IPFRR) schemes. The designed scheme is able to efficiently detect failures in Ethernet multipoint networks. Fast failure discovery and fast switch-over as supported by IPFRR allow faster recovery than the IGP-driven recovery as discussed in the previous chapter. The evaluated failure detection can also be used in combination with the self-configuring approach of the next chapter.

⋆⋆⋆

W. Tavernier, D. Papadimitriou, Puype B., D. Colle, M. Pickavet, and P. Demeester.

Published in Lecture Notes in Computer Science (LNCS), Springer, 2009.

Abstract Bridged Ethernet and IP routing and forwarding are without any doubt the most deployed network technologies, mainly because of their robust, low-cost and easy-to-deploy-character. Solutions to provide fast recovery over these technologies are thus a highly desired feature. Fast recovery not only needs a fast switchover (e.g. using IP-FastReRoute), but also requires fast failure detection. However, few efficient failure detection mechanisms are available over Ethernet

networks, i.e. multi-access broadcast capable networks. We present an efficient multipoint Bidirectional Forwarding Detection (BFD) scheme running directly over Ethernet. We prove its value through emulation and evaluate its time performance and resource consumption.

4.1 Introduction

The growth of the Internet and networks in general, has been significantly supported by the two crown jewels of the networking economy: IP and Ethernet. The first is the layer 3 technology used for inter-network addressing, routing and forwarding, the second is the layer 2 technology of choice for intra-network forwarding, especially in LAN environments. The success of both technologies was emphasized last decades by the numerous efforts to extend them in terms of traffic engineering capacity, resilience possibilities and scalability. This resulted into highly advanced (but also more expensive) technologies relying on constraint-based routing to provide traffic engineering capabilities including extended protection and recovery features, tunneling and aggregation functionality. A well-known example of this category of complex control technologies is Generalized Multi-Protocol Label Switching (GMPLS). Currently, GMPLS-extensions are being developed to allow the setup and teardown of Ethernet-data paths, referred to as Carrier Ethernet (e.g. Ethernet Label Switching [1] and Provider Backbone Bridge-Traffic Engineering (PBB-TE) [2]).

One way to avoid the high expenses and complexity of control suites such as GMPLS, and still provide highly resilient networks using IP over Ethernet, is to use a technique such as IP Fast ReRoute (IPFRR). IPFRR defines a re-routing mechanism at the IP layer which provides protection against a link or router failure by invoking locally pre-determined loop-free alternate paths (a.k.a. repair paths). An important aspect of such a mechanism is the prevention of packet loss caused by transient loops which can occur during the re-convergence of the network following a failure. IPFRR thus provides a highly desirable mechanism to minimize packet loss during failure events, especially for real-time services such as Voice-over-IP (VoIP) or IPTV. However, IPFRR needs to be triggered by a failure detection mechanism (or a failure notification mechanism). In a landscape dominated by switched Ethernet positioned as networking layer device interconnection layer 2 technology, practice learns that almost no fast failure detection mechanism can be used to efficiently trigger re-routing mechanisms such as IPFRR. Failure notification mechanism which are inherently slower than failure detection mechanisms, are therefore no further investigated in the context of this paper.

This paper proposes a new failure detection mechanism applicable at the Ethernet level. The objective consists in enhancing the interplay between the failure detection mechanisms for multi-access broadcast capable Ethernet networks (or

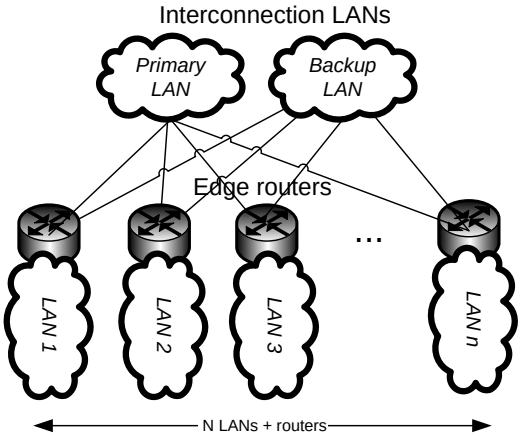
LAN[1]-segments) interconnecting IP routers and IPFRR. The structure of the paper is as follows: in Section 4.2 we discuss existing failure detection methods for use over Ethernet networks on the given scenario of dual-homed LANs and characterize their shortcomings. In the next Section (Section 4.3) we present a new multipoint BFD (MP BFD) variant for Ethernet. The last Section (Section 4.4) describes the implementation of the failure detection mechanism presented in Section 4.3 in an emulation environment, as well as its performance regarding time and resources. The paper concludes with Section 4.5, by summarizing our main results and outlining possible future direction of investigations.

4.2 Failure detection over Ethernet networks

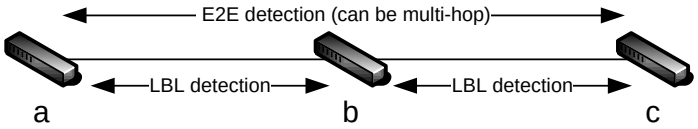
This paper focuses on failure detection mechanisms, i.e. the mechanisms involved in the detection phase that occurs after e.g. a link or node failure, but before the actual actions initiated by the recovery mechanism. In the set of phases involved in the global failure recovery process, failure detection positions thus as turning point between failure occurrence and the recovery operation itself by acting as a trigger. Let us consider the scenario depicted in Figure 4.1a: n Ethernet LANs attached to n edge IP routers connected to a primary and a backup interconnection LAN. Edge routers thus communicate to each other via either the primary or the backup Ethernet LAN. In the ideal situation, from the moment interconnectivity between two edge routers fails via the primary LAN, the backup LAN is used as fast as possible for communication. IPFRR (see [4] and [5]) can be used as an effective switchover mechanism at the IP layer to recover from a detected failure in the primary LAN. However this requires a failure detection mechanism between the edge routers over the primary and/or backup LAN, to trigger the IPFRR mechanism.

Before existing failure detection technologies are introduced, let us define the *scope of the detection*. We distinguish two scopes: detection on a *link-by-link (LBL)* basis and detection on an *end-to-end* (E2E) data path basis (see Figure 4.1b). The first single hop scheme allows a link to be monitored independently of the data paths crossing that link. From Figure 4.1, (Ethernet) data paths are defined by the set of nodes crossed by the Ethernet flow in between a pair of edge routers within the Ethernet interconnection LANs. As such, the single task of LBL detection is to signal whether a link is up, or down. Whereas this provides a perfect mean to determine if a link, and what link exactly has failed in a network, it does not directly indicate the data paths or traffic streams over the network that are affected by the failure. To tackle the latter, the failure detection mechanism needs to adapt

[1] In this paper the term Ethernet LAN (Local Area Network) refers to a network consisting of one or more Ethernet segments (point-to-point or multipoint) interconnected via Ethernet bridges, using the multipoint mechanisms of broadcasting, flooding, learning and learned forwarding (see reference [3] for the standard specification).



(a) Base scenario



(b) LBL vs. E2E failure detection

Figure 4.1: Failure detection over Ethernet

its scope to a specific end-to-end data path within the network. This allows to indicate if a traffic stream following the monitored path from a given source to a given destination, is affected by a failure. This type of detection, referred to as *E2E failure detection*, does not give any information on the exact cause (what link or node) of a failing end-to-end data path.

In the remainder of this section, we subdivide our analysis of existing failure detection mechanisms into i) routing protocol dependent, and ii) routing protocol independent failure detection mechanisms. The former includes Open Shortest Path First (OSPF) Hello messages, whereas the latter includes Ethernet data link layer detection mechanisms and Bidirectional Forwarding Detection (BFD).

4.2.1 Routing protocol related mechanisms

4.2.1.1 Open-Shortest Path First protocol (OSPF)

is a link-state routing protocol which maintains neighbor relationships (e.g. between the edge routers of Figure 4.1) by periodically sending OSPF Hello messages at every HelloInterval (Type 1 packets, see [6]). Routing adjacencies are

	Min Hello Interval	Default Hello Interval	Targeted detection time	Scope	Layer	Maturity
Routing protocol dependent						
OSPF	1s	10s	seconds	LBL	IP (L3)	Stable, mature
OSPF+fast hello	1ms	NA	tens to hundreds of ms	LBL	IP (L3)	Ongoing
RSTP	1s	2s	seconds	LBL	Ethernet (L2)	Stable, mature
Routing protocol independent						
Hardware detection	<1ms	NA	milliseconds	LBL	PHY (L1)	Stable, mature
IEEE 802.1ag CFM	1ms	NA	tens to hundreds of ms	LBL+E2E	Ethernet (L2)	Ongoing
BFD	1ms	NA	tens to hundreds of ms	LBL+E2E	Independent	Stable, mature

Table 4.1: Failure detection overview

formed between selected neighboring routers for exchanging routing information. The RouterDeadInterval is used to determine the number of seconds before the router's neighbors will declare it Down when they stop hearing the router's Hello Packets (by default, 4 times the HelloInterval). This event typically triggers the reconvergence of the routing protocol for the new topology. By default, Hello timers can be configured in the order of seconds (default value of 10s), with a minimum of 1 second, resulting in a failure detection time ranging from 2 to 4 seconds. Such detection time is too slow to allow seamless recovery of real-time services as outlined in the Introduction. We validated these numbers by means of emulation (see Section 4.4). Initiatives have thus been undertaken to increase the frequency of transmitting Hello packets in OSPF (see e.g. [7]). However these approaches never found broad acceptance, because OSPF was not initially designed for performing fast link failure detection (but only detecting and maintaining neighbor relationships). Indeed, decreasing the Hello timers would result into a high risk of overloading the routing process because of the frequent sending and receiving of Hello messages (which would be from 10 to 1000 times smaller than initially planned). In addition, the lack of handshake mechanism to agree on the HelloInterval to be used between neighboring routers would result into manual configuration of the HelloInterval, which is undesirable in larger networks.

4.2.1.2 (Rapid) Spanning Tree Protocol

The (R)STP protocol is a distance-vector routing protocol used in (VLAN-)bridged Ethernet to i) construct a loop-free logical tree topology originated at the root bridge (the tree spans all bridges of the entire Ethernet broadcast domain or subdomain), and ii) avoid broadcasted frames of being endlessly looped through the network. STP, specified in IEEE 802.1d, is based on a break-before-make paradigm (hence slow by construction). Subsequent attempts such as RSTP, specified in IEEE 802.1w, consider a make-before-break approach with faster convergence

time. The RSTP failure detection mechanism is also based on Hello timers (as in OSPF). The Hello interval can be configured between 1 and 10 seconds, having a default value of 2 seconds. A link is considered Down after a given number of subsequent Hello messages is lost (3 by default, see [3]). The failure detection mechanism of RSTP is still too slow given the Hello-interval granularity in terms of seconds for the recovery of real-time services. More generally, none of these additional mechanisms/extensions fundamentally solve the root cause of slow convergence resulting from the count-to-infinity problem intrinsic to distance-vector protocols. In practice, this means that only if the remaining topology after failure does not include any cycle - and if the root bridge is not affected by the failure - the re-convergence time is acceptable. If manufacturers declare quick(er) convergence times, this is because of combination with hardware-detection (see next section), and because they do not take into account worst-case topologies prone to i) count-to-infinity, or ii) topologies with slow convergence properties such as linear/ring topologies.

4.2.2 Routing protocol independent mechanisms

4.2.2.1 Hardware detection

The most obvious example of routing protocol independent failure detection is performed at the PHY level (layer 1). Various PHY technologies support the detection of the loss of signal at link level. The Ethernet PHY allows to detect decreases in voltage between two links. While this provides for a sound and fast method for failure detection (in the order of tens of microseconds), a major issue is that it can only detect link-errors, meaning that remote link failures are by definition undetectable in case the signal is terminated before reaching the destination.

4.2.2.2 Ethernet Connectivity Fault Management (CFM)

The ongoing IEEE 802.1ag standard defines a set of OAM mechanisms to help operators in managing their bridged Ethernet environments. The CFM suite comprises loopback, traceroute, and continuity check functions. In particular, the last function is interesting for failure detection in our research space. The idea consists in supporting a multicasted heartbeat (short-length protocol data unit) along the network, that is configurable at millisecond granularity. Unfortunately, the details of IEEE 802.1ag protocols, are not yet standardized, nor implemented.

4.2.2.3 Bidirectional Forwarding Detection (BFD)

BFD, specified by the Internet Engineering Task Force (IETF, see [8]), positions itself as a protocol to detect faults in the bidirectional path between two forwarding

engines, including interfaces, data link(s), and to the extent possible the forwarding engines themselves, with potentially very low latency (tens to hundreds of ms). In its *asynchronous mode* (base mode), BFD defines a Hello-protocol which sends control packets between two end points at a negotiated transmission (TX) rate and reception (RX) rate. Both rates range in order of microseconds. A three-way handshake mechanism sets up BFD sessions. Sessions are declared Down if control packets at one of either side arrive beyond a computed level of negotiated transmission rates (using a multiplication factor with the remote TX). The *demand mode* allows a system, once a BFD session is established, to ask the other system to stop sending BFD control packets, except when the system decides to explicitly verify connectivity. In this case a short sequence of BFD Control packets is exchanged, and then the system acknowledges the reception of this sequence of packets. An additional *Echo function* provides an independent detection mode by sending control packets to the remote system that itself loops them back through its forwarding path towards the sender. BFD is being standardized for IP and MPLS-environments ([9]). BFD can be used over IP by encapsulating BFD control packets into UDP. When running over connectionless forwarding environments such as IP, BFD supports link-by-link sessions (LBL or single hop, [10]), and end-to-end sessions (E2E or multi-hop [11]).

Now, let us reconsider a part of the network from Figure 4.1a: a set of n LAN's having their edge routers connected to a (primary) Ethernet LAN. If we want to apply failure detection between the edge routers using BFD over IP as currently specified, we can set up $n(n-1)/2$ LBL IP BFD sessions[2], i.e. one session per pair of routers. Though this scheme is valid, from the resource consumption perspective this mode of operation is not efficient, as it results into $(n-1)$ BFD sessions running over every interface connecting an edge router to the LAN (N-square scalability problem). Given the multi-access broadcast capable nature of the interconnection network (Ethernet), this results in a waste of resources, because (1) the outgoing direction of the interface connecting a router to the multi-access network does not take advantage of the fact that the (multi-access) LAN network itself can replicate its BFD messages, and (2) every router needs 2 timers per other edge router: one for scheduling the BFD message to be sent, and one to check if a BFD message has been received sufficiently recent, while only the last one is needed. In addition the Address Resolution Protocol (ARP) is needed to retrieve the MAC addresses to be used for setting up the BFD sessions over IP. The observation that the BFD definition from [8] does not account for the special nature of broadcast and multicast capabilities of certain multi-access network technologies, is also acknowledged by the multipoint BFD effort at IETF (see [12]). By relying on the multicast nature of the underlying technology (again targeting IP or MPLS), a uni-

[2]no matter the underlying Ethernet LAN topology, at the IP layer nodes are only at single hop distance from each other, resulting into the applicability of LBL BFD sessions

directional multipoint BFD session can be set up from a head-end node to a set of tail-end nodes enclosed by a multicast address. Sending BFD Control packets at regular intervals, as determined by the TX value in the BFD Control packet header, allows tail-end nodes to detect loss of connectivity with the head-end node. Three possibilities to notify the head-end node are discussed in [12] in case of loss of multicast connectivity: (1) (unreliable) asynchronous BFD feedback of the tails in case of failure, (2) the head periodically triggers the tails to reply (poll-sequence) with session status, and (3) request a poll-sequence but in case of failure initiate a unicast BFD session-setup (on-demand unicast BFD session). Whereas these multipoint BFD extensions constitute definitely a step forward, they still inherit on the characteristics of the IP layer (or the MPLS layer) and do not account for the specific nature of an Ethernet network. For the given network depicted in Figure 4.1a, this makes the detection mechanism traffic dependent: point-to-point BFD sessions (for unicast traffic) or multipoint BFD sessions (for multicast traffic). Thus point-to-point BFD sessions are still required for detecting liveliness of links over which unicast traffic is transmitted towards its corresponding destination address. In particular, on a multi-access link/network, BFD Control packets are transmitted with the source and destination addresses set as part of the corresponding IP subnet.

Therefore, we suggest to extend the BFD specification such as to take benefit of the properties specific to Ethernet technology and keep BFD mechanism independent of the traffic exchanged between the two end-points of the session. It is important to underline that the proposed mechanism is not transposable at the IP level because multipoint BFD sessions can only ”detect” failures for multicast traffic forwarding, i.e. upon failure, no conclusion can be drawn for what it concerns unicast traffic.

4.3 BFD over Ethernet

Multi-access broadcast capable environments such as Ethernet LAN segments inherently allow for reaching any port (identified by their MAC address) within the network. Instead of encapsulating the BFD packets into UDP messages over IP datagrams, we suggest to encapsulate BFD packets directly into an Ethernet MAC frame. In this scheme, the end points of the BFD sessions are directly identified by the MAC addresses of the corresponding interfaces[3]. As depicted in Figure 4.1a, whereas the edge routers are at multi-hop distance in the Ethernet layer. If we apply BFD failure detection at the Ethernet layer between the given edge routers at the Ethernet layer, we have two choices: (1) using E2E BFD sessions over Ethernet, and (2) using Multipoint BFD (MP BFD) sessions over Ethernet. The first

[3]Source and destination addresses are only needed during BFD session setup, running sessions use the discriminator field (see [8])

option results again into the N-square scalability problem as described in Section 4.2.2.3. Indeed, except the addressing specifics, LBL BFD sessions over IP are analogue to E2E BFD sessions over Ethernet. The second option is further detailed in Section 4.3.1.

4.3.1 Bi-directional failure using Multipoint BFD over Ethernet

True failure detection in an Ethernet LAN network as illustrated Figure 4.1a, needs bidirectional failure detection capability between all nodes, taking advantage of the multi-access property of the medium, independently of the properties of the traffic being sent on the network. Setting up a single multipoint BFD (MP BFD) session per node at the Ethernet layer is therefore sufficient. The scheme works as follows: (1) Each node sets up a single MP BFD session for which control packets have as source MAC address the local outgoing interfaces address and use a well-known group MAC address as destination MAC address, (2) Each node listens to the group MAC address on which they periodically receive MP BFD control packets from $(n-1)$ sources. For this purpose, a well-known group MAC address is dedicated to the BFD protocol such that "edge routers" can listen from senders. The actual group MAC address is taken from the set of reserved codepoints for MAC group addresses as specified by IEEE standards[4]. Setting up the MP BFD session can be performed either manually for every node, or triggered for all nodes once a first MP BFD session has been set up (the group MAC address from the first session can then be re-used in order to trigger the other MP BFD sessions). The MP BFD session reaches the Up-status, when all outgoing MP BFD Control packets declare the session to be Up. Every node connected to the Ethernet network receives thus $(n-1)$ MP BFD Control Packets having an Up-status. In case one of these nodes does not receive any BFD Control Packet from one (or more) sources after the calculated Detection Time, it declares the MP BFD session Down. This means that the declaring node has lost connectivity with the source of the MP BFD session. The Detection time is defined as the period of time without receiving BFD packets. As for BFD for asynchronous mode, the Detection Time is calculated as follows: at the receiving (local) node, this time is equal to the value of DetectMult received from the remote node, multiplied by the agreed transmit interval of the remote node[5]. The DetectMult value determines the (minimum) number of missing packets in a row before declaring the session Down. Figure 4.2 illustrates the scenario where 3 edge routers are interconnected by a single switch[6]. Each edge router connected to the multi-access broadcast

[4] http://standards.ieee.org/regauth/groupmac/tutorial.html
[5] the greater of bfd.RequiredMinRxInterval and the last received Desired Min TX Interval (see [8])
[6] $n = 3$ and replace the primary LAN by a single switch from Figure 4.1

capable network maintains thus one transmitting MP BFD head session and keeps listening tail sessions to $(n-1) = 2$ other MP BFD head sessions.

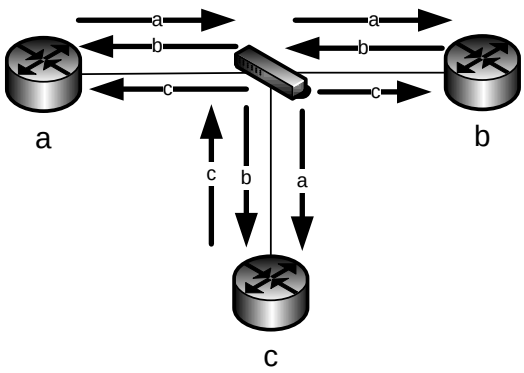


Figure 4.2: Asynchronous mode in MP BFD

On the contrary to existing mechanisms (see Section 4.2), the formulated Ethernet-specific MP BFD scheme allows a given receiver to detect unreachability of the sender, once the receiver does not hear from one (or a set of) sources, i.e. it does not receive BFD control packets during a pre-determined period. Upon detection (and declaration of the corresponding session down), the receiver assuming bidirectionality in the Ethernet link layer, declares the logical link to that address Down and can trigger re-routing at the IP layer (e.g. IPFRR). Applied on the network from Figure 4.1a, re-routing can imply to route over the backup LAN instead of routing over the primary LAN (or vice versa).

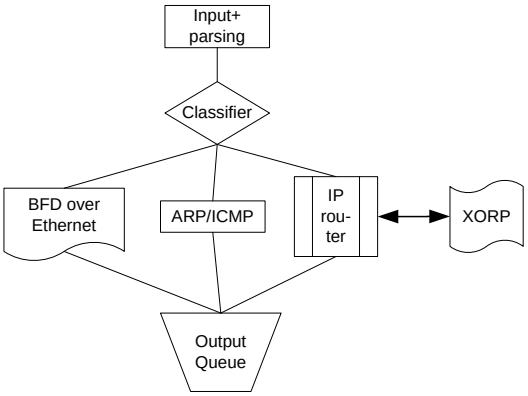
4.4 Performance in emulation

In order to evaluate the performance of the proposed MP BFD protocol scheme over Ethernet, we implemented it in an emulation environment. This experimental implementation allows us to benchmark the added value of the proposed mechanism in terms of time and resource consumption efficiency.

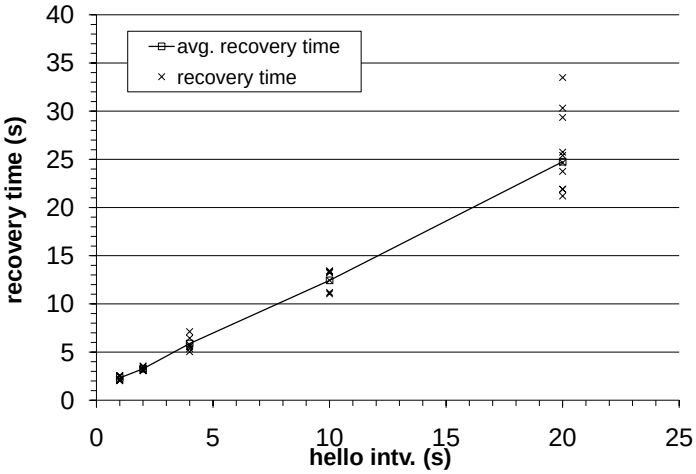
4.4.1 Architecture and environment

The implementation of the MP BFD protocol over Ethernet is designed as an element in the Click Modular Router ([13]) framework. The Click software architecture allows for building flexible and configurable routers. A Click router is assembled from packet processing modules called elements that can be linked to each other into a directed graph representing the router configuration. Because

Click does not comprise an OSPF-element as part of its suite of components, we interconnected the Click router tables with the OSPF daemon from XORP ([14]). As a result, an edge router from Figure 4.1 can be emulated by a Linux PC, running both Click and XORP. A simplified node architecture of such router is shown in Figure 4.3a. The resulting emulation network set up within the Emulab environment ([15]) runs on the virtual wall of ilab.t at IBBT (where each node is emulated on a dual core CPU of 2.0GHz).



(a) Node architecture in emulation



(b) OSPF reconvergence time

Figure 4.3: Emulation

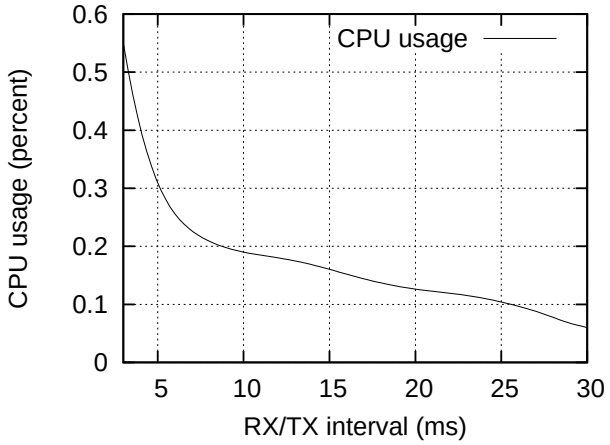
4.4.2 BFD over Ethernet

BFD failure detection was integrated into a new element of the Click Modular Router software. As described in Section 4.3, the implementation does not rely on UDP/IP for encapsulation of the BFD control packet. Instead, the BFD control packet is directly encapsulated into an Ethernet frame using a custom (configurable) Ethertype[7]. It is important to underline once again that the encapsulation scheme is not an implementation specific decision but plays a critical role in the protocol architecture. Indeed, running MP BFD over UDP/IP would limit its applicability outside of the targeted scope of this paper. The element supports the following BFD over Ethernet modes: Link-By-Link (LBL) BFD sessions, End-to-End (E2E) BFD sessions, E2E BFD sessions over Carrier Ethernet tunnels (see [1]), and MP BFD sessions (4.3.1). The Click handler interface triggers the setup of BFD sessions. The setup can also be triggered using a socket interface, such that new BFD sessions can be triggered from external applications (such as Dragon GMPLS [1]). With the latter it is possible to send a feedback signal using the same socket interface. The arguments needed to set up a BFD session are: (1) the MAC address of the local interface (source node) and the remote interface (destination node), the latter can be set to a group address for multipoint BFD as explained in Section 4.3.1), and (2) the local interface identifier that leads to the remote interface (e.g. eth1).

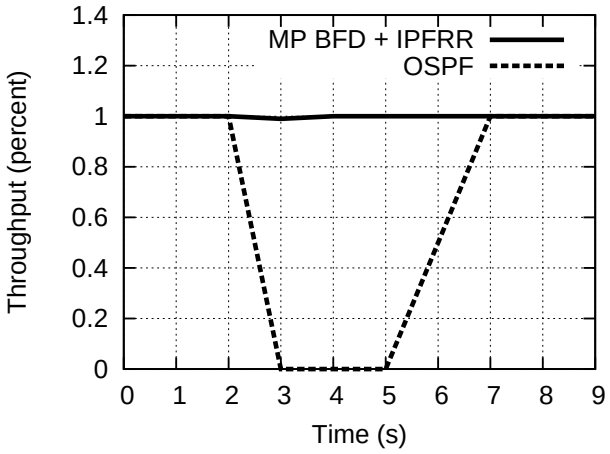
To test the CPU performance of our implementation, we set up a single hop BFD session and hold it for 30 min using equal transmission (TX) and receiver (RX) timer value. Figure 4.4a shows the average CPU usage compared to the configured RX and TX values. Even with an experimental implementation, the trend shows an exponential decay for decreasing RX and TX values.

In order to determine the lowest possible detection times (time performance) that are possible using our implementation, the following experiment was set up on a ring topology of 19 nodes. E2E BFD sessions over Ethernet were set up with increasing hop-sizes, varying from 1 to 18 hops. Using equal values for the transmission (TX) and receiving (RX) timers, we started setting both values at both ends at 200 ms, using a multiplier equal to 3 (see Section 4.2.2.3). Next we waited for 15 minutes. If the session did not time out for a single time, we considered the session to be stable, then we decreased both timers in steps of 5 ms. Meanwhile traffic streams varying from 100 Mbps to 1Gbps were sent through the Click nodes. Using this methodology, the lowest, stable RX and TX values that we found compared to the length of the paths they monitored (number of hops) is shown in Figure 4.4c. This figure shows that failure detection times of 15 ms were possible on our emulated network.

[7]The Ethertype field of the Ethernet frame header determines what payload is carried by the Ethernet frame. Standard values are 0x800 for IP or 0x8100 for VLAN-tagged Ethernet frames



(a) Detection time

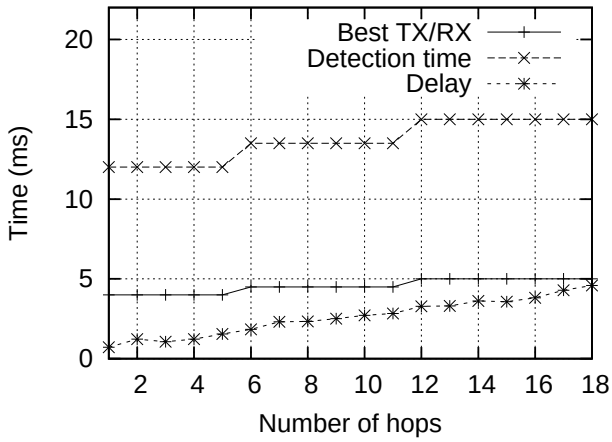


(b) Bandwidth usage

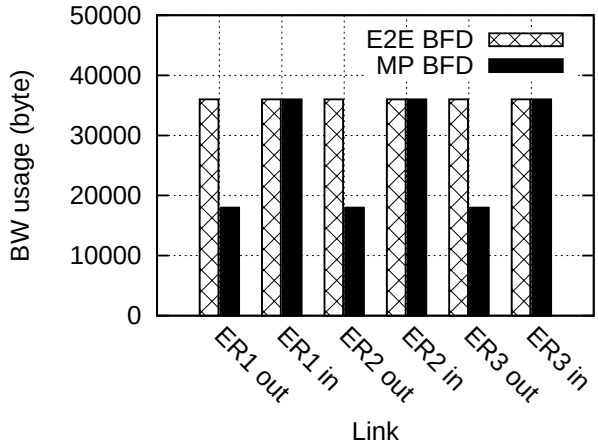
Figure 4.4: Emulation results

4.4.3 XORP OSPF performance

A preparatory experiment consisting of 8 individual runs using XORP and Click on the topology, confirmed the theoretical reconvergence times stated in Section 4.2.1.1. As illustrated in Figure 4.3b we found the fastest reconvergence time of about 2.5 seconds on average by using 1 second for the HelloInterval and the double for the RouterDeadInterval. When the HelloInterval was increased, we found a linear trend in the reconvergence time, as it can be observed from Figure 4.3b.



(c)



(d)

Figure 4.4: Further emulation results

4.4.4 OSPF vs. MP BFD over Ethernet with IPFRR

For the purpose of performance comparison between OSPF Hello vs. MP BFD over Ethernet in combination with IPFRR recovery mechanism, we consider the topology depicted in Figure 4.1a using 3 LANs/edge routers ($n = 3$). The goal of this experiment is to compare throughput performance of standard OSPF-based solution vs. MP BFD over Ethernet in combination with IPFRR when a link fails in the primary LAN. In our setup, the TX and RX timers of the MP BFD sessions are set to 5ms (with multiplier=3) and a failure is introduced at the third second. Figure 4.4b shows the throughput graph (for constant traffic flow of 500 Mbps) of

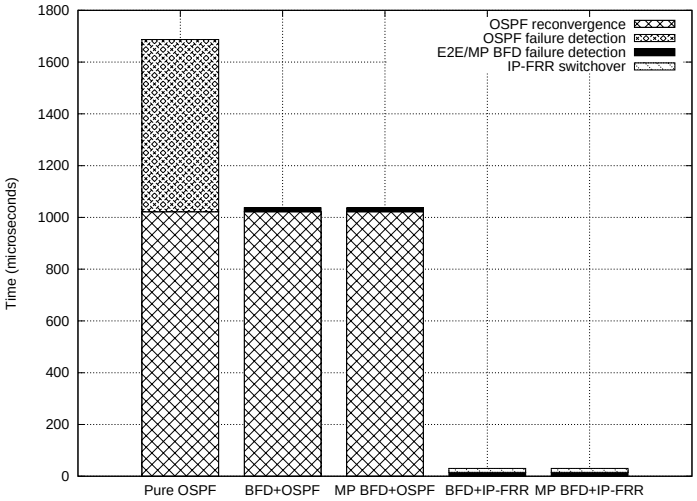


Figure 4.5: Time comparison

a pure OSPF-based solution compared to the approach combining MP BFD with IPFRR. Where less than a percent packet loss happens for the latter during the third second (failure detection time took 15 ms, IPFRR-switchover took another 15 ms), the pure OSPF-based solution resulted in 100 percent packet loss for more than 2.5 seconds (resulting from the reconvergence time). This confirms the expected gain in time performance. Figure 4.5 illustrates the time that is resulting from OSPF failure detection, OSPF reconvergence, (MP) BFD failure detection and IPFRR switchover. Figure 4.4d shows the gain in resources, comparing MP BFD with E2E BFD over Ethernet for the given base network with 3 edge routers (ER's). Figure 4.4d makes clear that MP BFD consumes half of the bandwidth less on the outgoing direction of the link towards the primary LAN. This obtained result is as expected, because for the outgoing direction MP BFD relies on the multiplication functionality of the Ethernet LAN.

In summary, the conducted experiments have proved the gain in terms of resource consumption and time performance of our implementation running (MP) BFD over Ethernet. This closes the gap of a highly needed feature of IP routers interconnected by LAN networking needing fast recovery.

4.4.5 Scalability analysis of MP BFD

Here below, we analyze the scalability of the MP BFD detection mechanism. Given our base scenario of Figure 4.1a, investigating the scaling of the proposed mechanism involves two main dimensions:

- The *size (of the primary and backup LAN) topology*, which influences: i) the

ability to recover within the LAN using RSTP-based mechanisms and ii) the values to be used for the RX and TX timers for the running BFD-sessions between the edge routers. Because neither failure detection, nor recovery using RSTP is envisioned (for reasons specified in Section 2.1) MP BFD does not suffer from the scaling effects of the LAN-topology with respect its influence on RSTP. However larger LAN-topologies do have an effect on the RX and TX-timers that can be used in the MP/E2E BFD configuration, as these timers need to be higher than the total delay (sum of propagation, transmission, forwarding, and scheduling delay) from sender to receiver.

- *Interconnecting more edge routers* (increasing n) to both the primary and the backup LAN, which scales as follows (independently of the underlying LAN topology): i) Linearly increasing set of transmitting MP BFD sessions (using the LANs broadcasting properties): 1 transmitting MP BFD session per edge router, ii) Squared increase with respect to the number of listening MP BFD sessions: every edge router listens to all other edge routers transmitting MP BFD frames, i.e. n(n-1) sessions.

With respect to the transmitting side, the solution has better scaling properties than any existing failure detection mechanism between n network nodes. The number of edge routers does not affect the timers' values that can be used for setting up the MP BFD sessions because their settings mainly depend on the diameter of the network topology. Indeed, independently of the number of edge routers, from the moment a failure is detected at the receiving nodes, the MP BFD session is declared Down by that node. This detection time is only determined by the DetectMult multiplied by the TX-time whose setting depends on the diameter of the LAN topology. This diameter determines the total delay incurred by the BFD packets transmitted from the sending node and the receiving nodes. The LAN's inherent broadcasting capability ensures that only one BFD frame needs to be transmitted from the detecting node while guaranteeing that the frame will be multiplied on branching points within the LAN network.

4.5 Conclusion and future work

This paper has shown the shortcomings of existing mechanisms for detecting failures between IP routers interconnected by Ethernet LANs. Therefore, a new multipoint BFD-variant over Ethernet has been proposed and evaluated by means of emulation through Click and XORP. The presented MP BFD scheme: (1) acknowledges the special character of the underlying data plane, by running directly over Ethernet, (2) allows faster failure detection than the classical OSPF Hello mechanism, and (3) uses less resources compared to E2E BFD sessions. The defined mechanism enables a faster trigger of recovery switching from a primary to a

backup Ethernet LAN in case a failure is detected in the former. Nevertheless, it does not take into account the possibility that only connectivity between a subset of the edge routers needs to be recovered, as it declares the entire MP BFD session Down and triggers the switchover for all edge routers. Future work could consist in detecting and determining which edge routers are affected by the failure by correlating the errors, such that only a partial switchover is needed. Another aspect that requires further investigation is the verification of the scaling performance that would allow confronting the analysis detailed in Section 4.4 against a large-scale execution of the proposed MP BFD scheme with IPFRR.

4.6 Acknowledgments

This research is partly funded by the Institute for the Promotion of Innovation through Science and Technology in Flanders (IWT-Vlaanderen) through the TIGER project in the European CELTIC framework and the FP7 project BONE.

References

[1] W. Tavernier, D. Papadimitriou, D. Colle, M. Pickavet, and P. Demeester. *Emulation of GMPLS-controlled Ethernet Label Switching*. In International Conference on Testbeds and Research Infrastructures for the Development of Networks and Communities (TridentCom), Washington, USA, 2009.

[2] A. Reid, P. Willis, I. Hawkins, and C. Bilton. *Carrier ethernet*. Communications Magazine, IEEE, 46(9):96–103, September 2008. doi:10.1109/MCOM.2008.4623713.

[3] IEEE. *Draft Standard for Local and Metropolitan Area Networks: Media Access Control (MAC) Bridges (Revision of IEEE Std 802.1D -1998 incorporating IEEE Std 802.1t -2001 IEEE Std 802.1w -2001) (Replaced by 802.1D-2004)*. IEEE Std P802.1D/D4, pages –, 2003.

[4] M. Shand. *IP Fast Reroute Framework*. RFC 5714, Internet Engineering Task Force, January 2010. Available from: http://tools.ietf.org/html/rfc5714.

[5] A. Atlas and A. Zinin. *Basic Specification for IP Fast Reroute: Loop-Free Alternates*. RFC 5286, Internet Engineering Task Force, September 2008. Available from: http://www.rfc-editor.org/rfc/rfc5286.txt.

[6] J. Moy. *OSPF Version 2*. RFC 2328, Internet Engineering Task Force, April 1998. Available from: http://www.rfc-editor.org/rfc/rfc2328.txt.

[7] Z. Kou. *Update to OSPF Hello procedure*. Internet-Draft draft-kou-ospf-immediately-replying-hello-02, Internet Engineering Task Force, January 2007. Work in progress. Available from: http://www.ietf.org/internet-drafts/draft-kou-ospf-immediately-replying-hello-02.txt.

[8] D. Katz and D. Ward. *Bidirectional Forwarding Detection*. RFC 5880, Internet Engineering Task Force, June 2010. Available from: http://tools.ietf.org/html/rfc5880.

[9] R. Aggarwal, K. Kompella, T. Nadeau, and G. Swallow. *BFD For MPLS LSPs*. RFC 5884, Internet Engineering Task Force, June 2010. Available from: http://tools.ietf.org/html/rfc5884.

[10] D. Katz and D. Ward. *BFD for IPv4 and IPv6 (Single Hop)*. RFC 5881, Internet Engineering Task Force, June 2010. Available from: http://tools.ietf.org/html/rfc5881.

[11] D. Katz and D. Ward. *BFD for Multihop Paths*. RFC 5883, Internet Engineering Task Force, June 2010. Available from: http://tools.ietf.org/html/rfc5883.

[12] D. Katz and D. Ward. *BFD for Multipoint Networks*. Internet-Draft draft-katz-ward-bfd-multipoint-02, Internet Engineering Task Force, February 2009. Work in progress. Available from: http://www.ietf.org/internet-drafts/draft-katz-ward-bfd-multipoint-02.txt.

[13] E. Kohler, R. Morris, B. Chen, J. Jannotti, and M. F. Kaashoek. *The click modular router*. ACM Trans. Comput. Syst., 18(3):263–297, 2000. doi:http://doi.acm.org/10.1145/354871.354874.

[14] M. Handley, O. Hodson, and E. Kohler. *XORP: an open platform for network research*. SIGCOMM Comput. Commun. Rev., 33(1):53–57, 2003. doi:http://doi.acm.org/10.1145/774763.774771.

[15] B. White, J. Lepreau, L. Stoller, R. Ricci, S. Guruprasad, M. Newbold, M. Hibler, C. Barb, and A. Joglekar. *An Integrated Experimental Environment for Distributed Systems and Networks*. In Proc. of the Fifth Symposium on Operating Systems Design and Implementation, pages 255–270, Boston, MA, December 2002. USENIX Association.

5

Self-configuring Loop-Free Alternates with High Link Failure Coverage

In this chapter, we extend the concept of Loop-Free Alternates (LFA) enabling fast recovery in connectionless IP networks. Our scheme enables fully automated configuration of the alternate forwarding entries, as well as covering almost 100 percent of potential link failures.

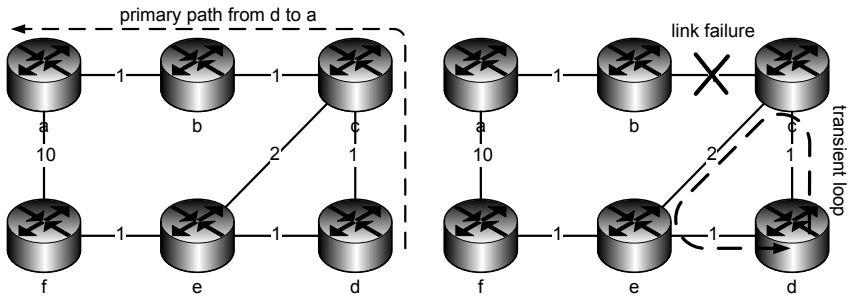
⋆⋆⋆

W. Tavernier, D. Papadimitriou, D. Colle, M. Pickavet, and P. Demeester.

To be published in Telecommunication Systems, Springer, 2012

Abstract A change in network topology triggers the re-convergence process of routing protocols. The re-convergence time of current routing protocols (e.g. OSPF) is constrained by the possibility of having transient loops due to the independent calculation of shortest paths between routers affected by a network failure. Several IP Fast-ReRoute (IPFRR) schemes have been developed to proactively calculate and install alternate forwarding entries almost instantaneously once a topology update message is received, without causing temporary microloops The IPFRR scheme which has been used most extensively so far makes use of Loop-Free Alternates (LFA). While these are easy to configure, LFAs still require manual configuration, and the resulting ratio of covered link failures is only

about 60 to 70 percent. This paper presents a logical extension of the Loop-Free Alternate concept, proposes a self-configuring scheme to populate the corresponding alternate entries, and evaluates the performance of the scheme with respect to coverage, configuration time and path length in a simulation environment.



(a) (shortest) path without link failure (b) transient loop upon link failure

Figure 5.1: Path of a packet entering at router d and exiting at router a in different network conditions

5.1 Introduction

The Internet has evolved from a pure data-communication network towards a communication network that transports various types of real-time network traffic such as VoIP and real-time video. As a consequence, the resulting availability requirements of the network have severely increased. Carrier-grade network recovery in today's communication networks prescribes recovery times of maximally 50 ms to avoid severe Quality of Service (QoS) decreases in the services carried over the network. Unfortunately the above requirement cannot be met by the original routing infrastructure of the Internet, consisting of Interior Gateway Protocols (IGP) to ensure that routing paths can be calculated within Autonomous Systems[1] (AS) such as OSPF [1], and Exterior Gateway Protocols (EGP) such as BGP [2] to allow routing between different ASes.

In this paper we will further focus on the use of IGP link-state routing protocols such as OSPF, however the application of the proposed scheme can be extended to EGP-context. Several studies (e.g. [3]) have indicated that IGP protocols can require between hundreds of milliseconds to even seconds to ensure full reconvergence of the routing tables. As will be detailed in the next section, this is a result of independently updating routing and forwarding tables between differ-

[1] a collection of connected Internet Protocol (IP) routing prefixes under the control of one or more network operators who presents a common, and expectedly consistent routing policy to the Internet

ent nodes in the network. As a result, the IP-FastReroute (IPFRR) framework was developed by the Internet Engineering Task Force (IETF). IPFRR enables faster recovery times by pre-calculating routes which can be locally activated, immediately after failure detection. The activated alternate routes by design avoid synchronization problems between the update of forwarding tables in different network nodes. However, while Loop-Free Alternates are currently the most popular IPFRR technique, due to its simplicity, it is unable to provide high network failure coverage, as well as having automated population of the alternate entries. Therefore, we propose a fast-rerouting technique extending the concept of LFAs to increase the failure coverage of the technique, and provide a self-configuration scheme to automatically detect and configure the resulting alternate paths.

The paper is organized as follows. First, the problem which our approach is targeting to solve is detailed in Section 5.2. Next, we describe existing work in Section 5.3 in the context of the described problem, and position our solution in the resulting space. In Section 5.4 we present the different aspects of our scheme in detail, and Section 5.5 details the time-related aspects of our self-configuring approach. Next, the proposed solution is benchmarked in Section 5.6, which reports on the experimental results we have obtained by simulation when running these procedures on topologies representative of core networks. Finally, Section 5.7 formulates the conclusions of the paper.

5.2 The problem of transient loops

Routers use local Forwarding Information Bases (FIB or forwarding table) that stores forwarding entries each comprising the outgoing interface to be followed by individual incoming datagrams for a given destination prefix. FIB entries are derived from the routes calculated using topology information distributed through routing protocols such as Open Shortest Path First (OSPF, [1]). These protocols discover the local topology (link states) and distribute the discovered information over the network using Link State Update messages.

Every router in a connection-less IP network, independently executes the above process. Exchanges of link state routing information resulting from topology changes (dynamic reaction to topological changes due to, e.g., link/node failures) lead to the re-computation of the routing paths and reconfiguration of the corresponding FIB entries (re-convergence). However, as every router performs the routing path computation independently of other routers, transient (micro-)loops may be formed during the periods when a network is re-converging due to inconsistent FIB entries. This problem is inherent to any asynchronous distributed routing protocol and caused by inconsistent FIB entries resulting from the propagation time of the routing updates as well as the time needed to re-compute and distribute FIB entries. Packets which are trapped into transient loops, never reach

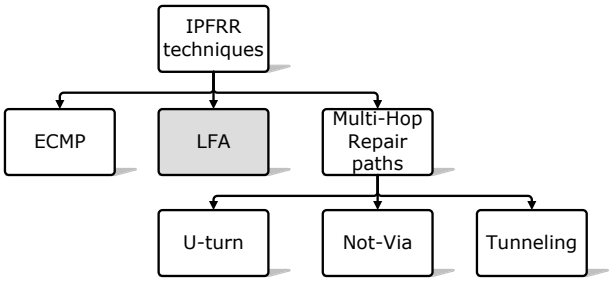


Figure 5.2: Overview of related work

their destination and are simply lost after TTL expiration.

This situation is illustrated in Figure 5.1. The figure depicts a network topology of six routers, interconnected by links with a given cost. When all links are fully operational, all datagrams entering the network via router d having router a as exit point, follow the shortest path (d, c, b, a). If we consider the case in which the link between b and c fails, router c almost directly re-calculates its routing and forwarding tables, and determines router e as next hop for datagrams towards router a. A micro-loop can now occur because router d and router e have not yet updated their forwarding tables due to the timed needed to propagate the link failure notification from router c towards d and e. The resulting transient loop is as follows: packets entering the network at router d are forwarded to router c, which sends them to router e using the updated forwarding table, while router e sends them back to d. For every next hop, the TTL counter in the IP header will be decreased. Once it reaches zero, packets will be dropped, resulting in packet loss. This situation only disappears when both router d and e have updated their forwarding tables accordingly.

5.3 Related work

The IPFRR framework proposed by IETF reduces the re-convergence time in connection-less IP networks upon link failures. IPFRR relies on: i) pre-calculating alternate forwarding entries unable to result into transient micro-loops, and ii) activating the pre-calculated entries upon link failure detection. As such, the resulting re-routing time is minimal: the time needed for each node, after the occurrence of a topological change, to use up-to-date FIB entries -without relying on the full IGP re-convergence- along loop-free alternate paths for the maximum number of destinations.

IPFRR techniques are applicable to a network employing conventional connection-less IP routing and forwarding. These should not be confused with MPLS Fast-Reroute (MPLS-FRR) schemes allowing local protection in connection-oriented

networks. Connection-oriented networks such as MPLS enable end-to-end connectivity by explicitly setting up and reserving forwarding entries following a predetermined path. This requires protocols such as RSVP(-TE) [4, 5] to set up connections in the network. This category of techniques has been extensively studied in [6, 7].

IP Fast-ReRoute (or fast repair) techniques can be classified into the following categories (see [8]), as illustrated in Figure 5.2.

5.3.1 Equal cost multi-path routing (ECMP)

Equal-Cost Multi-Path routing (ECMP [9]) is a routing strategy where next-hop packet forwarding to a single destination can occur over two or more (shortest) paths towards the same destination. In that case, packets are distributed according to some rule (e.g. hash function) over the available next hops corresponding to the determined paths. ECMP routing can be used in conjunction with most routing protocols, since it is an independent per-hop decision that is limited to a single router. Its main purpose is related to the substantial increases in bandwidth it can offer, by ”load-balancing” traffic over multiple paths. As a side-effect, it provides robustness to the routing. From the moment a failure affects one of the found equal-cost paths, the alternate paths can still be used as repair paths. The main drawback of ECMP results from its static behavior. Robust load-balancing techniques relying on this principle have been proposed in [10].

5.3.2 Loop-free alternate (LFA) paths

A Loop-Free Alternate path [11] exists when a direct neighbor of the router adjacent to the failure has a path to the destination that can be guaranteed not to traverse the failure (loop-free neighbor condition). Applied to the example given in Figure 5.1, node c can pre-configure an LFA towards destination a via router x, if

$$d(x, a) < d(x, c) + d(c, a).$$

While no router x can be found in Figure 5.1, slightly modifying the topology as in Figure 5.3 allows router g to be used as LFA, because sending a datagram to router g won't induce the datagram to be sent back. This example illustrates that not all topologies can provide LFA protection for every possible link failure. The average coverage on common networks (that is strongly dependent on the topology) shows variations from 60 to 90 percent [12–14]. Indeed, when a link or a node fails, only the neighbors of the failure are initially aware that the failure has occurred and only neighboring node to the failure repair the failure. These repairing routers have to steer datagrams to their destinations although most other routers in the network are unaware of the nature and the location of the failure. A common limitation in most

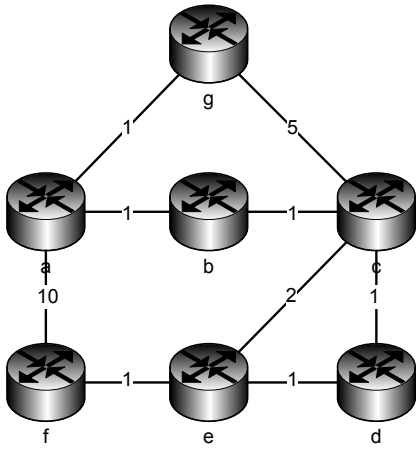


Figure 5.3: Modified topology allowing LFA from router c to router a upon link failure of b-c

of the base LFA mechanism is its inability to indicate the identity of the failure and to explicitly steer the repaired datagram around the failure. Consequently, the extent to which this limitation affects the repair coverage is topology dependent.

An advanced LFA solution [15] consists in sequencing the FIB updates either spatially (topologically ordered FIB update from far-end to the near-end neighbor contiguous to the failure) or temporally (timely synchronized FIB updates). For instance, ordered FIB update provides 100 percent loop-free convergence at the expense of a FIB update time proportional to $R.MAX_{FIB}$, where, R is the max (hop) length among paths to edge r used to reach destination t (downstream SPF neighbor prior to the failure) and MAX_{FIB} is a network-wide constant that reflects the maximum time T_{max} required to update a FIB irrespective of the change required. Hence, degrades proportionally to the path length, i.e., FIB updates are actually committed at the near-end after reception of a completion message traveling back from the source of max (hop) length among path to edge r used to reach destination t. This solution is not considered outside network maintenance operation as it suffers from slow activation.

Whereas the alternate approaches in Section 5.3.3 try to change the underlying forwarding model to improve coverage, an alternate approach consists of adding links to the network, such as to improve the resulting coverage against link failures. This type of study has been reported in [16].

5.3.3 Multi-hop repair paths

When there is no feasible loop-free alternate path, it may still be possible to locate a router, which is more than one hop away from the router next to the failure, from which traffic will be forwarded to the destination without traversing the failure. Multi-hop repair paths are more complex both in the computations required to determine their existence, and in the mechanisms required to invoke them. Multi-hop repair paths techniques can be further classified as:

- Mechanisms where one or more alternate FIBs are pre-computed in all routers, and the repaired datagram is instructed to be forwarded using a "repair FIB" by means of a per-datagram signaling method involving, e.g., the detection of a "U-turn" [17].
- Mechanisms functionally equivalent to a loose source route that is invoked using the normal FIB. These include tunneling-based approaches [18] that consist in "by-passing" the topology change by pre-configuring a tunnel whose path is not affected that change. There are multiple variants of "tunnel-based solutions": single-sided (near-end or far-end), double-sided (near-end and far-end), and distributed (tunnel segments). They all suffer from the same problems: i) computational complexity, ii) tunnel pre-configuration and maintenance, and iii) impact on forwarding plane. Thus, they all involve a high degree of configuration for tunnels that in turn decrease the forwarder performance. Other mechanisms such as the Not-Via technique [19] employ special addresses that are installed in the FIBs together with pre-computed routes that avoid certain components of the network. This technique encapsulates the datagram to an address that explicitly identifies the network component that the repair path must avoid. This produces a mechanism that always achieves a repair, provided the network is not partitioned by the failure.

5.3.4 Objectives for improving Loop-Free Alternates schemes

The above section gave an overview on existing fast path repair/fast re-routing techniques for connection-less IP networks. In practice, Loop-Free Alternates has proved to be the most adopted approach in existing operational networks albeit not capable of providing high failure coverage in all networks. Contrary to the approach described in [16], this paper won't propose techniques to extend the network to improve coverage, but will propose an extension of the concept of Loop-Free Alternates (LFA).

Our contribution is threefold: i) the proposed technique relies on distributed learning of the loop-domain at each node and "best-alternate path" to a given destination. Both can either be performed on-line or by mining the link-state routing

topology and the routing table (RT) entries; ii) the proposed re-routing scheme does not assume modification of the link-state routing protocol operations outside of the transient re-routing periods (as alternate forwarding entries take local precedence over default IGP routing entries). Once, the IGP has re-converged unflagging datagrams leads to the use of the primary path entries; iii) the coverage of the proposed re-routing scheme is almost 100 percent. Relying on the described properties, our approach aims to:

- Maximize the percentage of links (or nodes) that can be fully protected (i.e., for all destinations)
- Maximize the percentage of destinations that can be protected for all link (or node)
- Minimize the stretch increase on the routing paths between source and destination.

5.4 Our approach: self-configuring LFA/LFNs

This section proposes a self-configuring fast-rerouting technique relying on (and extending) the concept of Loop-Free Alternates. The proposed approach consists of three phases:

1. **Loop-Free Node (LFN) detection**: during this phase detection of nodes is performed, which are not in the loop domain of their originating nodes with respect to probed destinations. The loop domain is determined by the set of nodes for which the loop-free neighbor criterion is not verified along certain alternate routing paths before occurrence of topological change (when traffic forwarded by node u and directed to destination t arrives at node v that forwards this traffic along a path that reaches node u, i.e., v is a not loop-free neighbor of u).
2. **Loop-Free path configuration**: the detecting node selects an alternate routing path that ensures loop-freeness up to loop domain boundaries by instantiating an alternate forwarding entry on each intermediate node (pointing to the loop-free neighbor).
3. **Loop-Free path activation**: upon failure occurrence, the node triggers that loop-free alternate path (when traffic from node u directed to t arrives at node v, the latter does not forward traffic along a path that reaches node u, i.e., v becomes a loop-free neighbor of u).

After introducing the assumptions and preliminaries of our approach, all of these phases will be detailed in the next sub-sections.

5.4.1 Definitions and assumptions

The network topology is modeled as a weighted undirected graph $G = (V, E, \omega)$ with positive edge cost $omega$, where V is the set of vertices or nodes ($|V| = m$) and E is the set of edges or links ($|E| = n$). A non-negative cost function $\omega : E \rightarrow Z^+$ associates a cost u, v to each link $(u, v) \in E$. For $s, t \in V$, let $d(s, t)$ denote the cost of the path $p(s, t)$ from s to t in G, where the cost of a path is defined as the sum of the costs along its edges. We first introduce the following distinction:

- For the pair $s, t \in V, s \neq t$, if there exists a vertex u adjacent to vertex s, (i.e., edge $(s, u) \in E(G)$) such that $d(u, t) < d(s, u) + d(s, t)$, i.e., u is a loop-free neighbor of s to destination t, then the path $(v_0(= s), v_1, \ldots, v_m(= t))$ is a *loop free alternate path* where $\forall i : d(v_i, v_m) < d(v_{i-1}, v_i) + d(v_{i-1}, v_m)$.

- For the pair $s, t \in V, s \neq t$, if there exists a vertex u adjacent to vertex s, (i.e., edge $(s, u) \in E(G)$) such that $d(u, t) < d(s, t)$, i.e., u is a downstream neighbor of s to t, then the path $(v_0(= s), v1, \ldots, v_m(= t))$ is a *distance decreasing downstream path*, where $\forall i : d(v_i, v_m) < d(v_{i-1}, v_m)$. As a particular case, neighbor u of node s is the downstream SPF neighbor of s for destination t, if node u provides the shortest path to t according to a shortest-path first (SPF) routing scheme.

Note that the set of distance decreasing downstream paths is a subset of the set of loop-free alternate paths meeting the condition $\forall i : d(v_i, v_m) < d(v_{i-1}, v_m)$. We define the *loop domain* of node $u \in V(G)$ as the set of node $B(u)$ such that if a path $p(s, \ldots, u, \ldots, w, \ldots, t)$ traverses node u and then node w it will loop back via node u before reaching destination t, i.e., w does not sit along a loop-free alternate path to destination t from node u.

5.4.1.1 Network assumptions

The proposed approach aims to accelerate the re-routing of traffic along loop-free alternate routing paths in link state routing networks. Upon failure occurrence, the failure detection technique is assumed to provide local information. Failure information propagation does not rely on associated fast failure notification protocol (operating next to the link-state IGP) or IGP parameter tuning. The only condition for our approach to be operational is that the loop domain's diameter is smaller than the flooding domain of the IGP. Otherwise, the technique resumes as a best exit-node selection to avoid loops inside the IGP routing domain but then relies on neighboring domains for the alternate path to re-merge with the primary path (outside the loop domain).

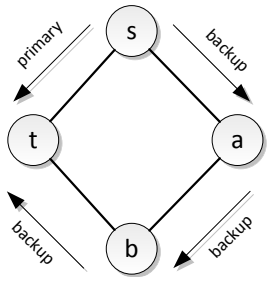


Figure 5.4: Interface-specific forwarding

5.4.1.2 FIB structure assumptions

A router consists of a Routing Information Base (RIB) and a Forwarding Information Base (FIB). In the context of this paper, the terms RIB and routing table are used equivalently since we assume that a single routing protocol is running in each routing domain. The FIB stores forwarding entries each comprising the outgoing interface to be taken by individual datagrams for a given destination prefix. The router model we use in this paper, allows to store as part of the FIB, an alternate forwarding entry for any given destination prefix. The use of the alternate entry is triggered by the indication of a flag (bit) in the header of an incoming datagram, further referred to as the alternate flag. Datagrams are marked with this flag, once a failure is detected on the link towards the next hop according to the primary forwarding entry.

In our router model, the forwarding decision is also conditioned on the incoming interface, which implies that the alternate entry for a given destination prefix can be different for datagrams arriving at interface x, compared to those arriving at interface y of a given router. This interface dependence allows us to keep using shortest path routing on the primary forwarding entries. To ensure that the alternate forwarding entries have node-wide significance, the identifier of the triggering node (that is the node that flags the datagram) should be known and stored at configuration time as part of the alternate entries and be included as part of the flagged datagram. This is illustrated in Figure 5.4 below. The shortest path towards node t from node s and b is via their direct link. On the other hand, if needed to use node-wide significant alternate routing entries, this would force node t to choose whether node s or node b is on the primary path (for datagrams arriving from any interface on s).

If the primary next hop of node u along its primary path to a given destination becomes unreachable due to a link or node failure, then i) the datagrams for that destination are flagged (as indicated before) and ii) the alternate forwarding entry for the interface corresponding to the failing link or node is chosen to forward the flagged datagrams along the alternate path. At node w, the use of the

alternate forwarding entry must not result into flagged datagrams being sent back to node u (rule.1). Along the alternate path, flagged datagrams arriving from primary interface (i.e., the interface corresponding to the next hop as indicated in the primary forwarding entry) or more generally any interface if the identifier of the triggering node can be retrieved from the incoming datagram, the alternate flag will automatically trigger the use of the alternate forwarding entry to avoid looping behavior (rule.2). To avoid that the flagged datagrams loop back to node u, the proposed technique comprises a cycle-free alternate path computation technique. This technique is described in the next section.

Initial FIB configuration We initiate the primary FIB (pFIB) of all nodes using the usual shortest-path computation techniques for (connection-less) link-state routing protocols such as OSPF or IS-IS. The alternate FIB (aFIB) stored at each node is initially a copy of the pFIB, using the same next hop for on all interfaces as the one determined by the shortest path calculation for the pFIB. With one noticeable exception: the aFIB entry corresponding to the primary forwarding entry is populated with the next hop according to the shortest path excluding the link indicated by the primary forwarding entry. We will refer to this entry as the Alternate Shortest Path entry (ASP entry). Note also that after configuration, the forwarding entries for which the primary and the alternate next-hop for the same destination are identical can be removed from the aFIB. Furthermore, FIB compression techniques (one entry for multiple prefixes) can be used to reduce the memory space used by the aFIB.

5.4.2 Loop-Free Node (LFN) detection

A Loop-Free Alternate (LFA) from a node u towards a node t indicates a next hop which differs from the next hop node of the primary (shortest) path. However, as previously discussed, such an LFA is not always available for a given (s, t)-pair[2]. In the latter case, all potential next hops will return the datagram back to node s. This implies that all neighbors of u are within its loop-domain with respect to t. By definition, all nodes contained in the loop-domain, will return datagrams back to the originating node.

Therefore, the first part of the proposed approach consists of each node u determining its loop domain $B(u)$ with respect to each destination t that it can reach (as indicated by its routing table entries). For this purpose, node u sends a probe message towards destination t on the interface directed to one of its non-shortest path from u to destination t. If the message returns to u (source of the probe message) the message didn't reach a node v located outside of the loop domain. We refer to

[2]remember that both source s and destination nodes t may reside outside the local IGP domain

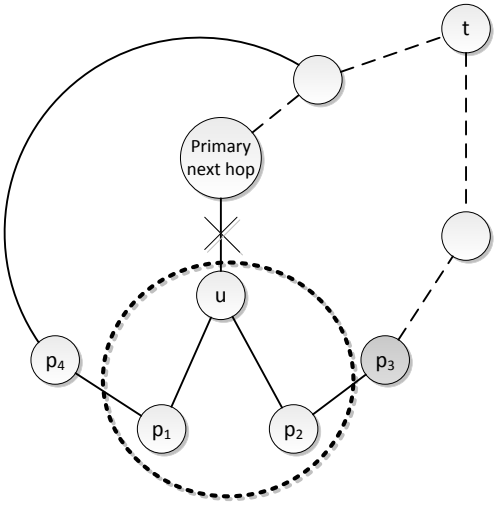


Figure 5.5: The probing process

such a node v as a loop-free node (LFN), and refer to the path $(u, \ldots, v, \ldots, t)$ as the loop-free alternate path (or more synthetically $p(u, v, t)$).

5.4.2.1 LFN using BFS+

As indicated earlier, in order to ensure a loop-free alternate path from a node u towards a destination t, the former needs to find a node (LFN) outside of its loop-domain with respect to t. For this purpose, we devise an extended Breadth First Search method (referred to as BFS+). The recursive mechanism works as follows:

- Send a probe message towards t via all neighbors of node u (hop count diameter 1 from node u).
- If all probe messages are intercepted by node u, mark the visited nodes, and repeat the procedure with the neighbors of the marked nodes (excluding the already visited nodes) until at least one probe message reaches a node v sitting outside the loop-domain of node u. Upon arrival of the probe message at node v, the receiving node v, referred to as the loop-free node (LFN),then sends an acknowledgment message towards node u.
- When multiple LFNs are found within a given diameter from node u, the LFN is chosen such that the alternate routing path towards t has the lowest similarity with the primary path from u to t[3].

[3]The similarity of paths from two nodes towards a third, can be measured by actively storing temporary forwarding states during the probing process, or using traceroute measurements as is performed in [20]

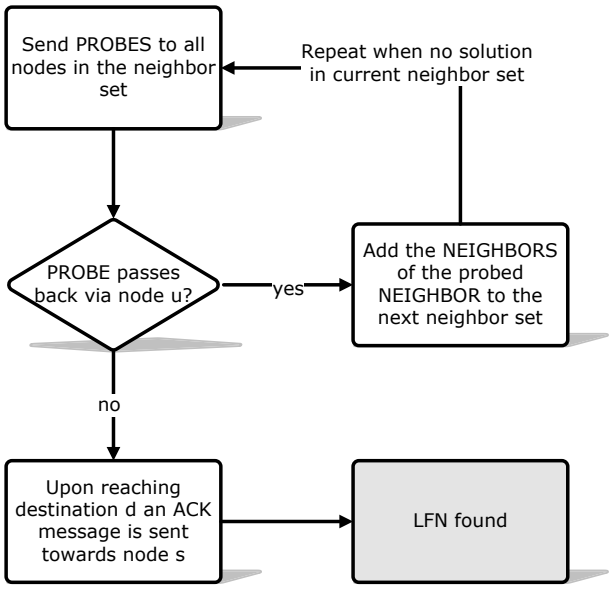


Figure 5.6: LFN search

The BFS+ algorithm is illustrated on Figure 5.5. The loop-domain of node u is indicated with the circle with dotted lines, containing the nodes p_1 and p_2. p_1 and p_2 are upstream nodes to u with respect to destination t. This implies from these nodes, the shortest path towards destination t will always pass via node u. Because these nodes are within hop count diameter 1, BFS+ will first send probe via these nodes towards destination t. When the node u intercepts these probe messages, BFS+ triggers probe messages to be send from hub nodes on hop count diameter 2. Afterwards, the nodes p_3 and p_4 are tested. Both nodes forward the probe message to t without passing node u, and thus are LFNs. However, because the path taken from node p_3 differs more from the primary path compared to the path taken from node p_4 (which uses the same last link towards t), p_3 is elected as the LFN by the procedure.

5.4.2.2 Learning from previous probes

Every node needs to repeat the loop-domain detection procedure for all destination address prefixes it locally stores in its routing table. If the latter procedure is executed independently for all prefixes, this can potentially result in a high number of probes to be initiated by every node.

Nevertheless, the number of probes can be drastically reduced if the knowledge of the loop-domain detection for one destination prefix can be re-used during the loop-domain detection phase for other destination prefixes. The latter is illustrated

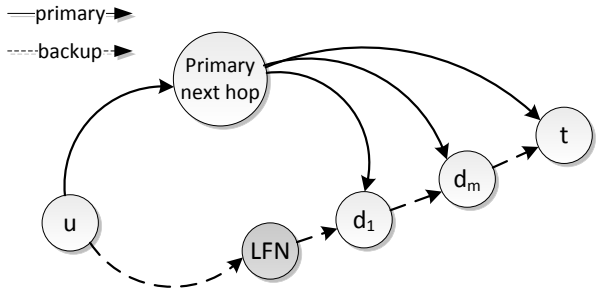


Figure 5.7: Reuse the same LFN for other destinations

in Figure 5.7. This figure illustrates that for a given source and destination node, all nodes between the detected LFN and the destination, can potentially benefit from using the same LFN. If we denote all the intermediate nodes between the LFN and destination t as $d_1, \ldots, d_m$, the same LFN can be used for all those nodes for which the primary next hop in u towards $d_1, \ldots, d_m$ is the same as for t.

5.4.3 Loop-Free Path configuration

Inside the loop-domain $B(u)$ of node u, along the non-shortest path that is selected as the loop-free alternate path and to which the probe message sourced at node u is forwarded, alternate forwarding entries are configured for that destination t. Indeed, the default forwarding entries at these nodes for destination t refer to a primary path that traverses node u. More precisely, $\forall w \in B(u)$ node w does not verify the loop-free condition, the path $p(w,t)$ includes node u, i.e., $p(w,u,t)$. When the probe message reaches node v, that message is returned to node u with the indication that no FIB entry configuration is required to reach destination t (node v verifies the loop-free condition: $d(v,t) < d(v,u) + d(u,t)$. Note that with the BFS+ technique (as documented in Section 5.4.2.1), the loop-free alternate path $p(u,v,t)$ is the non-shortest path that differs the most from the shortest path (considered as the primary path) before failure of a link incident to u along the primary path from u to t, $p(u,t) : v \notin p(u,t)$.

Once an LFN node is discovered and selected using the previously described BFS+ technique, the loop-free alternate path towards the LFN must be configured. This operation is realized by installing the alternate forwarding entries along the alternate routing path from node u (whose source of traffic is node s) to the LFN. For this purpose, the node u sets its forwarding entry towards the LFN as its alternate entry towards destination t. The same procedure is used as the indicated next hop(s) until the LFN is reached.

The ALFA-learning (Automated Learning of Loop-Free Alternates) procedure

executes the above LFN detection and LFN path-configuration processes from all nodes towards all destinations. When a node detects that the outgoing interface corresponding to the primary routing entry for a given destination is not available, based on a loss-of-signal event or a Hello-timer timeout (as in OSPF), the alternate forwarding entry towards the destination is used. This procedure will bring the packet to the LFN (as it was previously configured to do so), and from then on, shortest path routing entries will bring the packet from the LFN to the destination.

5.4.4 Loop-Free Path activation

Upon failure detection by node u (assume, e.g., the failure of one of the links incident to node u along its primary path towards destination t), the loop-free alternate path is activated. The action of activation by node u of its loop-free alternate path $p(u, v, t)$ refers to the triggering operation of the alternate forwarding entry along the loop-free alternate path inside the loop domain of node u, $B(u)$. The alternate forwarding entries are triggered from the reception of datagrams including as indication in their header[4] that these datagrams were re-routed by node u along the loop-free alternate path. Activation of the alternate forwarding entries is performed until reaching node v. Outside of the loop-domain of node u, datagrams remain flagged but without triggering any action at the nodes traversed by these datagrams (the alternate and the primary forwarding entries are indeed identical). This condition is sufficient to guarantee that the path $p(v, t)$ followed by the datagrams leaving the loop-domain is loop-free, i.e., as long as the path $p(v, t)$ is the distance decreasing SPF downstream path to destination t (the path $p(v, t)$ does not re-enter the loop domain of node u). When exiting the local routing domain (i.e., the link state routing protocol flooding domain), the datagrams flagged by the re-routing node u are unflagged by the boundary node of the domain.

Alternate FIB configuration As previously explained, upon occurrence, once a single failure is locally detected by a given router, incoming datagrams directed towards the affected destinations are flagged, and the datagrams are forwarded according to the alternate forwarding entry (the ASP entry as defined here above). However, because downstream routers still forward flagged datagrams according to their locally computed shortest path, it is likely that the flagged datagrams will be looped back to the flag-originating-node (FON), causing a forwarding loop. To avoid forwarding loop situations, we combine two techniques: i) the discovery of a node referred to as the loop-free node (LFN) which sits outside of the loop-domain of a given node with respect to a given destination, and the LFN is out of the loop-domain of the given node with respect to the LFN itself, and ii) the configuration of

[4]This could be a specific flag (one bit is sufficient) in the IP header. Currently such a field has not been standardized yet.

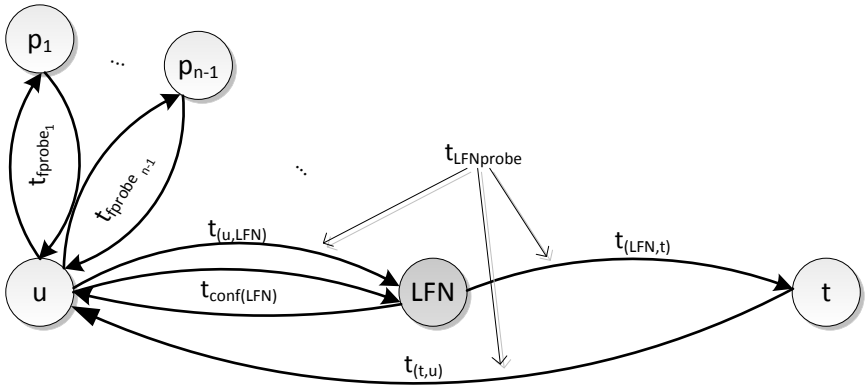


Figure 5.8: Time model of ALFA

the aFIB entries along the path towards the given LFN, this path is the one referred to as the alternate path. The loop-domain of a given node u for a given destination t is defined as the set of downstream nodes (with respect to the directionality of the traffic flow towards destination t) that forward incoming datagrams received from node u along a path that traverses node u. Once flagged, the datagrams reach the LFN, the path followed according to the rest of the aFIBs lead to the destination without looping back to the original node again.

5.5 Time model of the self-configuration process

This section formulates a time model of the ALFA procedure. This model allows to characterize the time needed to find a loop-free node, to configure the corresponding alternate entries, as well as the resulting recovery time. For this purpose, we make the following assumptions:

- all links in the topology have the same characteristics, resulting in equal propagation time T_{prop} (for example $800\mu s$)
- the processing time of a control message by each node is T_{ctrl} (for example, $5\mu s$)
- the time needed to configure an alternate FIB entry is T_{conf} (for example, $10\mu s$)
- T_{gen} denotes the time needed to generate a message (either a probe or an acknowledgment, for example, $10\mu s$)
- in case of link failure occurrence, T_{det} denotes the local failure detection time (in the node adjacent to the failed link, for example, $20ms$)

- the time needed to activate an alternate forwarding entry is T_{act} (for example, $10\mu s$)
- the path length in hop count between node a and node b are reflected as $H(a, b)$

If we take the above into account, the ALFA procedure requires the following time to find and configure a loop-free alternate towards a given destination (referred to as t) from a given node (indicated as u), in case the pure BFS-search is used.

$$T_{alfa(u,t)} = \sum_{i=1}^{n-1} T_{fprobe_i} + T_{LFNprobe} + T_{conf(LFN)}$$

The formula reflects that the time required by the ALFA procedure for a given node pair can be decomposed into i) the time needed to perform $n-1$ probes, each referring to probes that pass via nodes within the loop-domain and returns to the probe initiating node, followed by ii) the time needed to find the LFN node sitting outside of the node u loop domain (remember that the LFN node stores a routing entry for a shortest path to destination t that does verify the loop-free condition), and iii) the time needed to configure the path towards the LFN.

A probe consists of three parts: i) the time to generate of the probe message by node u, T_{gen}, ii) the time needed to propagate the probe message from node u towards the probed node p_i, $T_{(u,p_i)}$, iii) the processing time by node p_i, T_{ctrl}, and iv) the time needed to return the probe message from p_i to the originating node u, $T_{(p_i,u)}$. This results into the following formula:

$$T_{fprobe_i} = T_{gen} + T_{(u,p_i)} + T_{ctrl} + T_{(p_i,u)}$$

Components of the form $T_{(a,b)}$ reflect the time needed to forward a control message (in this case a probe packet) from node a to b along the shortest path. For given propagation and control packet processing times, this can be written out as below.

$$T_{(a,b)} = H(a, b).T_{ctrl} + (H(a, b) + 1).T_{prop}$$

The time needed for probes reaching the LFN node v, consists of five parts: i) the time to generate of the probe message by node u, T_{gen}, ii) the time needed to propagate the probe message from node u towards the probed node v, $T_{(u,v)}$, iii) the processing time by node v, T_{ctrl}, and iv) the time to generate an acknowledgment message T_{gen}, and v) the time needed to return the acknowledgment message from v to the originating node u, $T_{(v,u)}$. This results into the following formula:

$$T_{LFNprobe} = 2T_{gen} + T_{(u,v)} + T_{ctrl} + T_{(v,u)}.$$

Once the LFN node v has been found, the last step of the procedure consists of configuring the alternate entries along the path from the initiating node u to node v, as shown in the formula below.

$$\begin{aligned} T_{conf(v)} &= H(u,v).T_{conf} + (H(u,v)+1).T_{prop} \\ &\quad + T_{gen} + T_{(v,u)} \end{aligned}$$

The resulting formula reflects that: i) each node along the alternate path from node u towards LFN node v needs to perform a FIB configuration action, which requires T_{conf} each, ii) the LFN node v generates an acknowledge packet (to indicate successful alternate path configuration), and iii) the resulting acknowledgment is forwarded back towards the originating node u.

The above time model characterizes the time consumed by the ALFA scheme when a pure BFS LFN-search is performed. Upon the application of BFS+, a small number of additional steps is needed. In this case, the ALFA procedure is not stopped when the first LFN is found but continued until all nodes within the same distance of the originating node u are probed. This will potentially result into an additional number of probes (T_{fprobe}'s) and
($T_{LFNprobe}$'s).

Depending on the use of ALFA as a protection (pro-active configuration of the alternate entries) or a restoration scheme (re-active configuration of the alternate entries), the resulting recovery time when a failure is detected is as follows.

$$\begin{aligned} T_{prot} &= T_{det} + T_{act} \\ T_{rest} &= T_{det} + T_{alfa(u,t)} + t_{act} \end{aligned}$$

5.6 Experimentation

5.6.1 Environment

To evaluate the proposed self-configuring ALFA scheme, a custom-made simulation environment was developed using Python and C++ optimizations. The default routing behavior (primary routing entries) of the benchmarked networks follows shortest path routes as configured by a distributed link-state routing protocol such as OSPF. To obtain representative results, the computed routes were randomized and made independent between the nodes in the network. This implies that, if multiple shortest paths are available between two network nodes, every run will randomly choose a route, and configure the routing tables accordingly. This performed independently on every network node. Every experiment has been re-run

Network	*Nodes*	*Links*	*Degree*		
			MIN	*AVG*	*MAX*
abilene	11	14	2	2.55	3
nobel-us	14	21	2	3.00	4
nobel-ge	17	26	2	3.06	6
garr	22	36	2	3.27	9
nobel-eu	28	41	2	2.93	5
geant2	30	47	2	3.13	8
renater	36	49	2	2.72	7
cost266	37	57	2	3.08	5
germany5	50	88	2	3.52	5
xwin	57	77	2	2.70	6

Table 5.1: Network topologies

100 times with these randomized settings, and the reported quantitative results are averages over these runs.

5.6.2 Network topologies

A set of 10 representative reference networks was used for evaluating the described techniques. Most of these networks are known for research purposes (e.g. [21]), or are research networks themselves. The number of nodes of these networks ranges between 11 and 57 nodes, and their node degree is in the range between 3 and 9. For some of these topologies, single connected nodes have been removed, because alternate routing paths are not possible for these anyhow. The properties of the reference networks are summarized in Table 5.1.

5.6.3 Benchmarked techniques

The techniques with the following labels were benchmarked:

1. **random**: this technique refers to the configuration of usual shortest path routing entries as performed by a link-state protocol such as OSPF, augmented with random alternate entry routing entries (the only requirement is that the alternate next outgoing interface is different from the primary outgoing interface).

2. **learn_backup_ipfrrlf**: this technique refers to the configuration of Loop-Free Alternates (also referred to as FRR-LFA) as backup entries as discussed in Section II.B [11], if they are available. Finding a loop-free alternate entry is performed by probing the paths from nodes' neighbors to check if they loop back towards the originating nodes.

3. **learn_backups_alfa**: alternate forwarding entries are configured using the proposed ALFA technique from Section 5.4, which finds Loop-Free alternates using the BFS+ method. In this case, the LFN is chosen within the diameter of the closest node out of the loop-domain, having a path towards the targeted destination which differs maximally within the probed neighborhood.

5.6.4 Results

This section discusses the performance of the mentioned techniques with respect to their ability to cover link failures (coverage), their consequences on the resulting length of the backup paths (stretch), their communication cost for learning adequate entries, and their sensitivity with respect to network characteristics.

5.6.4.1 Analysis of the required number of probes

Apart from the link failure coverage and other performance measures, which will be discussed in detail later, it is crucial to have an idea for the resulting number of probes which are triggered over the network, to avoid that the network is flooded with probes. For this purpose, we evaluated two variants of probing: i) probing as induced by the BFS+ process (referred by the straight line), as described in Section 5.4.2.1, and ii) reduced probing relying on learned LFNs (referred by the dotted line) as described in Section 5.4.2.2.

Figure 5.9 illustrates that the majority of LFN searches need less than five probes to find an adequate LFN using BFS+. In case learning optimizations are used, the average number of probes decreased by at least 25 percent. While the upper bound on the required number of probes was in most networks not more than 20, the *renater*, *germany50* and *xwin* network had larger spread in the number of probes required (up to 40 probes needed for some LFNs in *xwin*).

5.6.4.2 Analysis of the size of the loop domain

Next to the concept of an LFA, which provides an alternate forwarding entry towards a destination's primary next hop is unavailable due to a link failure, we introduced the LFN as a node along an LFA routing path which can be located more than one hop away from the originating node. From this perspective, it is interesting to investigate the distribution (or histogram) of hop counts between the originating node and the number of discovered LFNs, for a given execution on a given configuration of routing tables for the given topologies as depicted in Figure 5.10. This figure illustrates the added value of our approach with respect to the existing LFA approach. This can be observed by the portion of LFNs which is at a hop count distance of more than one. Whereas for most investigated topologies,

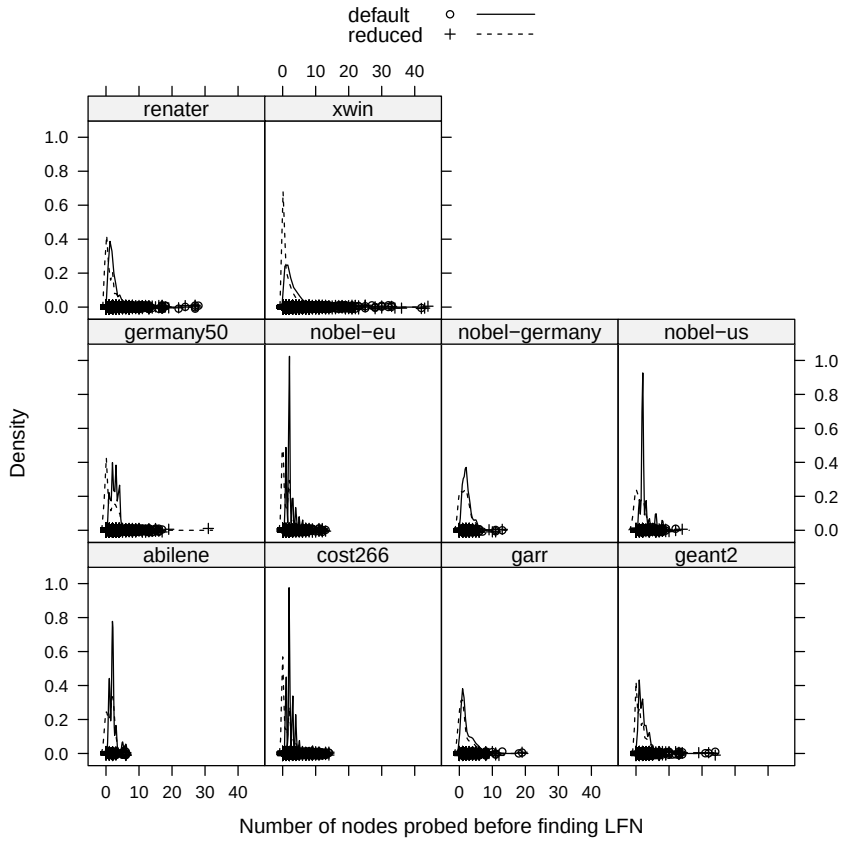


Figure 5.9: Distribution of the number of probes needed before LFN is found for a given source-destination pair

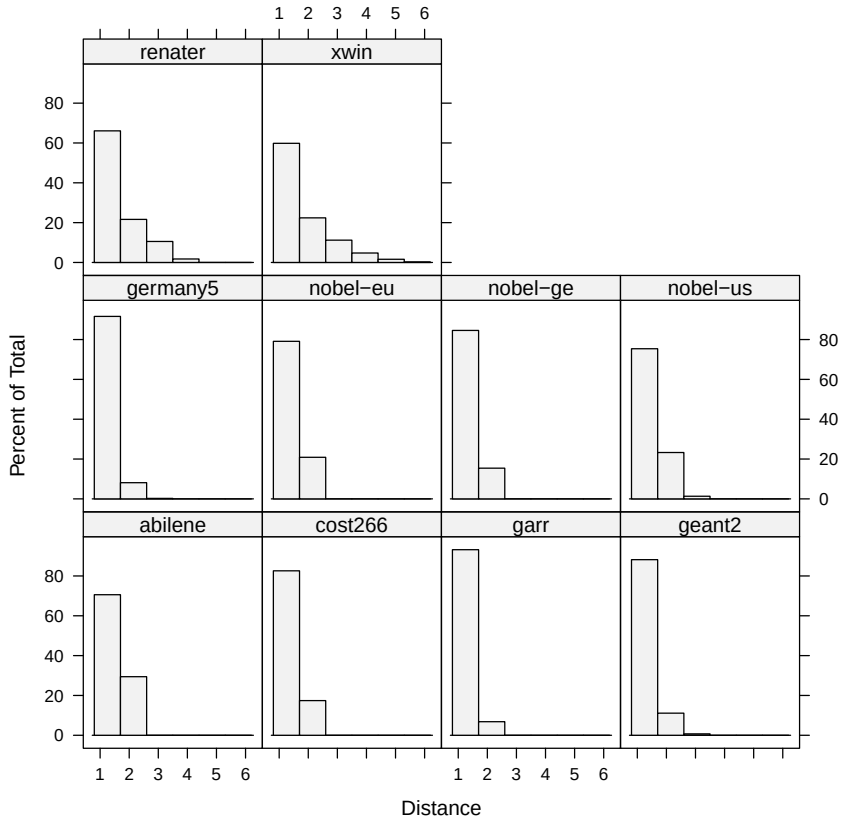


Figure 5.10: Distance from the originating node towards the LFN in hop count

LFNs are at most two hops away from the originating node, the histograms of the *renater* and *xwin* network illustrate a ”longer tail” in the resulting distribution. In these networks, only a small fraction of the LFNs was found to six hops away from the originating node. This might be explained by the fact that these networks contain sets of nodes which are only connected through cycles. If the distance between the LFN and the originating node in those cases is considerably closer via the left part of the cycle which interconnects the originating node with the network, loop-free alternate paths must go sufficiently far along the right part of the cycle to avoid that the resulting probes would return via the originating node. In other words, the loop-domain of network nodes which are mainly connected via a cycle can be rather large. Adding shortcuts in the resulting cycle would thus decrease the size of the resulting loop-domain. In its extreme, if only sufficient links are added, the resulting loop-domain could be reduced to a diameter of one hop. The latter is exactly the idea which is the basis for the research done in [16].

Whereas the results of Figure 5.9 provided an indication on the required probing effort, Figure 5.10 gives an indication of the required configuration effort. The longer the distance (in hop count) from the originating node to the LFN, the more configurations of alternate entries are required (cfr. Section 5.4.3 and Figure 5.5). In addition the resulting hop count distribution of LFNs illustrates that not only the average number of probes is very limited (cfr. previous section), but also that a very high percentage of probes remains very local to the originating node.

5.6.4.3 Coverage

For each topology (see Table 5.1), every possible single link failure was simulated. When a configured alternate forwarding entry (using one of the three presented techniques) is not able to recover the connectivity between all network nodes for a given link failure, the link is considered to be uncovered. The percentage of links which cannot be fully recovered upon link failure is denoted as the link failure coverage of the technique (the complement of this percentage denotes the percentage of links which can induce cycles among at least one source destination pair). Figure 5.11 depicts the link failure coverage of all considered re-routing schemes on all evaluated networks. From this figure, we may observe that provisioning randomly alternate entries is (as expected) is only able to cover 20 to 50 percent of the link failures. FRR-LFA is able to cover larger percentages of link failures, typically between 50 and 80 percent. The ALFA technique which we propose is able to cover almost all link failures, or at least 95 percent. The probability that a link failure will cause a cycle when selecting the alternate routing path between a random pair of nodes can be calculated, by evaluating the connectivity between all pairs of nodes, for all possible single link failures. Figure 5.12 shows the resulting end-to-end cycle probability induced by a single link failure for the experimented networks for all the considered schemes. One can observe from this figure that the

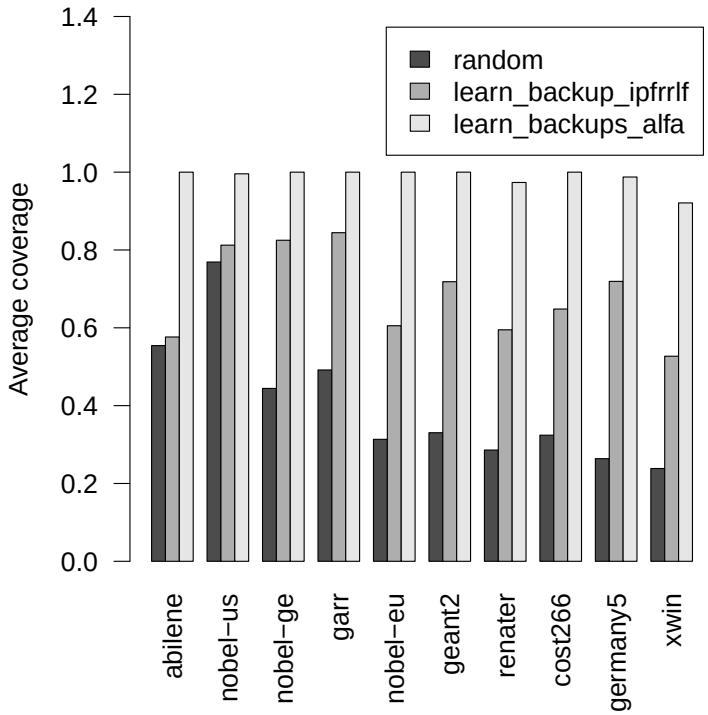


Figure 5.11: Average link failure coverage vs. topology

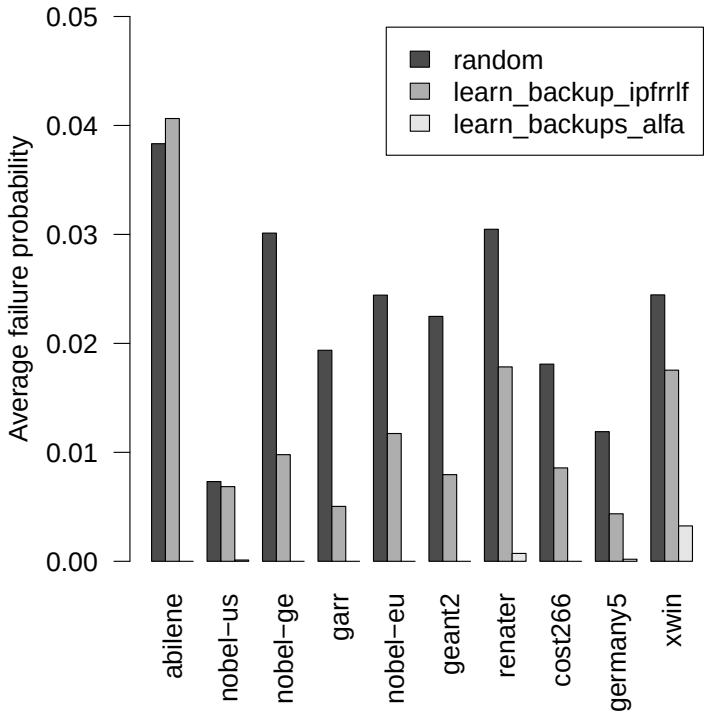


Figure 5.12: Average failure probability vs. topology

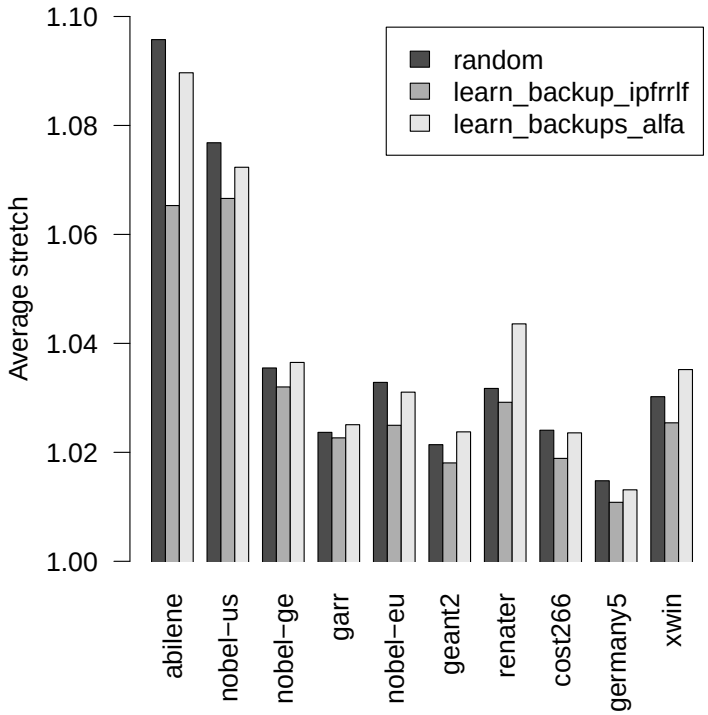


Figure 5.13: Average communication cost vs. topology

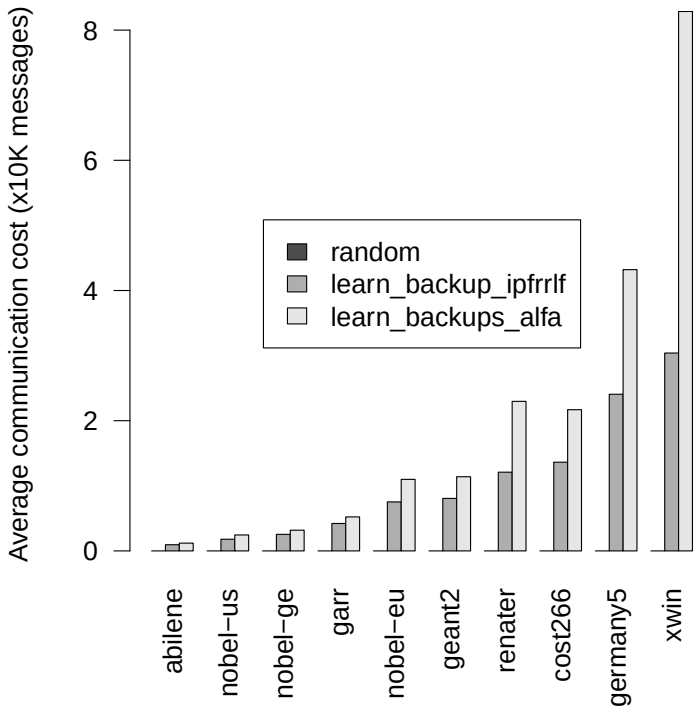


Figure 5.14: Average communication cost vs. topology

probability that a cycle occurs between a given pair of nodes is highest when only providing random alternate entries, and lowest - close to zero - when using the ALFA scheme. For small networks, the difference between provisioning random alternate forwarding entries and FRR-LFA is negligible.

Network	***Default***			***Reduced***		
	MIN	***AVG***	***MAX***	***MIN***	***AVG***	***MAX***
abilene	0.12412	0.18055	0.22083	0.09188	0.14772	0.21119
cost266	0.55599	0.93187	1.55045	0.36584	0.61421	0.94127
garr	0.24987	0.43363	0.83345	0.19345	0.35004	0.67867
geant2	0.36108	0.64775	1.19942	0.26918	0.46562	0.83510
germany50	0.73973	1.38857	2.07243	0.46899	0.87714	1.34882
nobel-eu	0.41582	0.62480	0.97031	0.25464	0.41695	0.69477
nobel-germany	0.21922	0.32173	0.48526	0.18698	0.26180	0.44172
nobel-us	0.20150	0.32054	0.49486	0.13864	0.25941	0.36591
renater	0.48997	0.93456	1.72960	0.33526	0.65577	1.25889
xwin	0.88634	1.94861	3.83970	0.56727	1.20505	2.54612

Table 5.2: Total time (in seconds) needed for self-configuration in different nodes in a network

5.6.4.4 Stretch

The previous metric measured the quality of the protection techniques in terms of the proportion of link failures they can potentially (fully) recover from. However, this metric doesn't give us information on the quality of the resulting alternate routing paths with respect to the path length. It may be expected, that higher recoverability could have a detrimental influence on the resulting path length. To assess this assumption, we calculate and depict the average stretch of the alternate routing paths of all techniques in all networks in Figure 4. The stretch metric indicates the ratio of the length of the alternate routing path (in hop count) vs. the length of the shortest routing path when no failure occurs. When the alternate routing path has the shortest length, the stretch is equal to 1. The average stretch calculates the average ratio over all resulting path lengths. Figure 5.14 illustrates that the length of the selected alternate paths taken out of all experimented techniques is at worst 10 percent longer than the (primary) shortest path between two nodes. FRR-LFA in general uses the shortest backup paths. This may be a consequence of the fact that, when using LFAs, only the first hop is different from the (primary) shortest path in the network, while the ALFA technique may use longer detour paths to ensure that the packet is out of the loop-domain of the failure detecting node. This explains the higher stretch values for the ALFA technique.

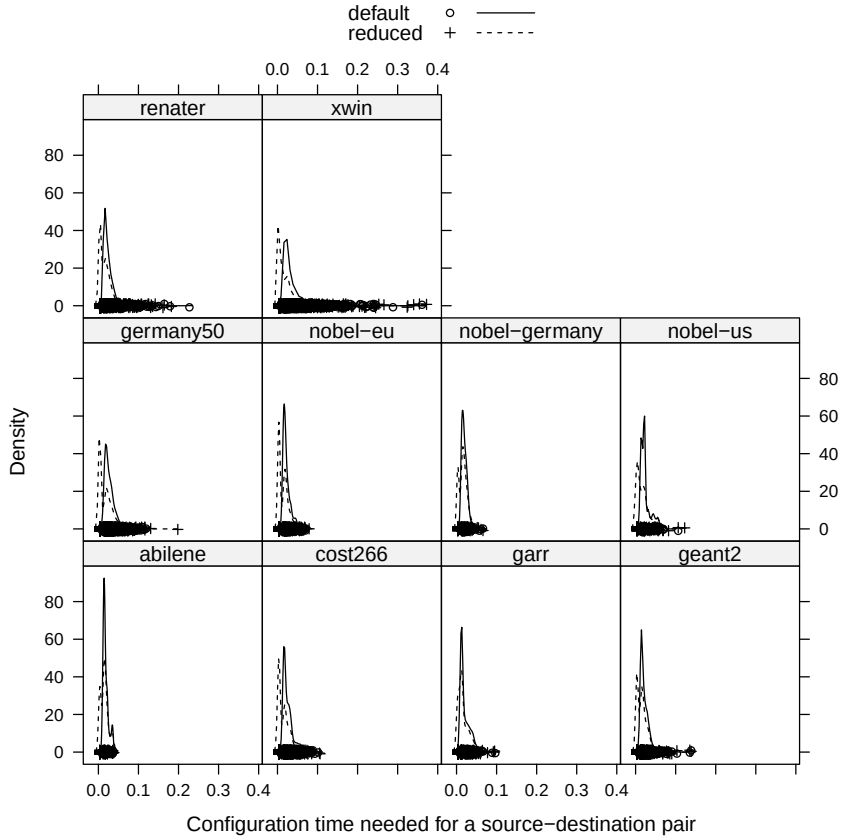


Figure 5.15: Distribution of the configuration time (in s) per source-destination pair

5.6.4.5 Communication cost

Finding adequate alternate routing paths ensuring that no cycles occur requires some probing and learning activity in the network. Clearly techniques relying on probing lead to a cost with respect the number of probing messages. Both FRR-LFA and the ALFA technique involve probing: the first probes for a loop-free alternate neighbor, the second probes for a node out of the loop-domain of the failure detecting node. Figure 5 depicts the number of probing messages that were needed before the required techniques converged. The results illustrate that ALFA has probing communication cost between 20 percent (for the smallest networks) and 270 percent (for the largest) higher networks.

5.6.4.6 Time evaluation

The time model specified in Section 5.5 allows us to derive an indication of the time that is required to execute the proposed self-configuration process over all source-destination pairs for the considered network topologies. The parameters that were used to evaluate the required configuration time are:

Parameter	Value
t_{prop}	$800\mu s$
t_{ctrl}	$5\mu s$
t_{conf}	$10\mu s$
t_{gen}	$10\mu s$
t_{det}	$20ms$
t_{act}	$10\mu s$

Table 5.2 lists the minimum, average and maximum time needed for calculating all alternate paths in a node of a network, for both the default BFS+ LFN search scheme, as well as the reduced probe scheme. It can be read from the table that average configuration times needed by nodes in the investigated networks are between 0.2 s and 2 s using the BFS+. Using the probe reduction method defined in Section 5.4.2.2, the required configuration time could be reduced between 20 and 40 percent. The maximum time needed by a node, and consequently the time needed by the entire network for complete self-configuration was less than 4 s in the investigated networks (maximum of 3.8 s in the *xwin* network). Given the fact that the entire self-configuration process can happen in background, this is a very acceptable number.

Closely related to the number of required probes (Section 5.6.4.1), Figure 5.15 depicts the distribution of the required time that is needed for finding an LFN and configuring the related alternate entries in a network. While the resulting time range can change significantly among the networks (leading to maximally 0.4 s for 1 entry), the average time needed for handling one entry is very close in all

networks. The large majority of LFNs requires about 0.024 s to handle one entry. If the optimized LFN search is used, the resulting time is reduced with about 30 percent, leading to 0.0174 s.

5.7 Conclusion

IP Fast-ReRoute techniques enable carrier-grade recovery in connectionless IP networks. Upon network failure detection, these techniques activate pro-actively calculated alternate forwarding entries to forward received traffic along an alternate path. The currently most adopted IPFRR-technique makes use of Loop-Free Alternates. While this is simple to apply, the scheme is not able to provide full link failure coverage and has no automated configuration scheme. In this paper, we proposed a self-configuring extension of Loop-Free Alternates (LFA) to automatically populate the alternate routing entries which operates without modifying the link-state routing protocol operations outside of the transient re-routing periods (as alternate entries take local precedence over default IGP routing entries). Using a simulation environment, we evaluated the performance of the proposed scheme with respect to the failure coverage, the required time for self-configuration and the resulting path length (stretch), on a representative set of 10 network topologies. Results show that our technique provides nearly 100 percent link failure coverage with an alternate path length at worst 10 percent longer than the (primary) shortest path.

Acknowledgment

This work is supported by the European Commission (EC) Seventh Framework Programme (FP7) ECODE project (Grant nr. 223936) and EULER project(grant nr. 258307).

References

[1] J. Moy. *OSPF Version 2*. RFC 2328, Internet Engineering Task Force, April 1998.

[2] Y. Rekhter, T. Li, and S. Hares. *A Border Gateway Protocol 4 (BGP-4)*. RFC 4271 (Draft Standard), January 2006. Available from: http://www.ietf.org/rfc/rfc4271.txt.

[3] P. Francois, C. Filsfils, J. Evans, and O. Bonaventure. *Achieving sub-second IGP convergence in large IP networks*. ACM SIGCOMM Computer Communication Review, 35(3):35–44, 2005.

[4] D. Awduche, L. Berger, D. Gan, T. Li, V. Srinivasan, and G. Swallow. *RSVP-TE: Extensions to RSVP for LSP Tunnels*. RFC 3209 (Proposed Standard), December 2001. Updated by RFCs 3936, 4420, 4874, 5151, 5420, 5711. Available from: http://www.ietf.org/rfc/rfc3209.txt.

[5] R. Braden, L. Zhang, S. Berson, S. Herzog, and S. Jamin. *Resource ReSerVation Protocol (RSVP) – Version 1 Functional Specification*. RFC 2205 (Proposed Standard), September 1997. Updated by RFCs 2750, 3936, 4495, 5946. Available from: http://www.ietf.org/rfc/rfc2205.txt.

[6] D. Hock, M. Hartmann, M. Menth, M. Pióro, A. Tomaszewski, and C. Żukowski. *Comparison of IP-based and explicit paths for one-to-one fast reroute in MPLS networks*. Telecommunication Systems, pages 1–12, 2011.

[7] A. Jarry. *Fast reroute paths algorithms*. Telecommunication Systems, pages 1–8, 2011.

[8] M. Shand and S. Bryant. *IP Fast Reroute Framework*. RFC 5714 (Informational), January 2010. Available from: http://www.ietf.org/rfc/rfc5714.txt.

[9] C. Hopps. *Analysis of an equal-cost multi-path algorithm*. 2000.

[10] A. Gunnar and M. Johansson. *Robust load balancing under traffic uncertaintytractable models and efficient algorithms*. Telecommunication Systems, pages 1–15, 2011.

[11] A. Atlas and A. Zinin. *Basic Specification for IP Fast Reroute: Loop-Free Alternates*. RFC 5286 (Proposed Standard), September 2008. Available from: http://www.ietf.org/rfc/rfc5286.txt.

[12] P. Francois and O. Bonaventure. *An evaluation of IP-based fast reroute techniques*. In Proceedings of the 2005 ACM conference on Emerging network experiment and technology, pages 244–245. ACM, 2005.

[13] M. Gjoka, V. Ram, and X. Yang. *Evaluation of IP fast reroute proposals*. In Communication Systems Software and Middleware, 2007. COMSWARE 2007. 2nd International Conference on, pages 1–8. IEEE, 2007.

[14] A. Hansen, T. Cicic, and S. Gjessing. *Alternative schemes for proactive IP recovery*. In Next Generation Internet Design and Engineering, 2006. NGI'06. 2006 2nd Conference on, pages 8–pp. IEEE, 2006.

[15] P. Francois, O. Bonaventure, M. Shand, S. Bryant, and S. Previdi. *Loop-free convergence using oFIB*. Work in Progress, 2008.

[16] G. Rétvári, J. Tapolcai, G. Enyedi, and A. Császár. *IP Fast ReRoute: Loop Free Alternates Revisited*. IEEE INFOCOM, available online: http://opti. tmit. bme. hu/˜ enyedi/ipfrr/, Shanghai, China, 2011.

[17] A. Atlas et al. *U-turn alternates for IP/LDP fast-reroute*. IETF Draft, Feb, 2006.

[18] S. Bryant, C. Filsfils, S. Previdi, and M. Shand. *IP Fast Reroute using tunnels*. Work in Progress in IETF, 2005.

[19] S. Bryant, M. Shand, and S. Previdi. *IP fast reroute using not-via addresses*. draft-bryant-shand-ipfrr-notvia-addresses-03. txt, 2006.

[20] N. Hu and P. Steenkiste. *Quantifying Internet end-to-end route similarity*. In Passive and Active Measurement Conference, volume 2006, page 76. Citeseer, 2006.

[21] S. Orlowski, R. Wessäly, M. Pióro, and A. Tomaszewski. *SNDlib 1.0survivable network design library*. Networks, 55(3):276–286, 2010.

6
Concluding remarks

The omnipresence of the Internet and its critical position in our modern information society imposes hard requirements on the recoverability of the network. However, the ever-increasing network usage, number of network devices and different types of network equipment impede the routing and resiliency challenges of the network, and the management tasks of the network operators in charge.

In this dissertation, novel routing and recovery schemes were designed, implemented and evaluated on connection-oriented Ethernet (Carrier Ethernet) and connection-less IP routing. In the next subsections, we recapitulate the contributions of the performed research, and put them into the context of recent trends in future network architectures, enabling directions for future work.

6.1 Routing and resilience in Carrier Ethernet

Traditional connectionless Ethernet technology, as used in LAN environments, lacks more advanced features such as wide-area scalability, guaranteed recovery times in the order of ms, traffic engineering and OAM capabilities which are desired in carrier networks. However, Ethernet switches are cheap, easy to manage (as almost no control software is required), and the supported bandwidth is high (up to 100 Gbps connectivity). For this reason, Ethernet Label Switching (ELS) was designed, implemented and evaluated in this thesis.

The resulting connection-oriented switching technology enables traffic engineering and recovery capabilities such as protection and restoration of paths and

path segments, which previously were only available in MPLS networks or optical network technologies. The work presented in Chapter 2 implemented and evaluated ELS in a realistic emulation environment. In the performed experiments, ELS was able to achieve high throughput and fast recovery times (sub 50 ms) in combination with BFD failure detection for point-to-point Label Switched Paths (LSP). Distributed control was made possible through the protocols of the GMPLS suite. Appendix A evaluated the setup and fast recovery of multipoint connectivity in the context of video streaming, and the interoperability of the ELS implementation was evaluated in Appendix B. Demonstrations at various European events confirm the readiness of the technology (see Section 1.7).

The performed experiments and research illustrate that Carrier Ethernet as implemented by ELS forms a capable alternative to connection-oriented IP/MPLS technology. Compared to circuit-switched optical technology, the benchmarked technology allows topological flexibility (meshed networks rather than ring topologies as in SDH), and can additionally benefit from statistical multiplexing (see Section 1.3.4).

6.1.1 Future directions and trends of Carrier Ethernet

Whereas performed research confirms the feasibility of Carrier Ethernet on top of commodity hardware, it remains an open question if the total cost for the network operator will be effectively lower when using Carrier Ethernet solutions compared to existing alternatives. Many market-related factors influence this situation.

In general, control software significantly increases the equipment price of routers and switches [1]. This is usually a consequence of the service contracts which are required for software maintenance, upgrades and support. However, for example ELS allows to re-use existing Ethernet VLAN-switching hardware not requiring any pre-installed control protocols (xSTP and MAC-learning even should be disabled). For the control part, network operators can rely on their own or third-party control software (e.g., GMPLS). From this angle, the proposed Carrier Ethernet architecture can be interpreted as a first step in the direction of control/-data plane separation. This increases the freedom of the network operator and the ability to evaluate multiple network control options. In addition, the possibility of being able to choose between different management or control suite implementations will open the market for control software, which most probably will result into cheaper cost control software. This is confirmed by the studies performed in [2].

Instead of sticking to Ethernet switching technology, two recent control/data plane separation evolutions push the idea even further:

1. **OpenFlow**:
   Instead of restricting connections to the L2-level, OpenFlow packet-based

switching technology [3] enables the setup of connections via any combination of header fields of L2/L3 packet headers. This enables a much wider connection concept than before, allowing almost unlimited scalability, virtualization and layering. The control of a network of OpenFlow switches is done via the OpenFlow protocol which can be steered by any software platform. The latter possibility gave rise to Software Defined Networking (SDN). Any type of routing diversity and recovery can be provided in these networks relying on a centralized software platform (possibly using distributed control devices) which has a global view of the topology.

2. **FORCES**:
   The FORCES initiative [4] at IETF is an alternative proposal to separate the interface between the forwarding and control elements. In this proposal, the forwarding plane is not restricted to any switching mode (connection-less or connection-oriented), and does not directly target SDN. Its main target is to allow reconnecting existing forwarding (e.g., OTN switch, Ethernet Switch, IP router, etc.) and control elements (GMPLS implementation, OSPF implementation, etc.) in a transparent and custom manner. This will enable the reconnection of different forwarding and control components, increasing the competition on the market between players with similar software or hardware products.

6.2 Network recovery in connection-less IP networks

The dynamics of current communication networks in terms topology changes, and variability in network usage (traffic), becomes a factor of increasing importance. Whereas the phone traffic or even the data traffic of the early days of the Internet was rather stable and predictable, today's Internet traffic reflects every change in our global society. To cope with this situation, network operators over-provision their networks. The resulting situation illustrates that the control schemes driving network routing need to include more adaptiveness and automation in their design. In this context, traffic-informed network recovery was evaluated, as well as self-configuring network recovery in the context of connection-less IP networks.

6.2.1 Traffic-informed network recovery

The research documented in Chapter 3 illustrated that network recovery may profit from applying learning functionality on network traffic. Indeed, when IP routers update their routing tables upon failure detection, a number of forwarding/routing entries can be affected and needing to be updated. As long as an entry is not updated, related network traffic is lost. Current IP routers do not involve network traffic information during this unordered process. This results into more packet

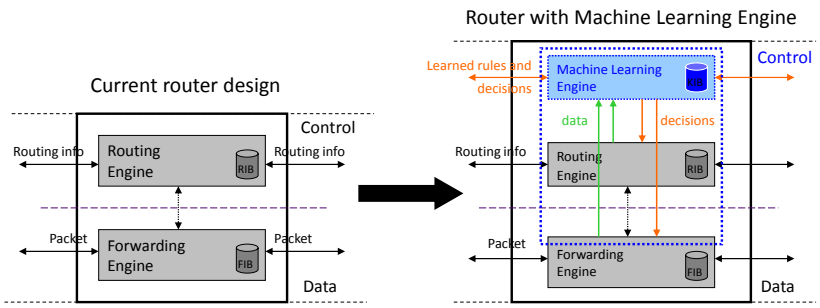


Figure 6.1: Learning-enabled routers

loss than is strictly needed. Whereas the initial studies of [5] on simulated network traffic raised high expectations in terms of potential packet loss reduction, later studies illustrated that real network traces behave very dynamic, which decreases the potential of using the most obvious network traffic models. We investigated the possible gain of this idea and validated on realistic network traces that using a network traffic model capturing the short term dynamics of network traffic could help in ordering and optimizing the routing table update.

One of the drawbacks of the proposed solution is that it requires learning at wire-speed. Traffic sampling methods could be evaluated, however the added value of these s probably limited, given the short time scale to be considered. From this perspective, the question remains if the potential gain of the proposed mechanism outweighs the resulting cost in terms of high speed hardware that would be required. In this context, applying learning patterns in control traffic as applied in [6] is more practical.

6.2.1.1 Future directions and trends

The idea of implementing traffic learning and adaptivity abilities into routers is in line with trends in recent research towards self-adaptive network elements. The recently finished ECODE project[1], proposes a novel router architecture which explicitly includes a (machine) learning component within the design of a router. The architecture is illustrated in Figure 6.1. This component allows bringing learning within the control loop of the network, such that routing, forwarding or admission control can be adapted as a direct consequence of learning applied on data or control traffic. This requires traffic monitoring points, adequate aggregation/sampling methods, and can potentially involve pattern analysis (e.g., for anomaly detection), modeling and/or prediction. Learning-related information can be distributed over

[1] lead by D. Papadimitriou from Alcatel-Lucent Bell, see http://www.ecode-project.eu/

the network to improve distributed learning and help cooperative behavior to improve the adaptiveness of the resulting routing infrastructure.

6.2.2 Self-configuring recovery

IGP-based routing is still widely used within today's networks. They scale relatively well, and require limited management effort from the network operator. The latter aspect is becoming increasingly important in a network infrastructure which is continuously growing and existing of more and more heterogeneous equipment. On the other hand, many network operators refrain from using these routing protocols in their backbone networks because of the low guarantees that can be given regarding the recovery time in case of network failures, which can be in the order of seconds to minutes.

For this reason, IP-FastReRoute (IP-FRR), and in particular Loop-Free Alternate (LFA) schemes have been designed and deployed. The underlying idea of these schemes is that alternate routes could be calculated pro-actively, and directly activated once a failure is detected. As a drawback these approaches are typically only able to recover from 60 to 70 percent of the link failures, and require configuration effort by the network operator.

Chapter 5 proposed a self-configuring Loop-Free Alternates with increased (near 100 percent) link failure coverage. This approach initiates search probes on an individual node basis, resulting in automated configuration of alternate entries. Upon failure occurrence, the loop-free alternate entries are activated in order to reach the destination. The activation of the effective switch-over could be driven by a failure detection scheme such as BFD. Chapter 4 extended BFD for efficient failure detection between IP networks interconnected via an Ethernet multi-access segment.

6.2.2.1 Future directions and trends

The proposed approach matches with the increasing need for a closed control loop in communication networks. Often, this is driven from the angle of network management where it is realized that the ever-growing network sizes, and the related network diversity result into an almost impossible challenge of efficient network configuration. This is confirmed by the increased interest for Autonomic Networking. The latter follows the concept of Autonomic Computing, an initiative started by IBM [7]. Its ultimate aim is to create self-managing networks to overcome the rapidly growing complexity of communication networks and to enable their further growth, far beyond the size of today. The resulting control loop is often referred as MAPE, which stands for: Monitoring, Analysis, Planning and Execution. This approach allows network elements to take decisions fully automated and autonomously, whereas otherwise decisions would be manually (re-)configured by

network operators. Several alternate autonomic control loops further extended this architecture [8, 9].

Despite the design of the autonomic network control schemes, the idea of a fully automated and closed control loop is too often ignored in the design of network protocols, especially in the context of network recovery. With the proposed scheme, a first step was made towards the direction of cheap and fast autonomic recovery of traditional IP networks. Future work could consist of building prototypes of the simulated approach, and extending alternative network recovery and signaling protocols according to the principles of autonomic networking, in order to fully close the network control loop with respect to network recovery.

References

[1] L. Network Strategy Partners. *Total Cost of Ownership Comparison: Ethernet Transport vs. VPLS and MPLS in the Metro Network* , 2008. Available from: http://www.nspllc.com/whitepapers.html.

[2] K. Casier, W. Tavernier, D. Colle, M. Pickavet, D. Papadimitriou, and P. Demeester. *Forecasting Cost Trends for Carrier Ethernet*. In GLOBECOM Workshops, 2009 IEEE, pages 1–6. IEEE, 2009.

[3] N. McKeown, T. Anderson, H. Balakrishnan, G. Parulkar, L. Peterson, J. Rexford, S. Shenker, and J. Turner. *OpenFlow: enabling innovation in campus networks*. ACM SIGCOMM Computer Communication Review, 38(2):69–74, 2008.

[4] L. Yang, R. Dantu, T. Anderson, and R. Gopal. *Forwarding and Control Element Separation (ForCES) Framework*. RFC 3746 (Informational), April 2004. Available from: http://www.ietf.org/rfc/rfc3746.txt.

[5] W. Tavernier, D. Papadimitriou, D. Colle, M. Pickavet, and P. Demeester. *Optimizing the IP router update process with traffic-driven updates*. In Design of Reliable Communication Networks, 2009. DRCN 2009. 7th International Workshop on, pages 115–122. IEEE, 2009.

[6] G. Das, D. Papadimitriou, W. Tavernier, D. Colle, T. Dhaene, M. Pickavet, and P. Demeester. *Link State Protocol data mining for shared risk link group detection*. In Computer Communications and Networks (ICCCN), 2010 Proceedings of 19th International Conference on, pages 1–8. IEEE, 2010.

[7] J. Kephart and D. Chess. *The vision of autonomic computing*. Computer, 36(1):41–50, 2003.

[8] S. Dobson, S. Denazis, A. Fernández, D. Gaïti, E. Gelenbe, F. Massacci, P. Nixon, F. Saffre, N. Schmidt, and F. Zambonelli. *A survey of autonomic communications*. ACM Transactions on Autonomous and Adaptive Systems (TAAS), 1(2):223–259, 2006.

[9] Z. Movahedi, M. Ayari, R. Langar, and G. Pujolle. *A Survey of Autonomic Network Architectures and Evaluation Criteria*. Communications Surveys & Tutorials, IEEE, (99):1–27.

A

Point-to-multipoint connectivity and recovery in Ethernet Label Switching

In this appendix, we extend the work of Chapter 2 by implementing and evaluating point-to-multipoint connectivity and recovery in Ethernet Label Switching.

⋆⋆⋆

W. Tavernier, D. Papadimitriou, D. Colle, M. Pickavet, and P. Demeester.

Presented at the 9th workshop in G/MPLS networks, Girona, 2011

Abstract Carrier-grade Ethernet is a recent technology evolution where the most attractive features of bridged Ethernet are enhanced with highly capable control and forwarding features. High speed physical (PHY) interfaces at low cost, combined with transport network capabilities make it the choice of the future for many network operators. We show that Ethernet Label Switching (ELS) is capable of efficiently setting up and recovering both point-to-point and point-to-multipoint data paths. We have evaluated both techniques in an emulation setup and show that high speed recovery is possible in a realistic setting.

A.1 Introduction

For decades Ethernet is dominating the LAN environment. Ethernet bridging has become a synonym for a cheap, plug-and-play and highly compatible network technology. The common Ethernet usage in enterprise networks together with ubiquitous deployment trends results in an ever increasing demand to providers for Ethernet services enabling to interconnect several branches of companies (e.g., Virtual Private LAN Service (VPLS) and Virtual Private Wire Services (VPWS).

This shift towards packet-oriented inter-connection services has also its consequences on the transport technology that is being envisioned by operators. Why would they still use more expensive circuit-based optical equipment, involving several conversion layers, e.g., GFP, VCAT, etc., if the majority of services is becoming more and more packet-based (elastic and streaming traffic). This reasoning will become even more striking, given the highly increasing Ethernet PHY speeds, going from 10 Gbps towards 40 and 100 Gbps. Therefore using Ethernet directly as a transport technology in access-, (metro-)aggregation or even core networks becomes more and more attractive.

However, base principles of Ethernet bridging of learning and flooding within a restricted virtual tree topology (based on Rapid Spanning Tree Protocol (RSTP)) is far too restrictive for use as a transport technology. This resulted in a spectrum of new Ethernet technology designs that were developed by IEEE, ITU-T and IETF. In this paper, we focus on Ethernet Label Switching (ELS) as one of the most attractive carrier-grade Ethernet technologies.

The structure of the paper is as follows. Section A.2 describes the architecture and main functional blocks of ELS. An overview is given of possible recovery techniques for ELS point-to-point connectivity in Section A.3. The next section (Section A.4) describes how ELS is able to provide multipoint connectivity and corresponding recoverability. A realistic experimental setup is described in Section A.5, where we show that a mix of point-to-point (P2P) and point-to-multipoint (P2MP) label switched paths (LSPs) can be efficiently combined and recovered in an emulation environment. Finally, a conclusion is given in Section B.6.

A.2 Ethernet Label Switching

ELS is a connection-oriented forwarding scheme that is able to use Generalized Multi-Protocol Label Switching (GMPLS) as its control suite with Open Shortest Path First-Traffic Engineering (OSPF-TE) for routing and Resource reSerVation Protocol-Traffic Engineering (RSVP-TE) for signaling logical data paths over an Ethernet Network. ELS relies on the Provider Bridges (IEEE 802.1ad) recommendation to perform label switching in a similar way as performed in Multi-Protocol Label Switching (MPLS) as specified in RFC 3031. It encodes the label in the

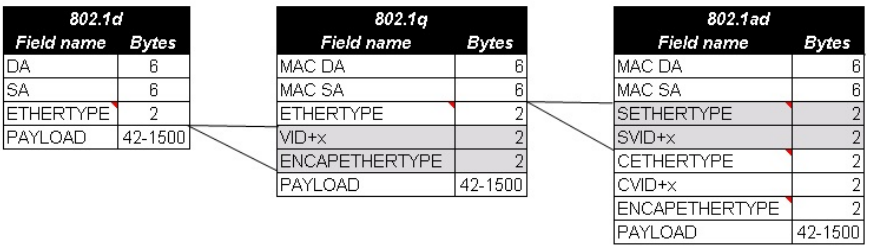


Figure A.1: IEEE Ethernet framing standards

S-VLAN ID (S-VID) tag field of the related frame header (see Figure A.1). The Ethernet S-VID label space has link local scope and local significance: thus providing 4096 (12 bits) values per interface and allowing intermediate ELS switches to translate the S-VID value resulting logically into a label swapping operation as it is the case in MPLS networks.

The logical data paths established using ELS are called Ethernet label switched paths (E-LSP). Intermediate nodes are called Ethernet Label Switching Router (E-LSR). Ingress/egress E-LSR where a LSP starts and ends, provide for a Ethernet Label Edge router (E-LER) functionality. Figure A.2 describes the label operations along an Ethernet LSP.

When a native Ethernet frame arrives to the ingress LSR, its E-LER function based on the information of the frame header, pushes the corresponding label (i.e. adding an S-TAG with the appropriate S-VID value). Then, the Ethernet VLAN-labeled frame is forwarded along the Ethernet LSP. For each E-LSR, the label is swapped (i.e. that the incoming S-VID is translated into an outgoing S-VID as defined in IEEE 802.1ad). When the frame reaches the egress LSR, its E-LER function pops the label (the S-TAG and so the S-VID are removed). Finally, the frame is sent to its destination as a native Ethernet frame.

It is important to underline that ELS maintains a control state per data path but keeps the forwarding paradigm of existing Ethernet switches unchanged except for the fact that forwarding entries are defined per port. In a sense, signaling is used to restrict the incoming and outgoing S-VID per port. The rest of the forwarding process is performed as specified in IEEE 802.1ad.

A.2.1 Forwarding tables and merging capability

The logical forwarding behavior borrows from existing terminology of MPLS forwarding as defined in RFC 3031. In practice, this means that three main tables were developed such as to allow the setup of end-to-end ELS LSPs. Instead of using IP prefixes to match incoming frames in the head-end ELS switch to the configured ELS LSP, the incoming port index, possibly in combination with the

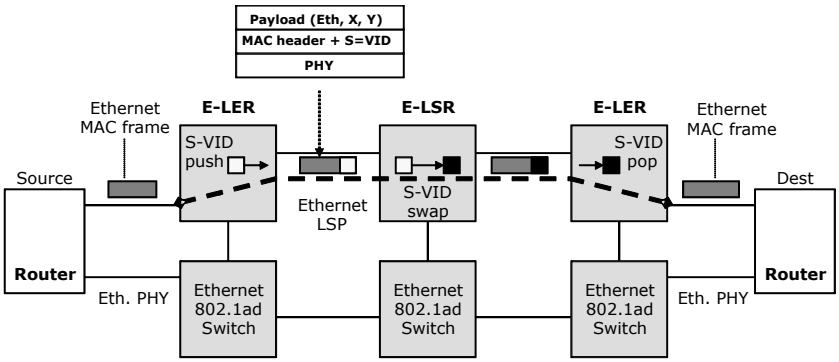


Figure A.2: GELS LSP

incoming (customer) VLAN-tag, is determining the ELS LSP to be used, i.e., outgoing frames are tagged with the S-VID associated to that LSP on the corresponding outgoing link). The following tables and associated actions were implemented :

- *FTN (FEC-TO-NHLFE)*: The FTN table is used at the ingress (or head-end) ELS switch to match or classify incoming frames (based on their incoming interface and VLAN-tag) to entries in the NHLFE table.
- *ILM (Incoming Label Map)*: The ILM table is used at intermediate and egress (or tail-end) ELS switches to match or classify incoming ELS frames based on their incoming interface and S-VID label to entries in the NHLFE table. Together with this matching operation, a POP action is performed on the incoming S-VID label, meaning that the S-VID label is taken away from the frame header, such as to make PUSH actions subsequently possible.
- *NHLFE (Next Hop Label Forwarding Entry)*: The NHLFE table is used at ingress, intermediate and egress ELS switches in order to forward them into the ELS network having a S-VID label on which can be further switched. This implies that a PUSH action, meaning that a S-VID label is attached to the frame header.

The link-local significance and the above forwarding table characteristics allow an efficient label re-usage in the network. Figure A.5 illustrates in a given network that two separate LSPs, sharing a common segment towards a destination, can use the same label. In this figure, the upper LSP (solid line for LSP 1) coincides with the lower LSP (dashed line for LSP 2) on the last segment of their path. If no label merging would be allowed, as in Figure A.5 (a), LSP 1 and LSP 2 would need to allocate a different label for the link between node d and node e (L4 for LSP 1 and L40 for LSP 2). If merging is allowed, as in ELS, both LSP's can use the same

label on the link d-e (L4 on the figure). On the lower level forwarding structure, label merging is allowed by coupling a two distinct ILM-entries (on the figure: frame with L3 from node c and frame with L30 from node d) to the same NHLFE entry (related to pushing label L40 and forwarding on link d-e in the figure).

A.3 Recovery of point-to-point (P2P) connectivity

The forwarding machinery of ELS allows for a wide range of recovery techniques point-to-point (p2p) data oaths (LSPs with one source and one destination), including both protection (recovery path is pre-computed and installed in the network) and dynamic re-routing a.k.a restoration (recovery path is computed after failure occurrence).

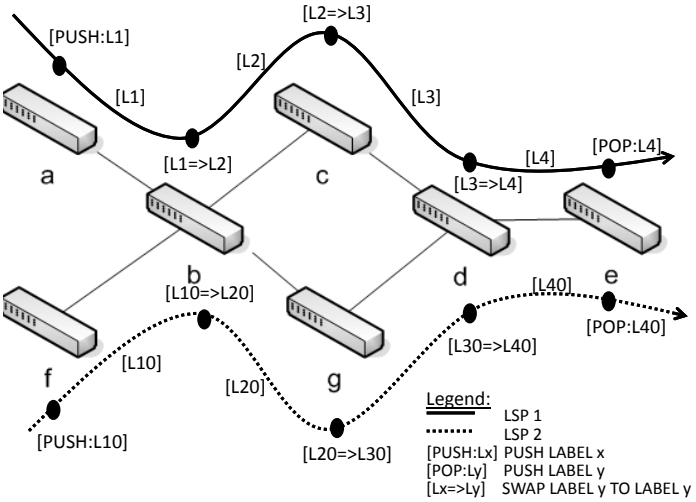
An overview of allowed recovery techniques is given in Figure A.4. Link and node recovery can be obtained by using a detour sub-path for the corresponding link (link bypass, e.g., one can recover from failing link b-c by passing via node f) or node (node bypass, e.g., failing node b can be circumvented by passing via node g) upon failure occurrence. Extending the concepts of node and link bypass towards bypassing an arbitrary segment of an LSP is referred to as segment recovery (e.g, recovery from a failure between node a and node d can be done via a backup segment via node i and j in the figure). At last, recovery from an arbitrary failure along the path of an LSP, can be achieved using end-to-end recovery (e.g., using the backup path via node k and l). The IETF has defined a set of signaling extensions for GMPLS segment recovery using RSVP-TE in RFC 4873 and for end-to-end recovery in RFC 4872.

Using protection techniques, the backup segments are pre-signaled and the forwarding entries in the backup path are already configured except for the branching points. Once the failure is detected, using for example hardware detection mechanisms or software-based techniques such as BFD (Bidirectional Forwarding Detection (BFD)), the branching points are notified and re-install the forwarding entries such that the backup paths are taken instead of the primary paths. When using dynamic re-routing techniques, the entire procedure (computing backup paths, signaling and installing forwarding entries) typically happens after failure detection.

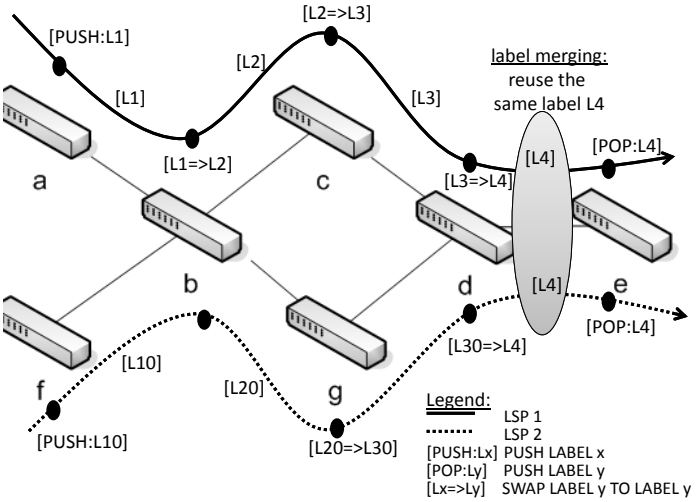
A.4 Point-to-multipoint (P2MP) connectivity

A.4.1 Provisioning and forwarding

A point-to-multipoint (p2mp) data path, i.e., an LSP with one source node and multiple destination nodes, can drastically increase the resource utilization. ELS



(a) without merging



(b) with merging

Figure A.3: Efficient label re-usage through merging

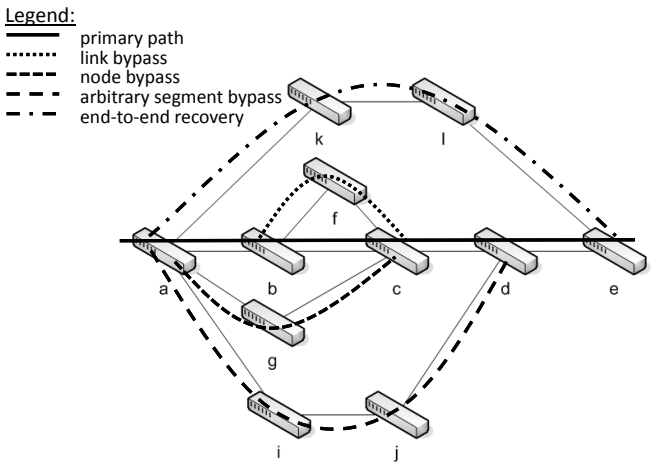


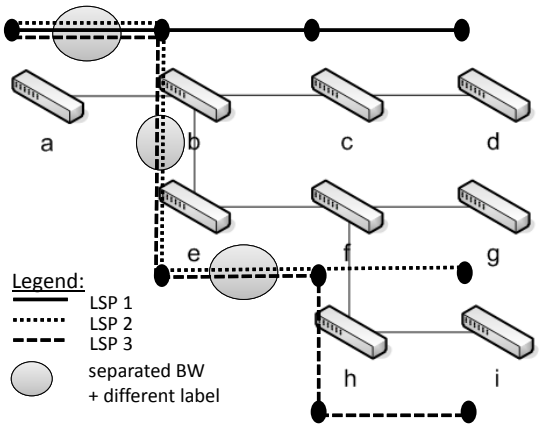
Figure A.4: P2P-recovery overview

allows p2mp LSPs at the logical low level forwarding structure because it is possible to link a given FTN- or ILM-entry to multiple NHLFE's. As such, an incoming frame can be replicated and sent towards multiple outgoing interfaces encapsulated with the label corresponding to the NHLFE's.

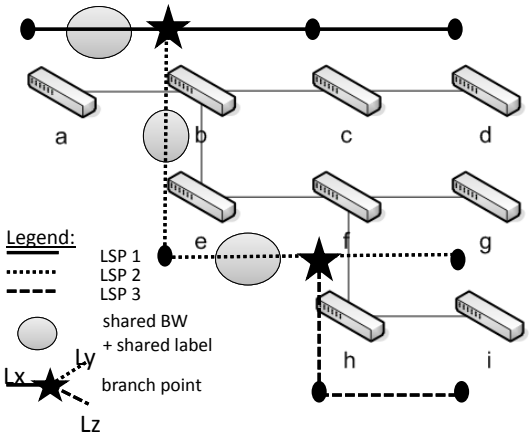
Figure A.5 illustrates the gain that can be achieved by using p2mp LSPs. Part (a) of the figure, shows how source node *a* can provide three nodes, respectively *d*, *g* and *i*, with a multicast traffic stream using 3 separate p2p LSPs. It is clear from this figure that three links in the network carry more traffic volume that is strictly needed. Link a-b carries three more times and link b-e and e-f two more times the same traffic because node *a* sends the same traffic stream over the three different p2p LSPs. In addition, the mentioned links use a separate label because otherwise the intermediate nodes would not be able to distinguish between these LSPs.

The same connectivity can be achieved using p2mp LSPs with a significant gain in resource efficiency. A p2mp LSP can be constructed via incremental addition of leaves to the p2mp LSP in signaling exchange where the root is the originating switch. A p2mp LSP can be set up in three phases: i) node *a* creates an p2p LSP towards node *d*, ii) node *a* creates an p2mp LSP towards node *e* by re-using the established LSP resulting in a branch point in node *b* which then continues towards node *e*, converting the original LSP to a p2mp LSP, and iii) node a triggers the setup of additional p2mp connectivity towards node *j* by branching in point *h*.

The resulting point-to-multipoint LSP is clearly more resource efficient because: the links a-b, b-e and f-h now share the same bandwidth and label. This leads a decrease of used bandwidth and label usage of factor 3 in link a-b, and half of the used bandwidth in links b-e and f-h.



(a) using p2p LSPs



(b) using a single multipoint LSP

Figure A.5: Multipoint connectivity

A.4.2 Recovery

Several recovery techniques exist to recover point-to-multipoint LSPs from failures. At first, all recovery techniques mentioned in Section A.3 can be re-used on parts of p2mp LSPs, i.e., any segment of the p2mp (between branching points) can be protected or restored from failure given sufficient redundancy in the network (as long as primary and protecting segments are shared-risk disjoint). As such, a p2mp can be made almost ”bullet-proof” by protecting every segment of the network by a backup segment. However, this technique may quickly lead to a lot of control overhead and bad scalability properties. For example, every protected segment requires additional setup (configuration or signaling) linked to its backup segment, and every segment needs corresponding failure detection sessions over these network parts. Therefore, the goal is to find the optimal number of protected segments that optimize resource sharing and recovery time.

End-to-end recovery of p2mp LSPs can also be handled using a backup p2mp LSP. This concept is illustrated in Figure A.6 where node a can provide nodes d, g and k with a multicast stream using either a primary p2mp LSP (using branch points node h and node i), or via a backup p2mp LSP (using branch points node b and f). In this example, both trees have large disjoint parts. From recoverability viewpoint this is a desirable characteristic. The setup using backup p2mp LSPs has several advantages: i) only two p2mp LSPs need to be set up (configured or signaled), and ii) multipoint failure detection sessions suffice for switch over traffic on the backup p2mp LSP. For example, multipoint BFD (see [1] and [2]) makes use of the configured p2mp LSP by allowing the source node to send Hello packets to all destination nodes in the p2mp LSP. Upon a pre-configured time-out interval (usually three times the inter-Hello time), destinations can notify the source node of a lossy p2mp LSP[1] to trigger the source node to switch over to the backup p2mp LSP. The main drawbacks of this technique are i) the notification time that is proportional to the leaf-root distance, ii) the ”freeze” of resource along the backup path (if the backup p2mp LSP is not provisioned with 0 capacity, i.e., path provisioned only), and iii) ensuring full recoverability mandates complete disjointness between the primary and the backup p2mp LSP.

A.5 Multipoint connectivity and recovery experimentation

In this section we will apply and evaluate the described techniques for point-to-multipoint connectivity in an emulation environment on a more realistic network scenario.

[1] the notification usually happens via an out-of-band (OOB) control network

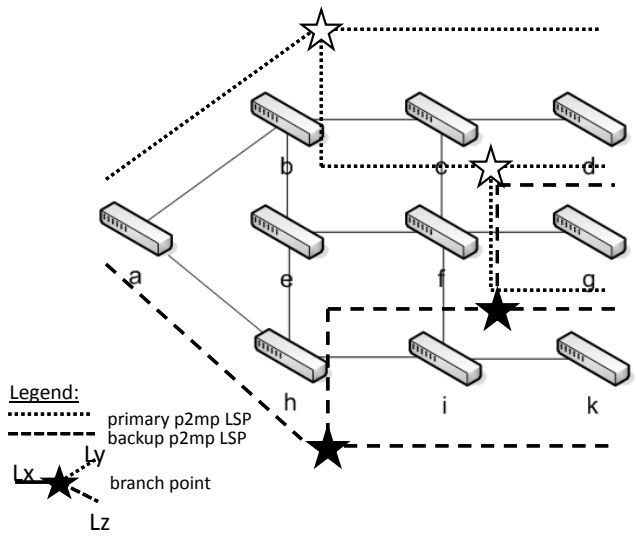


Figure A.6: Backup p2mp LSP

A.5.1 Scenario

In Figure A.7 we show a small-sized network which has some similarities to a metro-access network of an Internet service provider. In the given scenario, we would like to connect a video server with a set of 5 video clients receiving the video stream being sent. As such, point-to-multipoint connectivity is desired with as much recoverability as possible, given the low redundancy in the network (only the five node ring consisting of nodes b, c, d, g, and j is able to provide a backup path).

A.5.2 Emulation platform

We evaluate point-to-multipoint techniques in the emulation platform that was presented in [3]. In our emulation environment, one network device part (ELS switch) is imitated by a computation system running custom software, e.g., a server blade or PC. Therefore, to emulate a whole network setup, typically a set of server blades or PCs is needed (a set of Linux PC's).

For benchmarking ELS technology, an execution environment was set up using the Click Modular Router that emulates the forwarding plane ([4]), and the Dragon VLSR GMPLS software that emulates the control plane[2] ([5]). The resulting node

[2]Dragon was only used for signaling p2p LSPs, p2mp LSPs were configured by management software

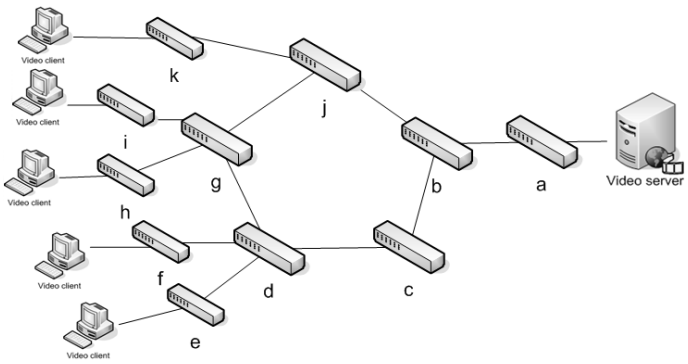


Figure A.7: Network architecture

architecture is shown in Figure A.8.

To allow arbitrary topologies with ELS emulation software, Emulab software was used on our local virtual wall at IBBT. Using a ns2 look-alike configuration script, Emulab software allows to define which software image needs to be installed on which PC part of the execution environment, and how several PC's need to be interconnected to each other, defining thus the topology.

A.5.3 Recovery of p2mp connectivity

For the given scenario, it is not difficult to set up a p2mp LSP in order to fulfill the requirement of providing all video clients with the video stream generated at source node a. We chose primary p2mp LSP shown in part (a) of Figure A.9 as basis.

Next, we evaluated two variants of p2mp LSP recovery: i) protecting the p2mp LSP using a backup p2mp LSP, and ii) recovering the p2mp LSP by means of a collection of protected segments. For the first, the chosen backup tree is shown in A.9 part (a), for the latter, we have protected all links of the ring composed by nodes b, c, d, g and j by the corresponding other half of the ring. The segment protection concept of the latter is shown for the protection of link d-c using the backup segment composed of the links d-g, g-j, j-b and b-c.

Failure detection is provided by our BFD implementation in Click. The implementation allows to detect failures at the following levels: i) link-failure detection, end-to-end failure detection and multipoint-failure detection (see [3] and [2]).

- For the first recovery variant, we only needed to set up the primary and backup LSP, as well as two multipoint BFD sessions. When connectivity is lost with the source node in one of the video clients, a notification message

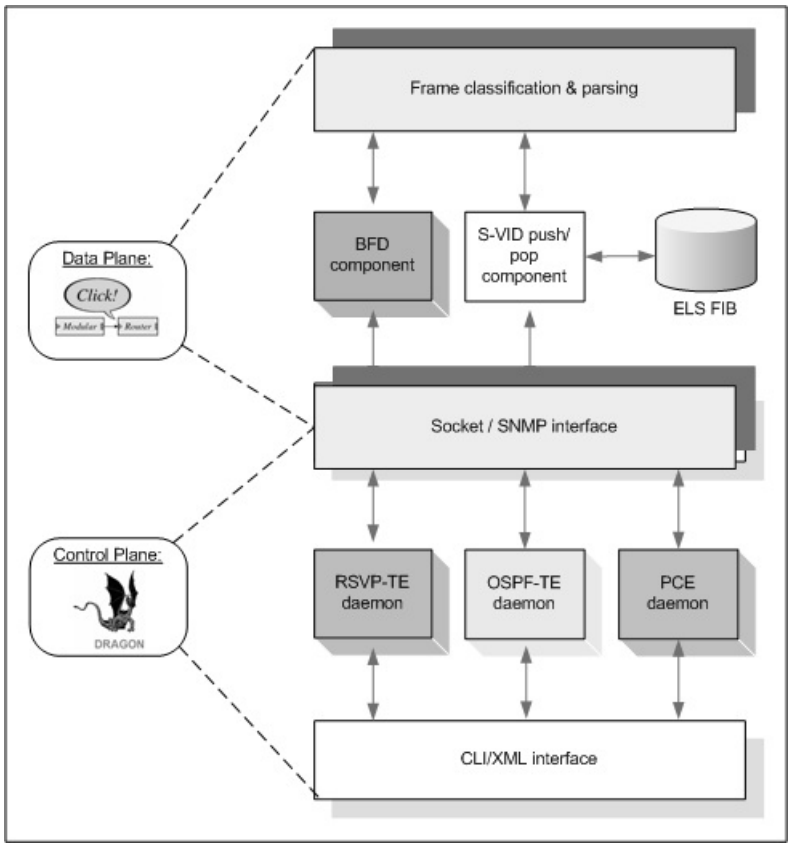
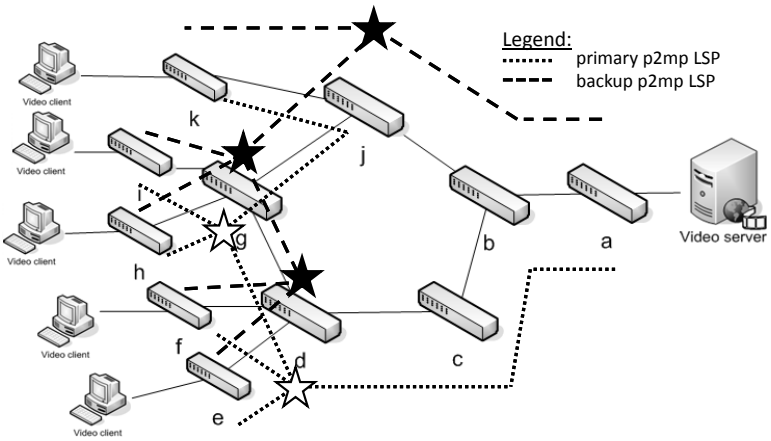
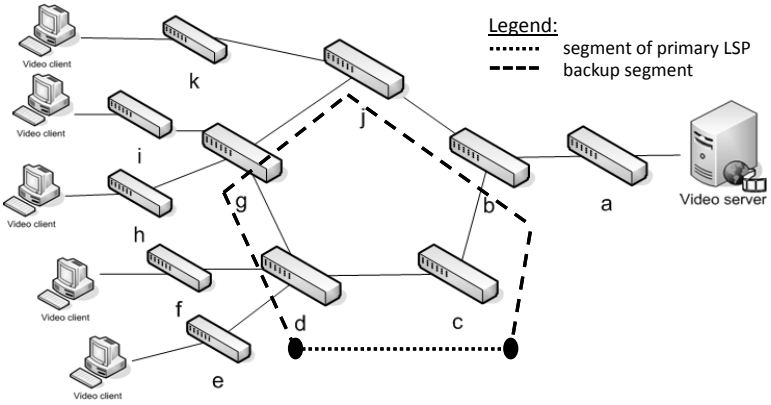


Figure A.8: Node architecture in emulation platform



(a) primary and backup p2mp LSP



(b) segment protection in p2mp LSP

Figure A.9: Recovery of p2mp LSPs

Timeline for variant 1:	Timeline for variant 2:	Event:
0	0	Failure
25	25	BFD Failure notification (5x5)
26	26	Trigger protection switch.
27	27	Send notify to branch/merge
35≤x≤38	35	Receive notify ack
35≤x≤38	35	Start fast-switchover
38≤x≤41	38	Switch-over finished

Table A.1: Recovery time upon failing link b-c or c-d

was sent using a out-of-bound (OOB) control network[3] to the source (see Figure A.10), triggering switch-over to the backup LSP (or vice versa for the switch-back).

- For the second recovery variant, the setup of 5 backup segments as well as 5 BFD sessions were needed to ensure the protection of the links in the middle ring of the network. When some of the nodes in the ring b-c-d-g-j detects that a link has failed, traffic is sent via the backup segment towards the next hop. Figure A.11 shows the recovery process when node c has detected that link d-c fails. From that moment incoming traffic at node c is sent to node d via the links c-b, b-j, j-g and g-d. The original p2mp LSP can now be continued in node d.

The recovery time needed for both approaches was very similar as it can be observed in Table A.1 when a link b-c or c-d was failing. As it can be seen in this table, the main difference results from the time needed for notifying the branch point. Whereas for the first variant (backup p2mp LSP), a notification needs to be sent to the source node a upon failure detection, the variant based on segment recovery could almost[4] directly trigger the switch-over of traffic. However, in our setup the observed difference was very minimal, because the largest time went to the internal control processing.

In Figure A.12 part (a), the performance of our BFD implementation is shown. It can be observed, depending on the length of paths (in hops) being monitored influence the lowest failure detection performance we could achieve. Up to 18 nodes, we were able to reach detection times of 15 ms using Hello-transmission speeds (TX) and equal reception timeouts (RX) of 5 ms (a failure being reported upon 3 missing Hello's). Part (b) of the figure, illustrates that the gap in the throughput using the above techniques is negligible compared to legacy bridged Ethernet failure detection and re-convergence based on RSTP.

[3]consisting of a single switch connecting all nodes
[4]depending on the endpoint of the link detecting the failure first

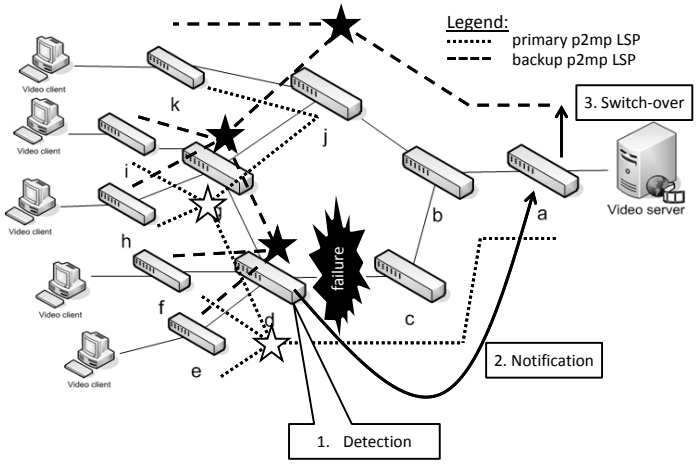


Figure A.10: Recovery process with backup p2mp LSP

Given these results, one could conclude that the there is no real reason to go for the more resource consuming variant which sets up recovery segments for all links. However, as can be seen from Figure A.9 part (a) and the described mechanism, the corresponding recovery mechanism is not able to fully recover from a failure in link d-g. This is the case, because the link is both needed by the primary and backup p2mp LSP. This illustrates that, depending on the specific network topology and likelihood of failures in specific links, that setting up dedicated segment protection can be worth to evaluate.

A.6 Conclusion

This paper introduced Ethernet Label Switching (ELS) as an efficient carrier-grade Ethernet technology for the provision and recovery of point-to-multipoint connectivity. Several recovery mechanisms were discussed and benchmarked in an emulation environment on a realistic network scenario. We showed that several alternatives are able to provide high recoverability, however some trade-offs must be considered depending on the specific failure likelihood and network topology.

A.7 Acknowledgment

This research is partly funded by The Institute for the Promotion of Innovation through Science and Technology in Flanders (IWT-Vlaanderen) through the TIGER project in the European CELTIC framework and the FP-7 project BONE.

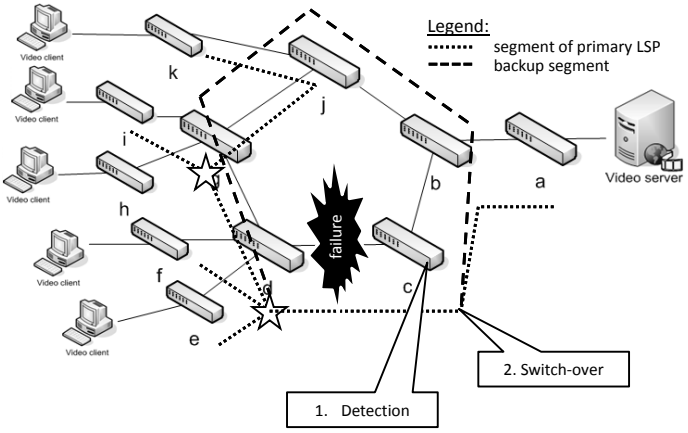
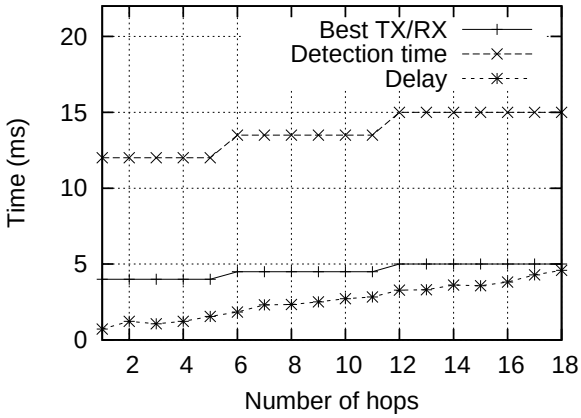
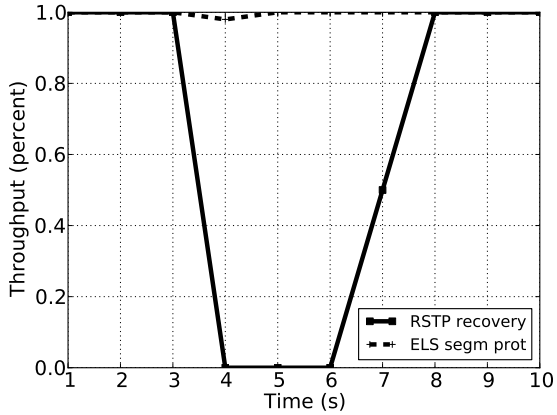


Figure A.11: Recovery process using segment recovery



(a) BFD time performance



(b) Throughput during recovery

Figure A.12: Performance of failure detection and recovery operation

References

[1] D. Katz and D. Ward. *BFD for Multipoint Networks*. Internet-Draft draft-katz-ward-bfd-multipoint-02, Internet Engineering Task Force, February 2009. Work in progress. Available from: http://www.ietf.org/internet-drafts/draft-katz-ward-bfd-multipoint-02.txt.

[2] W. Tavernier, D. Papadimitriou, B. Puype, D. Colle, M. Pickavet, and P. Demeester. *Fast Failure Detection in Multipoint Networks*. In G. Nunzi, C. M. Scoglio, and X. Li, editors, IPOM, volume 5843 of *Lecture Notes in Computer Science*, pages 51–64. Springer, 2009.

[3] W. Tavernier, D. Papadimitriou, D. Colle, M. Pickavet, and P. Demeester. *Emulation of GMPLS-controlled Ethernet Label Switching*. Testbeds and Research Infrastructures for the Development of Networks and Communities, International Conference on, 0:1–9, 2009. doi:http://doi.ieeecomputersociety.org/10.1109/TRIDENTCOM.2009.4976212.

[4] R. Morris, E. Kohler, J. Jannotti, and M. F. Kaashoek. *The Click modular router*. In SOSP '99: Proceedings of the seventeenth ACM symposium on Operating systems principles, pages 217–231, New York, NY, USA, 1999. ACM. doi:http://doi.acm.org/10.1145/319151.319166.

[5] X. Yang, C. Tracy, J. Sobieski, and T. Lehman. *GMPLS-Based Dynamic Provisioning and Traffic Engineering of High-Capacity Ethernet Circuits in Hybrid Optical/Packet Networks*. In INFOCOM, 2006.

B

Interoperability Experiment of VLAN Tag Swapped Ethernet and Transmitting High Definition Video through the Layer-2 LSP between Japan and Belgium

This chapter reports on an interoperability experiment set up between two Carrier Ethernet implementations: one at Keio University in Japan and the other one at Ghent University in Belgium. The experiment validates the resulting connection by streaming transporting High Definition Video through the Layer-2 LSP.

⋆⋆⋆

Sho SHIMIZU, Wouter TAVERNIER, Kou KIKUTA, Masahiro NISHIDA, Daisuke ISHII, Satoru OKAMOTO,Didier COLLE , Mario PICKAVET, Piet DEMEESTER, and Naoaki YAMANAKA

Published in IEICE transactions on communications, 2010

Abstract The first global interoperability experiment of GMPLS controlled Ethernet with VLAN tag swapping between two different implementations is successfully demonstrated. High definition video streaming is realized through a newly established Layer 2 Label Switched Path (L2- LSP). The results of this experiment can be applied to designing reliable Layer 2 networks.

B.1 Introduction

Wide area Ethernet is attractive for the next generation Internet backbone architecture, especially for carrier environments. This is because Ethernet is the most common net working technology and it's cost effective. In addition, the link bandwidth has been increasing: starting from 10 Mbps about 30 years ago, and reaching up to 100 Gbps nowadays. Ethernet became applicable to Wide Area Network (WAN) although it originated from Local Area Network (LAN) technology.

Providing an Ethernet Virtual Line (EVL) between customers is a basic service component. EVL is provisioned as an Ethernet Virtual LAN (VLAN) path. The Ethernet VLAN path can be established with VLAN technologies, especially with tag-based VLANs, which is standardized in IEEE 802.1Q [1]. The VLAN ID/tag is part of the Ethernet header. Wide area Ethernet using Ethernet Label Switching (ELS) [2] also forwards frames according to the value of its VLAN tag.

VLAN configuration of all switches along the Ethernet VLAN path is required when setting up or tearing down the path. Generalized Multi-Protocol Label Switching (GMPLS) [3] is a set of network control protocols to provide a next generation high performance transport network, and can be used for automatically configuring these switches in path provisioning. Therefore, we are proposing to employ GMPLS protocols for automatic Ethernet VLAN path provisioning. To increase the scalability of Ethernet, an automatic VLAN configuration technique is an important challenge especially for WAN.

In addition, the scalability of VLAN technology is an issue in wide area Ethernet. In the conventional VLAN tag- based Ethernet network (IEEE 802.1Q), a VLAN tag must be globally unique in a whole network. In other words, different Ethernet VLAN paths cannot reuse the same VLAN tag. In addition, only 12 bits are assigned to the field of VLAN tag. These imply that wide area Ethernet cannot support over 4096 Ethernet VLAN paths. That number of paths is not sufficient in WAN. We have proposed an effective network architecture to increase network scalability [4].

This paper presents experimental results of GMPLS controlled Ethernet VLAN path provisioning with VLAN tag swapping between Japan and Belgium. We successfully demonstrated the following things: 1) Interoperability between two different VLAN tag swapping based Ethernet implementations; one has been developed by Keio University, and the other has been developed by Ghent University, 2) International Q-in-Q frame [1] transmission between Japan and Belgium, 3) High definition video streaming through the established Ethernet VLAN path.

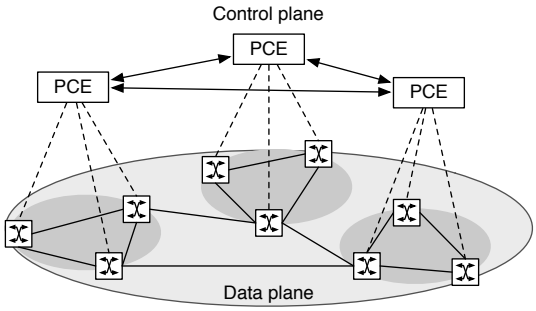


Figure B.1: Centralized wide area Ethernet

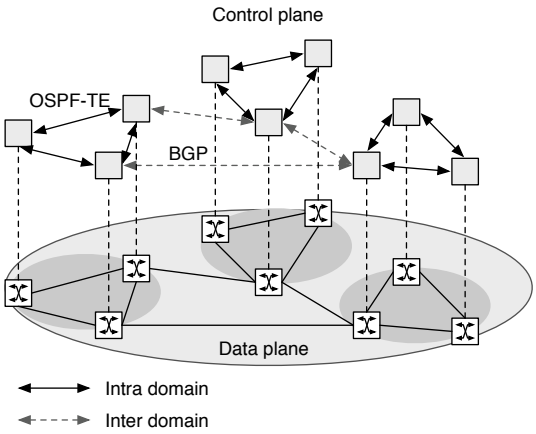


Figure B.2: Decentralized wide area Ethernet

B.2 Wide Area Ethernet Architecture

In this section, two types of wide area Ethernet architecture are discussed. One is a centralized model, and the other is a decentralized model.

Figure B.1 shows the architecture of the centralized model. It has a Path Computation Element (PCE) based architecture. A PCE has responsibilities of resource management and path calculation in a domain. When an L2-LSP is requested, a layer-2 switch demands the PCE responsible for the corresponding domain to set up a path. The PCE calculates a path coordinating with PCEs in other domains. Finally, the PCE reserves an available VLAN tag for the new path.

Figure B.2 shows the architecture of the decentralized model. It employs GMPLS control protocols such as OSPF- TE, BGP, and RSVP-TE instead of PCEs.

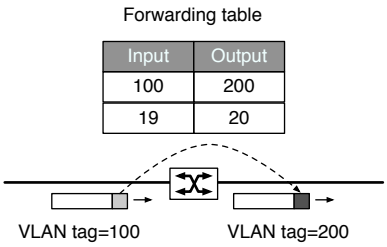


Figure B.3: VLAN tag swapping

The resource information is distributed by OSPF-TE within a domain, and the information is shared among layer 2 switches in a domain. The resource information between other domains is advertised by BGP. When an L2-LSP is ready to set up, RSVP-TE signaling from the source switch is carried out towards the destination. A path is calculated in the source switch according to the current network information, and then an available VLAN tag is reserved as triggered by signaling.

The above architectures are compared. One of the ad- vantages of the centralized model is complete management of network resources. In the centralized model, a PCE manages all of network resources within the domain, and information about the current network resources can be received without delay. On the other hand, the lack of scalability is one of the largest drawbacks in the centralized model. Man- aging all network resources is a heavy task when the number of nodes in a domain increases. High performance PCEs are required. In addition, a PCE is single point of failure. The centralized model is weak against failures.

Therefore, the decentralized model is assumed in this paper. In this model, it is desirable that network resources are locally managed. However, a VLAN tag must be globally unique in the conventional network. To extend scalability of Ethernet, we have introduced VLAN tag swapping [4]. Figure B.3 shows an example of VLAN tag swapping. The VLAN tag of an incoming frame is replaced with another VLAN tag for the corresponding outgoing frame. In this example, two configurations are stored in the forwarding table. The VLAN tag of the incoming frame is 100. It matches the first configuration of the forwarding table, therefore the VLAN tag of the outgoing frame becomes 200. In wide area Ethernet with VLAN tag swapping, a VLAN tag must be unique in a link, and the same VLAN tag can be reused in the other links (link local labeling). Therefore, the scalability increases, and the restriction of the number of connections is virtually eliminated.

Figure B.4 shows the signaling sequence when the L2-LSP is established. There are 4 nodes, divided into 2 domains. Source routing is employed. First, Node A explicitly designates the path within Domain X and implicitly designates the path within Domain Y. A node checks availability of the VLAN tag of the in-

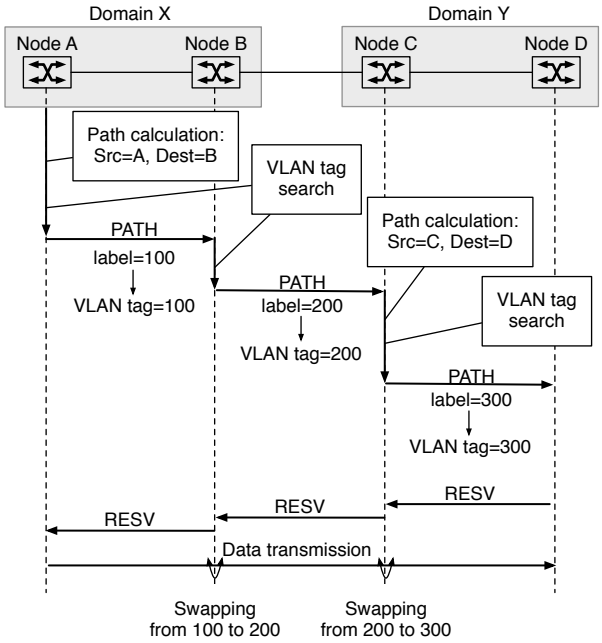


Figure B.4: Signaling sequence of L2-LSP establishment

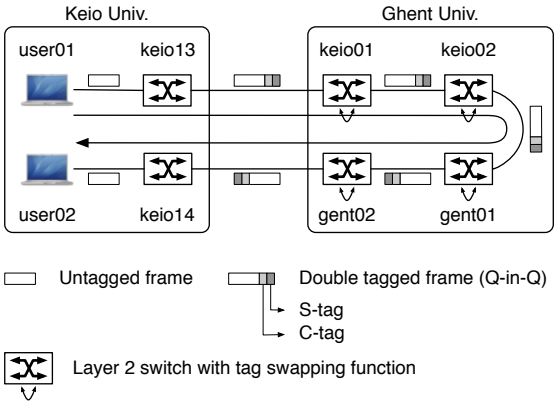


Figure B.5: Experimental setup of demonstration

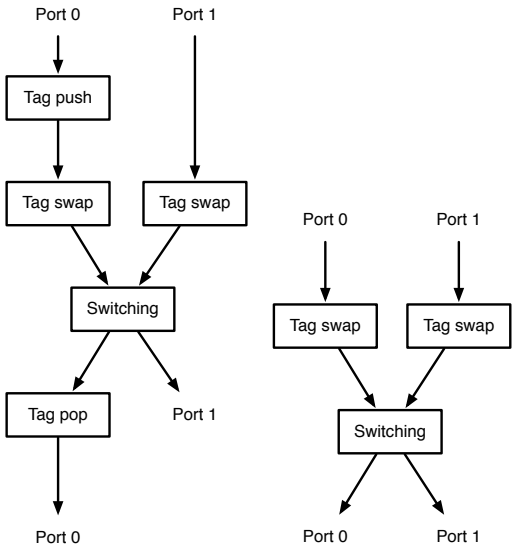
coming link and searches for an unused VLAN tag of the outgoing link when it receives a signaling message. Then, it manages a new mapping entry from label to VLAN tag. In the figure, Node B dynamically searches for an unused VLAN tag, then VLAN tag 200 is found as an unused VLAN tag. After all the entire procedure of establishing a new path is completed, data transmission can happen through the VLAN tag swapped path.

B.3 Experiments

B.3.1 Experimental Setup

Figure B.5 shows the experimental setup of the international interoperability experiments between Japan and Belgium. There are 6 Ethernet switches (keio01, keio02, keio13, keio14, gent01, and gent02) and 2 end users (user01 and user02). The switches are controlled and configured by GMPLS protocols, and they contain VLAN tag swapping functionality. The data plane of all switches is based on the Click Modular Router framework [5]. The data plane of keio01, keio02, keio13, and keio14 is developed by Keio University, and that of gent01 and gent02 is developed by Ghent University. Figure B.6(a) and Fig. B.6(b) show the schematic diagram of the configurations of Click Modular Router in keio13 and keio01, respectively.

Two switches, keio13 and keio14, are placed in Keio University, Tokyo, Japan, and the other 4 switches are placed in Ghent University, Ghent, Belgium. The switches in Japan work as edge switches and the switches in Belgium work as core switches. The ingress edge switch accepts untagged Ethernet frames and encapsulates them into Q-in-Q VLAN tagged frames for transmission to the core



(a) keio13 (b) keio01

Figure B.6: Click configuration

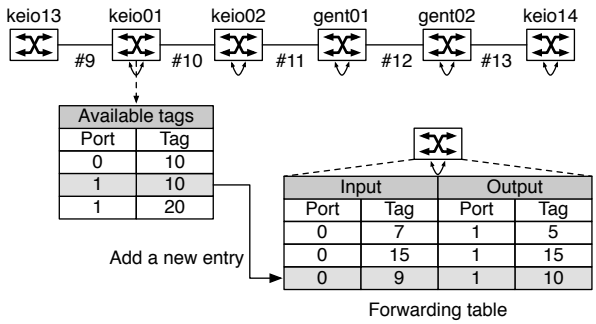


Figure B.7: Changing configurations of switch when signaling is occurred

switches as shown in Fig. B.6(a). Every core switch then swaps the outer VLAN (S-VID) to forward it towards its next hop. Finally the egress edge switch removes the Q-in-Q VLAN tag.

B.4 Path establishment

RSVP-TE establishes an L2-LSP between keio13 and keio14. Figure B.7 shows the established L2-LSP in the experiment. The number below a link describes the

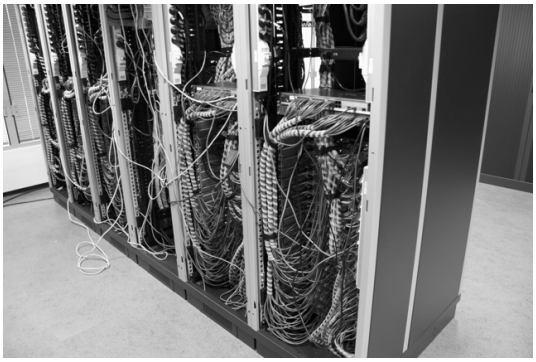


Figure B.8: 4 Core switches placed in the ilab.t testbed in Ghent University, Belgium

value of the VLAN tag assigned to the link. For example, the VLAN tag of the link between keio13 and keio01 is 9, and that of the link between keio01 and keio02 is 10. In this case, the VLAN tag is swapped from 9 to 10 at keio01. Each switch manages used tags and available tags.

When reserving a path, each switch is looking for an avail- able tag on the output port. In Fig. B.7, it is assumed that the left port is port 0, and the right port is port 1 for all nodes. The first available tag of port 1 is used for a new L2-LSP at keio01. After the tag is assigned for a new L2-LSP, the configuration of the forwarding tables occurs automatically. In this case, the shaded entry (Input port, Input tag, Output port, Output tag) = (0, 9, 1, 10) is added. Similarly, all the other switches are configured, and finally the L2-LSP is established between keio13 and keio14.

B.5 High definition video transmission and numerical results

A high definition video stream was transmitted through an L2-LSP, which is established by RSVP-TE. Figure B.9 shows the sender, the receiver of a high definition video stream, and the edge switches, which are placed in Keio University, Japan. The sender corresponds to user01 in Fig. B.5, and the receiver corresponds to user02 in Fig. B.5. The core switches are located the ilab.t testbed in Ghent University, Belgium as shown in Fig. B.8.

Figure B.11 shows the experiment of transmitting high definition video. The video stream was captured by a video camera attached to the sender, and the TV monitor is attached to the receiver. The high definition video stream from the camera is transmitted through the established L2-LSP. The video is successfully displayed on the TV monitor at high definition quality as shown in Fig B.11.

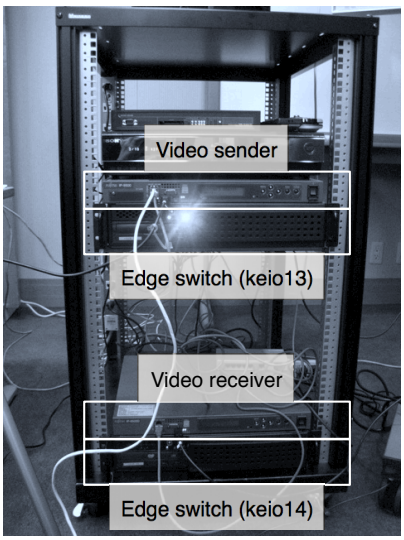


Figure B.9: High definition video sender, receiver, and 2 edge switches placed in Keio University, Japan

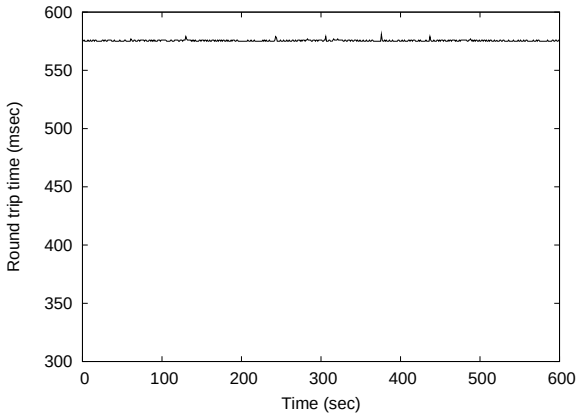


Figure B.10: Round trip time between user01 and user02

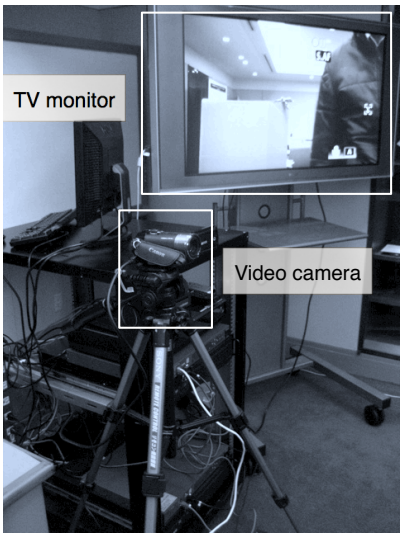


Figure B.11: High definition video captured by video camera is displayed on TV monitor

The round trip time and the UDP bandwidth between two users (user01 and user02) are measured. Figure B.10 shows the round trip time measured by ping for 10 minutes. An ICMP packet is sent every 1 second from user01. The average round trip time is 575.5 ms and the standard deviation is 0.64 ms. This result indicates that tag swapping does not affect stability of the round trip time. Figure B.12 shows the UDP throughput measured by iperf [6] for 10 minutes. user01 is the source node and the transmissions rate of the source node are 10Mbps, 15Mbps, and 20Mbps, respectively. The measured throughput is satisfactory for high definition video transmission.

The number of VLAN paths supported in the conventional system is at most 4096. On the other hand, our pro- posed system can support at least 4096 VLAN paths. The value varies with the number of hops of the paths through the network. 4096 VLAN tags can be used on each link in our proposed system, and there are 5 links in this experiment. The maximum number N of VLAN paths allowed in this experiment is expressed as follows:

$$\begin{aligned} N &= max(x), \text{ such that } \sum_{i=0}^{x} h_i \leq 4096 \times 5 \\ &\approx \frac{4096 \times 5}{h}, \end{aligned} \tag{B.1}$$

where i is the index number of a VLAN path, h_i is the number of hops of the VLAN path i, and h is the average number of hops of the VLAN paths. The

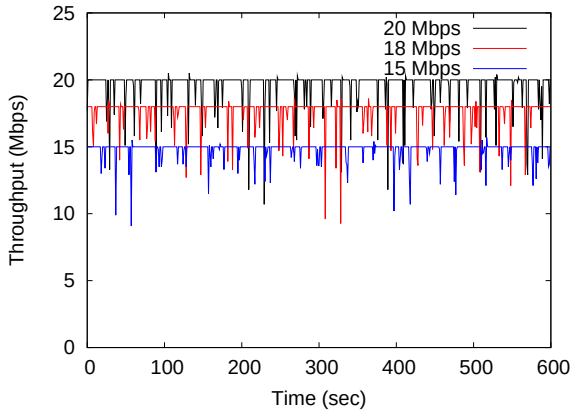


Figure B.12: UDP throughput between user01 and user02

VLAN tag search time is O(1) with the number of nodes n since a hash table is used for the forwarding table in our implementation.

From this experiment, we verified the interoperability between two different VLAN tag swapping based Ethernet implementations. We successfully transmitted high definition video stream between Japan and Belgium with Q-in- Q frames. This result proves the feasibility of VLAN tag swapped based Ethernet, and proves that wide area Ethernet is realistic.

B.6 Conclusion

The scalability of Ethernet is a key issue in wide area Ethernet. To cope with the issue, VLAN tag swapping is an effective solution. In this paper, we reported on an interoperability experiment between two different implementations of Ethernet VLAN swapping in Japan and Belgium. In the experiment, a GMPLS controlled L2-LSP was established between Japan and Belgium, and high definition video was transmitted through the L2-LSP. We successfully demonstrated the interoperability of VLAN tag swapping and Q- in-Q frame transmission between the two countries. This result confirmed that VLAN tag swapping is a good solution for wide area Ethernet.

B.7 Acknowledgment

This work is supported in part by a Grant-in-Aid for the Global Center of Excellence for high-Level Global Cooperation for Leading-Edge Platform on Access Spaces from the Ministry of Education, Culture, Sport, Science, and Technology

in Japan, Grant-in-Aid for Scientific Research (19360178), and ”Lambda Access Project” funded by the National Institute of Information and Communication Technology (NICT). One of the authors appreciate Yoshida Scholarship Foundation for its financial support.

References

[1] *IEEE 802.1ad. IEEE Standard for Provider Bridges*. IEEE.

[2] W. Tavernier, D. Papadimitriou, D. Colle, M. Pickavet, and P. Demeester. *Emulation of GMPLS-controlled ethernet label switching*. In Testbeds and Research Infrastructures for the Development of Networks & Communities and Workshops, 2009. TridentCom 2009. 5th International Conference on, pages 1–9. IEEE, 2009.

[3] E. Mannie. *Generalized Multi-Protocol Label Switching (GMPLS) Architecture*. RFC 3945 (Proposed Standard), October 2004. Updated by RFC 6002. Available from: http://www.ietf.org/rfc/rfc3945.txt.

[4] K. Kikuta, M. Nishida, D. Ishii, S. Okamoto, and N. Yamanaka. *Establishment of VLAN tag swapped path on GMPLS controlling wide area layer-2 network*. In Optical Fiber Communication-incudes post deadline papers, 2009. OFC 2009. Conference on, pages 1–3. IEEE, 2009.

[5] E. Kohler, R. Morris, B. Chen, J. Jannotti, and M. Kaashoek. *The Click modular router*. ACM Transactions on Computer Systems (TOCS), 18(3):263–297, 2000.

[6] A. Tirumala, F. Qin, J. Dugan, J. Ferguson, and K. Gibbs. *Iperf: The TCP/UDP bandwidth measurement tool*, 2005.