Universidade Estadual de Campinas
Faculdade de Engenharia Elétrica e de Computação
Departamento de Computação e Automação Industrial

Proposta de Roteamento Plano Baseado em uma Métrica de OU-Exclusivo e Visibilidade Local

Autor: Rafael Pasquini
Orientador: Prof. Dr. Maurício Ferreira Magalhães

Tese de Doutorado apresentada à Faculdade de Engenharia Elétrica e de Computação como parte dos requisitos para obtenção do título de Doutor em Engenharia Elétrica. Área de concentração: **Engenharia de Computação**.

Banca Examinadora

Prof. Dr. Maurício Ferreira Magalhães (Presidente) ... DCA/FEEC/Unicamp
Prof. Dr. Carlos Alberto Kamienski ..... UFABC
Prof. Dr. Pedro Frosi Rosa ..... FACOM/UFU
Prof. Dr. Edmundo Roberto Mauro Madeira ..... IC/Unicamp
Prof. Dr. Eleri Cardozo ..... DCA/FEEC/Unicamp

Campinas, SP
Junho/2011

FICHA CATALOGRÁFICA ELABORADA PELA
BIBLIOTECA DA ÁREA DE ENGENHARIA E ARQUITETURA - BAE - UNICAMP

Pasquini, Rafael
P265p Proposta de roteamento plano baseado em uma métrica de OU-Exclusivo e visibilidade local / Rafael Pasquini. – Campinas, SP: [s.n.], 2011.

Orientador: Maurício Ferreira Magalhães.
Tese de Doutorado - Universidade Estadual de Campinas, Faculdade de Engenharia Elétrica e de Computação.

1. Redes de computadores - Protocolos. 2. Internet. 3. Arquitetura de redes de computador. I. Magalhães, Maurício Ferreira. II. Universidade Estadual de Campinas. Faculdade de Engenharia Elétrica e de Computação. III. Título.

Título em Inglês:	A flat routing proposal based on the XOR metric and local visibility
Palavras-chave em Inglês:	Computer networks - Protocols, Internet, Architecture of computer networks
Área de concentração:	Engenharia de Computação
Titulação:	Doutor em Engenharia Elétrica
Banca Examinadora:	Carlos Alberto Kamienski, Pedro Frosi Rosa, Edmundo Roberto Mauro Madeira, Eleri Cardozo
Data da Defesa:	06.06.2011
Programa de Pós Graduação:	Engenharia Elétrica

COMISSÃO JULGADORA - TESE DE DOUTORADO

Candidato: Rafael Pasquini

Data da Defesa: 6 de junho de 2011

Título da Tese: "Proposta de Roteamento Plano Baseado em uma Métrica de OU-Exclusivo e Visibilidade Local"

Prof. Dr. Maurício Ferreira Magalhães (Presidente): _____

Prof. Dr. Carlos Alberto Kamienski: _____

Prof. Dr. Pedro Frosi Rosa: _____

Prof. Dr. Edmundo Roberto Mauro Madeira: _____

Prof. Dr. Eleri Cardozo: _____

Resumo

Roteamento é uma das principais funções em redes de computadores, sendo responsável pelo encaminhamento de tráfego entre todos os pares de nós de origem e destino. O princípio de roteamento comum a protocolos usados mundialmente requer a presença de informação sobre todos os destinos disponíveis, em todos os roteadores compondo a rede, de tal forma a garantir a entrega de tráfego. Desta forma, construir redes em grande escala usando tal princípio de roteamento é amplamente aceito como não escalável. O problema intrínseco destes mecanismos está relacionado ao fato de que as tabelas de roteamento acompanham o crescimento da informação de roteamento presente na rede.

Por outro lado, existem mecanismos de roteamento disponíveis na literatura que requerem apenas uma fração de toda a informação de roteamento presente na rede, provendo um melhor controle para a taxa à qual as tabelas de roteamento crescem. Neste cenário, um espaço de identificação plano é usado para identificar, univocamente, todos os nós presentes na rede e relações de vizinhança no espaço de identificação plano são estabelecidas através de uma rede sobreposta, construída sob um substrato de rede, como, por exemplo, uma rede IP. Entretanto, manter a coerência desta rede sobreposta é desafiador, uma vez que os nós podem mudar seus pontos de conexão no substrato de rede, resultando no uso de diferentes endereços (IPs) e requerendo mecanismos para manter ativa a associação entre a rede sobreposta e o substrato de rede.

Neste contexto, este trabalho propõe o uso de uma organização de rede alternativa, onde nenhum substrato de rede é necessário para prover a comunicação entre nós no espaço de identificação. O mecanismo de roteamento plano proposto é baseado em uma métrica de ou-exclusivo (XOR) e no conceito de visibilidade local. Tal combinação acarreta a criação de uma estrutura de rede em malha e promove a integração entre o espaço de identificação plano e a estrutura física da rede. Basicamente, o mecanismo de roteamento baseado em operações de XOR efetua roteamento diretamente sob identificadores planos.

Este trabalho apresenta a especificação completa do protocolo, descrevendo suas principais propriedades relacionadas ao tamanho das tabelas de roteamento, à quantidade de mensagens de sinalização necessárias para convergir o sistema de roteamento e os caminhos obtidos. Além disso, este trabalho detalha a instanciação do mecanismo de roteamento plano proposto em três diferentes cenários: 1) redes de *data centers*, 2) redes veiculares *ad hoc* (VANETs) e 3) o sistema de roteamento entre domínios da Internet.

Palavras-chave: Roteamento baseado em XOR, Roteamento Plano, Visibilidade Local.

Abstract

Routing is one of the main functions of computer networks, being responsible for traffic forwarding between all pairs of source and destination nodes. The common routing principle of protocols used in networks worldwide requires the presence of information about all available destinations, in all routers composing the network, in order to assure traffic delivery. In this way, building large scale networks using such routing principle is widely regarded as non scalable. The intrinsic problem of such mechanisms is related to the fact that routing tables follow the growth of the routing information present in the network.

Conversely, there are routing mechanisms available in the literature which require just a fraction of the overall routing information present in the network, providing a better control for the rate at which the routing tables grow. In such scenario, a flat identity space is used to uniquely refer to all nodes present in the network, and neighborhood relations at the flat identity space are established through an overlay network, built on top of a substrate network, such as an IP network. However, maintaining the correctness of the overlay network is challenging, since nodes can change their attachment points at the substrate network, resulting in the use of different addresses (IPs), and requiring mechanisms to keep the association between the overlay network and the substrate network active.

In this context, this work proposes the usage of an alternative network organization, where no substrate network is required to provide the communication between nodes at the flat identity space. The proposed flat routing mechanism is based on the bitwise exclusive or (XOR) metric and in the concept of *local visibility*. Such combination leads to the creation of a mesh network structure and promotes the integration between the flat identity space and the physical network structure. Basically, the proposed XOR-based routing mechanism performs routing directly on top of flat identifiers.

This work presents the entire protocol specification, describing its main properties related to the size of the required routing tables, the amount of signaling messages needed to converge the routing system, and the obtained paths. Afterwards, this work also details the instantiation of the proposed flat routing mechanism in three different scenarios: 1) data center networks, 2) *ad hoc* vehicular networks (VANETs) and 3) the inter-domain Internet routing system.

Keywords: Flat Routing, Local Visibility, XOR-based Routing.

Aos meus pais, Paulino e Maria Luiza

Agradecimentos

Ao meu orientador e co-orientador, Profs. Maurício Ferreira Magalhães e Fábio Luciano Verdi, sou grato pela orientação e dedicação durante todo o desenvolvimento deste trabalho. Espero continuarmos trabalhando em conjunto por muitos anos.

À pesquisadora Annikki Welin, sou grato por toda ajuda e acolhida, não somente no período que permaneci na Ericsson Research em Estocolmo, Suécia, mas durante todo o desenvolvimento deste trabalho.

Ao Prof. Rodolfo Oliveira da Universidade Nova de Lisboa, sou grato pela colaboração que tivemos nos últimos anos. Espero que esta colaboração esteja apenas em fase inicial.

A todos os colegas de pós-graduação do LCA, sou grato pelas críticas, sugestões e, principalmente, pelos momentos de descontração que tornaram a realização deste trabalho possível. Jamais esquecerei os momentos que passamos nestes anos. Será que um dia publicaremos algumas das inúmeras histórias? Ou será que foram todas estórias?

À minha família sou grato pelo apoio durante esta jornada, oferecendo sempre um refúgio. Tenho total convicção de que sem as incontáveis viagens para Andradas este trabalho não seria possível.

À Ericsson Research do Brasil, pelo apoio financeiro.

Finalmente, presto uma homenagem ao meu amigo Luis Fernando Carvalho Machado, o Alemão. A sua partida prematura deixou o mundo mais triste.

Sumário

Lista de Figuras

Lista de Tabelas

Siglas e Abreviaturas

A-STAR Anchor-based Street and Traffic Aware Routing, 69
AODV Ad hoc On-demand Distance Vector, 69
API Application Programming Interface, 129
ARIB Association of Radio Industries and Business, 69
AS Autonomous System, 11
ASV Advanced Safety Vehicle, 69

Bf Bloom filter, 85
BGL Broadcast Group Leader, 72
BGP Border Gateway Protocol, 11
BRITE Boston university Representative Internet Topology gEnerator, 82

CAIDA Cooperative Association for Internet Data Analysis, 18
CAR Connectivity-Aware Routing, 69
CDF Cumulative Distribution Function, 94
CEN Comité Européen de Normalisation, 69
CIDR Classless Inter-Domain Routing, 12

DC Data Center, 55
DFZ Default Free Zone, 13
DHT Distributed Hash Table, 12
DID Domain IDentifier, 100
DRAM Dynamic Random-Access Memory, 130
DSDV Destination-Sequenced Distance-Vector, 26
DSR Dynamic Source Routing, 69
DSSS Driving Safety Support System, 69

EC2 Elastic Compute Cloud, 62
ECMP Equal Cost Multi-Path, 65
ETSI European Telecommunications Standards Institute, 69

FQDN Full Qualified Domain Name, 102
FSR Fisheye State Routing, 69

GFS Google File System, 55
GPSR Greedy Perimeter Stateless Routing, 69
GSR Geographic Source Routing, 69

HIP Host Identity Protocol, 100

IAB Internet Architecture Board, 13
IBR Identity-Based Routing, 14
ID Identifier, 13
ID/Loc Identifier/Locator, 13
IEEE Institute of Electrical and Electronics Engineers, 69
IETF Internet Engineering Task Force, 13
IP Internet Protocol, 11
ISO International Organization for Standardization, 69
ITS Intelligent Transport Systems, 69

lcp longest common prefix, 30
LDP Location Discovery Protocol, 59
LIB Landmark Information Base, 84
LID Landmark IDentifier, 84
LISP Locator Identifier Separation Protocol, 81
LSH Locality Sensitive Hash, 108

MAC Media Access Control, 55
MPR MultiPoint Relay, 78

NAT Network Address Translator, 20
NIC Network Interface Card, 55
NID Node IDentifier, 100

OLSR Optimized Link State Routing, 25
OSPF Open Shortest Path First, 11

PI Provider Independent, 81

RIP Routing Information Protocol, 11
ROFL Routing on Flat Labels, 17

RR ROFL Ring, 27
RRG Routing Research Group, 13

SAE Society of Automotive Engineers, 69
SCOREF Système COopératif Routier Expérimental Français, 69
simTD Safe and Intelligent Mobility - Test fielD, 69
SRAM Static Random-Access Memory, 130
SUMO Simulation of Urban MObility, 76

TCAM Ternary Content-Addressable Memory, 130
TCP Transmission Control Protocol, 127
TORA Temporally-Ordered Routing Algorithm, 69

UIP Unmanaged Internet Protocol, 12

VANET Vehicular *ad hoc* Network, 15
VLAN Virtual Local Area Network, 16
VLB Valiant Load Balance, 65
VRR Virtual Ring Routing, 17

XOR eXclusive OR, 14

Produção Bibliográfica do Autor

Pedidos de Propriedade Intelectual:

1. Patente Internacional: *"Reducing routing tables with XOR and Landmarks"*, Inventores: R. Pasquini, F. L. Verdi, M. F. Magalhães e A. Welin, Solicitante: Ericsson AB, Data do Preenchimento: 15/10/2010. (em estudo de viabilidade)

Capítulos de Livro:

1. F. L. Verdi, C. E. Rothenberg, R. Pasquini e M. F. Magalhães. *"Novas Arquiteturas de Data Center para Cloud Computing"*. Publicado em: Mini Cursos do 28º Simpósio Brasileiro de Redes de Computadores e Sistemas Distribuídos - SBRC 2010, p. 103-152, Organizado por: C. A. Kamienski, L. P. Gaspary, M. P. Barcellos, ISBN: 9772177497006, Idioma: Português. Gramado - RS, Brasil. 24 - 28 de Maio de 2010.

2. R. Oliveira, A. Garrido, M. Luis, R. Pasquini, L. Bernardo, R. Dinis e P. Pinto. *"Performance Analysis of XOR-Based Routing Protocols in Vehicular ad hoc Networks"*. Publicado em: Internet Policies and Issues. New York, 2011, v. 8, Editora: Nova Publishers, Organizado por: B.G. Kutais, ISBN: 978-1-61122-840-3. Idioma: Inglês.

Trabalhos Publicados em Revistas:

1. R. Pasquini, R. Oliveira, F. L. Verdi, M. F. Magalhães e A. Welin. *"Towards Local Routing State in the Internet"*. Submetido para a IEEE Communications Letters em 16/02/2011. (submetido)

Trabalhos Publicados em Anais de Congressos:

1. R. Pasquini, F. L. Verdi e M. F. Magalhães. *"Integrating Servers and Networking using an XOR-based Flat Routing Mechanism in 3-cube Server-centric Data Centers"*. 29º Simpósio Brasileiro de Redes de Computadores e Sistemas Distribuídos - SBRC 2011. Campo Grande - MS, Brasil. 30 de Maio - 3 de Junho de 2011.

2. R. Oliveira, A. Garrido, R. Pasquini, M. Luís, L. Bernardo, R. Dinis e P. Pinto. *"Towards the use of XOR-based Routing Protocols in Vehicular ad hoc Networks"*. 73º IEEE Vehicular Technology Conference - VTC 2011. Budapeste, Hungria. 15 - 18 de Maio de 2011.

3. R. Pasquini, F. L. Verdi, R. Oliveira, M. F. Magalhães e A. Welin. *"A Proposal for an XOR-based Flat Routing Mechanism in Internet-like Topologies"*. IEEE Global Communications Conference - GLOBECOM 2010. Miami, Flórida, Estado Unidos. 6 - 10 de Dezembro de 2010.

4. R. Pasquini, F. L. Verdi, M. F. Magalhães e A. Welin. *"Bloom filters in a Landmark-based Flat Routing"*. IEEE International Conference on Communications - ICC 2010. Cidade do Cabo, África do Sul. 23 - 27 de Maio de 2010.

5. R. Pasquini, F. L. Verdi e M. F. Magalhães. *"Towards a Landmark-based Flat Routing"*. 27º Simpósio Brasileiro de Redes de Computadores e Sistemas Distribuídos - SBRC 2009. Recife - PE, Brasil. 25 - 29 de Maio de 2009.

6. R. Pasquini, L. B. de Paula, F. L. Verdi e M. F. Magalhães. *"Domain Identifiers in a Next Generation Internet Architecture"*. IEEE Wireless Communications & Networking Conference - WCNC 2009. Budapeste, Hungria. 5 - 8 de Abril de 2009.

7. W. Wong, R. Villaca, L. Paula, R. Pasquini, F. L. Verdi e M. F. Magalhães. *"An Architecture for Mobility Support in a Next Generation Internet"*. The 22nd IEEE International Conference on Advanced Information, Networking and Applications - AINA 2008. Okinawa, Japão. 25 - 28 de Março de 2008.

8. W. Wong, R. Pasquini, R. Villaça, L. de Paula, F. L. Verdi e M. F. Magalhães. *"A Framework for Mobility and Flat Addressing in Heterogeneous Domains"*. 25º Simpósio Brasileiro de Redes de Computadores e Sistemas Distribuídos - SBRC 2007. Belém - PA, Brasil. 28 de Maio - 1 de Junho de 2007.

9. M. Siqueira, F. L. Verdi, R. Pasquini e M. F. Magalhães. *"An Architecture for Autonomic Management of Ambient Networks"*. Intelligent Communications - Autonomic management and services 2006 - IntellComm 2006. Paris, França. 25 - 29 de Setembro de 2006.

Resumos expandidos publicados em *Workshops*:

1. F. L. Verdi, R. Pasquini and M. F. Magalhães. *"Flat Routing in Internet-like Topologies"*. I Workshop de Pesquisa Experimental na Internet do Futuro - WPEIF 2010, evento paralelo ao SBRC 2010. Gramado - RS. 24 - 28 de Maio de 2010.

2. R. Pasquini e M. F. Magalhães. *"Flat Routing on a Binary Identity Space"*. Segundo Encontro dos Alunos e Docentes do Departamento de Engenharia de Computação e Automação Industrial - II EADCA. FEEC/Unicamp. 26 - 27 de Março de 2009.

3. R. Pasquini e M. F. Magalhães. *"Roteamento Flat para uma Arquitetura de Internet de Próxima Geração baseada em Identificadores de Domínios"*. Primeiro Encontro dos Alunos e Docentes do Departamento de Engenharia de Computação e Automação Industrial - I EADCA. FEEC/Unicamp. 12 - 13 de Março de 2008.

Capítulo 1

Introdução

Roteamento é uma das principais funções em redes de computadores, sendo responsável pelo encaminhamento de tráfego entre pares de nós de origem/destino. Normalmente, a estrutura de roteamento de uma rede é composta por um conjunto de roteadores, os quais utilizando um protocolo de roteamento efetuam a troca de informações relativas aos destinos disponíveis na rede para gerarem as tabelas de roteamento. Baseando-se nestas tabelas, o tráfego pode ser encaminhado entre os nós, sendo também responsabilidade do protocolo de roteamento manter as tabelas de roteamento atualizadas, representando a condição mais recente da rede após alterações em sua estrutura, de tal forma a garantir a entrega de tráfego.

Os protocolos de roteamento em operação na maioria das redes estão organizados em três classes: 1) estado do enlace, 2) vetor de distância e 3) vetor de caminho. Exemplos clássicos de protocolos em cada uma destas classes incluem o protocolo de estado do enlace OSPF (*Open Shortest Path First*) [2], o protocolo de vetor de distância RIP (*Routing Information Protocol*) [3] e o protocolo de vetor de caminhos BGP (*Border Gateway Protocol*) [4]. Basicamente, o princípio de roteamento dos protocolos mencionados requer que roteadores compondo a rede tenham informação relativa a todos os destinos disponíveis, de tal forma a garantir a correta entrega de tráfego.

Desta maneira, construir redes em grande escala utilizando tal princípio de roteamento é amplamente aceito como não escalável [2, 5]. Por exemplo, a utilização do OSPF em sistemas autônomos (ASs) de larga escala requer uma estrutura de rede hierárquica para se tornar escalável. O AS como um todo é dividido em regiões menores denominadas áreas OSPF, que são conectadas através de uma região central chamada de núcleo. Neste cenário, roteadores possuem apenas o mapa topológico de sua própria área, disseminando uma versão resumida da topologia no núcleo. Consequentemente, mudanças que ocorram no AS são isoladas dentro das áreas, não perturbando o AS como um todo com as disseminações de atualizações.

Um outro exemplo onde a escalabilidade é alcançada usando-se uma organização hierárquica

é o atual mecanismo de roteamento entre domínios da Internet baseado no IP. Originalmente, o espaço de endereçamento IP era alocado usando o conceito de classes IP [6], sendo o sistema de roteamento entre domínios inicial da Internet considerado uma solução de roteamento plano, no qual a escalabilidade não era problema devido ao reduzido número de informações de roteamento existentes (as classes IP alocadas). Entretanto, com a popularização da Internet no início dos anos 90, o espaço de endereçamento IP tornou-se escasso devido à alocação ineficiente de IPs resultante da utilização das classes IP fixas.

Historicamente, a Internet foi projetada para operar em um cenário composto por um conjunto reduzido de redes [7, 8], provendo a comunicação entre um número controlável de dispositivos. Contudo, o cenário real enfrentado pela Internet mostrou-se totalmente contrário, atingindo o número de cinco bilhões de dispositivos conectados à rede em aproximadamente sete anos após sua popularização [9]. Dessa forma, tal crescimento explosivo da rede levou o sistema ineficiente de alocação de endereços a quase colapsar, causando uma escassez prematura de endereços IP e forçando a introdução de *patches* no mecanismo de roteamento da Internet para amenizar tal problema [10, 11, 12].

De maneira simplista, a principal alternativa foi a proposta do CIDR (*Classless Inter-Domain Routing*) [10], que removeu o conceito de classes IP fixas, introduzindo o conceito de prefixos IP de tamanho variado. Consequentemente, a alocação de endereços IP tornou-se mais eficiente uma vez que as redes passaram a receber blocos de IPs mais apropriados às suas necessidades reais, evitando o desperdício de endereços. Entretanto, o mecanismo de roteamento entre domínios que era originalmente plano tornou-se hierárquico, criando um cenário onde uma organização de rede em formato de árvore era utilizada para propiciar uma agregação eficiente de endereços IP em prefixos IP. Basicamente, a agregação era essencial para a escalabilidade em nível mundial, uma vez que o mecanismo de roteamento baseado no CIDR continuava a requerer informação sobre todos os destinos (conhecimento global) presentes na rede para garantir a entrega de tráfego. O problema intrínseco deste cenário de conhecimento global é o fato de as tabelas de roteamento acompanharem o crescimento da informação de roteamento presente na rede.

Por outro lado, existem mecanismos de roteamento disponíveis na literatura que requerem apenas uma fração da informação de roteamento presente na rede. A principal característica desses mecanismos de roteamento está relacionada às melhores taxas de crescimento das tabelas de roteamento, uma vez que as tabelas não acompanham a quantidade de informação de roteamento disponível na rede. Exemplos de mecanismos de roteamento incluídos nesta classe são principalmente soluções de DHT (*Distributed Hash Table*), tais como o Chord [13], Pastry [14], Tapestry [15], Kademlia [16] e o UIP (*Unmanaged Internet Protocol*) [17, 18], normalmente utilizados em comunicações *peer-to-peer*. Nestes cenários, um espaço de identificação plano é utilizado para

univocamente referir-se aos nós presentes na rede, estabelecendo relações de vizinhança no espaço de identificação plano através de uma rede sobreposta. Esta rede sobreposta é construída sob um substrato de rede (por exemplo, uma rede IP) que provê a comunicação direta entre nós na rede sobreposta.

O principal requisito destas propostas é garantir a correta organização da rede sobreposta, permitindo que os nós estabeleçam as relações de vizinhança com os nós adequados. Basicamente, inconsistências introduzidas na rede sobreposta afetam a entrega de tráfego. Entretanto, manter a correta organização da rede sobreposta pode ser desafiador, uma vez que os nós podem mudar seus pontos de conexão na rede de substrato, acarretando mudanças em seus endereços atuais (IPs) e requisitando mecanismos para manter ativa a associação entre a rede sobreposta e a rede de substrato. Além disso, os vizinhos necessários na rede sobreposta podem ser nós que estão fisicamente distantes na rede de substrato. Desta forma, as relações estabelecidas no espaço de identificação plano destas propostas são totalmente desacopladas da estrutura física da rede.

Neste contexto, este trabalho propõe um mecanismo de roteamento plano que visa a integração do espaço de identificação plano com a estrutura física da rede, apresentando um mecanismo de roteamento plano que roteia utilizando puramente identificadores planos, ou seja, não há a necessidade de se utilizar uma rede de substrato para encaminhar tráfego entre nós no cenário proposto. Consequentemente, o mecanismo proposto gera suas tabelas de roteamento considerando as relações que os nós possuem no espaço de identificação plano e as relações que os nós possuem na estrutura física da rede, introduzindo o aspecto de localidade para espaço de identificação plano. Este aspecto de localidade do mecanismo proposto é um conceito denominado *visibilidade local* neste trabalho, sendo relacionado com a habilidade que o mecanismo de roteamento plano proposto possui para entregar tráfego sem um conhecimento global da rede.

1.1 Motivação

Em 2006, o RRG (*Routing Research Group*) do IETF organizou uma reunião do IAB (*Internet Architecture Board*) com o objetivo de investigar e apontar os principais fatores causadores dos problemas de escalabilidade do mecanismo de roteamento entre domínios atual da Internet. Como resultado desta reunião, o IETF publicou um relatório [5] no qual um conjunto de importantes questões foram levantadas, tendo as questões relacionadas à sobrecarga semântica do IP recebido uma atenção especial. Fundamentalmente, o IP desempenha duas funções na arquitetura de roteamento atual da Internet, atuando como identificador e localizador dos nós e tornando difícil a utilização de novas demandas relacionadas à mobilidade, *nodes renumbering* e *multi-homing* [19, 20].

Por exemplo, o uso de *multi-homing* compromete a agregação de endereços IP em prefixos IP,

afetando a operação do CIDR e causando um crescimento acelerado das tabelas de roteamento presentes na região central da Internet (denominada DFZ - *Default Free Zone*). A taxa de crescimento das tabelas de roteamento da DFZ é apresentada na Figura 1.1. Basicamente, tal crescimento está estressando a estrutura da DFZ devido à necessidade de conhecimento global adotada pelo mecanismo de roteamento da Internet, que força as tabelas de roteamento a seguirem o crescimento da rede.

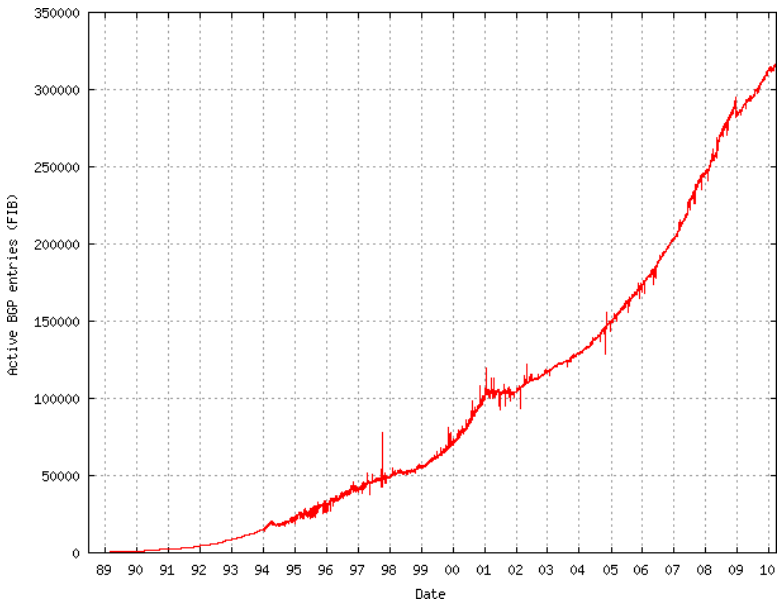


Fig. 1.1: Curva representando o crescimento das tabelas de roteamento do BGP que está comprometendo a DFZ da Internet. Informação extraída do *BGP reports* [1].

Desta maneira, o RRG definiu um conjunto de possíveis requisitos para uma arquitetura de roteamento futura para a Internet [21], motivando a pesquisa de novos paradigmas de roteamento. Neste contexto, um conjunto de novas propostas emergiram com o objetivo de resolver a sobrecarga semântica do IP através da separação entre identificadores e localizadores. Entretanto, estas propostas geralmente efetuam a separação de identificadores e localizadores inserindo uma nova camada na pilha de protocolos dos nós finais [22, 23, 24, 25, 26, 27], restringindo a camada IP atual à função de localizador. Consequentemente, tal abordagem baseia-se na existência de novos sistemas de mapeamento, nos quais informações relativas aos identificadores planos dos nós estão associadas aos seus respectivos localizadores.

Sendo assim, o gargalo atualmente localizado no sistema de roteamento IP é aliviado, sendo totalmente transferido para os novos sistemas de mapeamento. Basicamente, a estrutura IP torna-se fixa, recuperando sua forte agregação de IPs e, consequentemente, reduzindo o tamanho das tabelas de roteamento. Ao mesmo tempo, torna-se responsabilidade dos sistemas de mapeamento prover informações corretas sobre a localização atual de todos os nós na Internet, ou seja, os sistemas de

mapeamento precisam ser projetados para suportar condições dinâmicas, uma vez que o cenário previsto inclui novas demandas tais como mobilidade, evitar a ocorrência de *nodes renumbering* e *multi-homing*.

Como alternativa aos desafios impostos pela manutenção dos sistemas de mapeamento [28], o trabalho recente do IBR (*Identity-Based Routing*) [29, 30, 31] dá um passo à frente e remove a necessidade por localizadores, propondo efetuar roteamento diretamente sob identificadores planos. O IBR organiza a rede em uma estrutura de anel virtual, na qual nós estabelecem relações de vizinhança com nós sucessores e predecessores de forma a propiciar o encaminhamento de tráfego. Entretanto, embora o IBR ofereça melhores níveis de escalabilidade relacionados à quantidade de informações presentes nas tabelas de roteamento, é preciso considerar questões ligadas a distribuição randômica de identificadores planos aos nós e, também, questões ligadas ao dinamismo da rede. Essencialmente, essas questões causam um espalhamento da informação de roteamento necessária aos nós em toda a rede, tornando difícil a criação da estrutura de anel virtual em cenários de larga escala. Além disso, o dinamismo da rede pode ocasionar o particionamento do anel virtual, impedindo o encaminhamento de tráfego e exigindo mecanismos auxiliares para unificar o anel virtual particionado.

Nesse cenário, este trabalho propõe o uso de uma organização de rede alternativa, na qual relações entre os nós são estabelecidas através de uma métrica de ou-exclusivo (XOR), levando à criação de uma estrutura de rede em malha que contribui para resolver problemas inerentes ao anel virtual. Em resumo, este trabalho está alinhado com os pontos levantados pelo RRG do IETF em [5], considerando essencial a separação entre identificadores e localizadores como forma de suportar as novas demandas arquiteturais de um futuro mecanismo de roteamento, porém opta pelo paradigma de roteamento introduzido pelo IBR, no qual localizadores são totalmente removidos do sistema de roteamento. Nós consideramos que os custos associados à manutenção dos sistemas de mapeamento comprometem a escalabilidade do sistema como um todo.

1.2 Contribuições desta Tese

A principal contribuição deste trabalho é o mecanismo de roteamento plano proposto, baseado em operações de XOR, que é capaz de efetuar encaminhamento de tráfego usando puramente identificadores planos, evitando o uso de um substrato de rede formada por localizadores. Consequentemente, o mecanismo de roteamento proposto elimina o uso de sistemas de mapeamento entre identificadores planos e seus respectivos localizadores. Conforme mencionado anteriormente, o mecanismo proposto organiza a rede em uma estrutura em malha, ao contrário da abordagem de anel virtual do IBR, estendendo propostas disponíveis na literatura [16, 17, 18], usadas para

criar redes sobrepostas. Este trabalho herda a organização das tabelas de roteamento e a função de roteamento disponíveis na literatura, ambas baseadas na operação de XOR, propondo um novo sistema de roteamento que propicia a integração entre o espaço de identificação plano e a estrutura física da rede.

Desta forma, uma importante contribuição deste trabalho é o mecanismo proposto para criar as tabelas de roteamento, que possibilita criar as tabelas baseadas em XOR usando o conceito proposto de *visibilidade local*. O mecanismo é totalmente dinâmico e distribuído, apresentando uma solução factível de ser implementada em redes reais, ao contrário dos modelos centralizados e teóricos investigados nas pesquisas de *compact routing* [32, 33]. Este trabalho apresenta a especificação completa do protocolo, incluindo detalhes sobre a sinalização usada para gerar as tabelas de roteamento.

O conceito proposto de *visibilidade local* tem o objetivo de integrar o espaço de identificação plano com a estrutura física da rede, provendo a base para o desenvolvimento de um mecanismo de roteamento ciente sobre a condição real da rede. Neste contexto, o mecanismo para criar as tabelas de roteamento é projetado para priorizar a inserção de vizinhos fisicamente próximos (em números de saltos), criando um cenário no qual a convergência do sistema de roteamento torna-se mais simples, evitando a disseminação de mensagens de sinalização através de toda a rede.

Além da integração entre o espaço de identificação plano e a estrutura física da rede que é oferecida pelo mecanismo proposto, ele também apresenta propriedades interessantes relacionadas ao tamanho das tabelas de roteamento, à quantidade de mensagens de sinalização necessárias para convergir o sistema de roteamento e à qualidade dos caminhos obtidos através da rede. Consequentemente, o mecanismo de roteamento proposto foi instanciado em três cenários diferentes: 1) redes de *data center*, 2) redes veiculares *ad hoc* (VANETs) e 3) o sistema de roteamento entre domínios da Internet.

No cenário de redes de *data center*, o principal desafio é tratar a enorme quantidade de servidores presentes na rede. Normalmente, propostas disponíveis na literatura adotam soluções baseadas no uso de VLANs, tunelamento e/ou rota na origem, criando um cenário no qual os servidores ficam totalmente isolados da infraestrutura de rede para obter escalabilidade. Por outro lado, até onde sabemos, o mecanismo proposto é a primeira solução de roteamento plano que totalmente integra os servidores localizados dentro do *data center* com a estrutura de rede, utilizando uma distribuição randômica de identificadores planos e propiciando a criação de tabelas de roteamento escaláveis contendo informação sobre os servidores compondo o *data center*.

O trabalho detalha a arquitetura de *data center* desenvolvida para extrair máximos benefícios do mecanismo de roteamento plano baseado em XOR. Os servidores são organizados, basicamente, em uma topologia baseada em cubo, na qual a distância física (em números de saltos) entre os servidores

é reduzida devido aos enlaces estabelecidos entre servidores. Essa característica permite aproveitar o conceito de *visibilidade local* proposto. Geralmente, propostas baseadas em cubo [34, 35, 36] necessitam de esquemas de endereçamento rígidos nos quais a posição dos nós está totalmente representada em seus endereços, exigindo mecanismos complexos para atribuir de forma correta tais endereços. A solução proposta nesta tese apenas requer unicidade na atribuição dos identificadores planos aos servidores, criando um cenário no qual uma distribuição totalmente randômica de identificadores é ideal para a instanciação de mecanismo de roteamento plano, simplificando o desenvolvimento de redes de *data centers*.

No cenário de redes veiculares *ad hoc*, o principal desafio é tratar o elevado nível de dinamicidade ocasionado pela frequente mudança na posição dos veículos em estradas e/ou ruas. Propostas disponíveis na literatura baseiam-se no conhecimento global, exigindo que os nós presentes na rede possuam informação sobre a topologia inteira da rede e/ou a posição atual dos nós disponíveis na rede [37, 38, 39]. Desta forma, manter esse conhecimento global sobre a informação de roteamento requer constantes trocas de sinalização, comprometendo a usabilidade destas soluções. Neste sentido, o aspecto de *visibilidade local* do mecanismo de roteamento plano baseado em XOR constitui uma alternativa interessante, na qual os nós priorizam a comunicação com nós fisicamente próximos de tal forma a encaminhar tráfego.

Duas versões diferentes do mecanismo de roteamento proposto são apresentadas para o cenário das VANETs, a primeira versão utiliza o mecanismo baseado em XOR proposto e a segunda versão estende o mecanismo para operar em conjunto com uma proposta que identifica conexões estáveis na rede [40, 41, 42]. Todo o trabalho em VANETs foi desenvolvido em conjunto com o Prof. Dr. Rodolfo Oliveira da Universidade Nova de Lisboa, Portugal, e encontra-se em fase inicial de desenvolvimento. Basicamente, resultados iniciais obtidos através de simulações são apresentados neste trabalho.

No cenário entre domínios da Internet, o objetivo é contribuir com os problemas de escalabilidade relacionados ao crescimento explosivo das tabelas de roteamento compondo a DFZ da Internet e, também, relacionados com a sobrecarga de sinalização resultante de mudanças na rede que causam disseminações frequentes de atualizações do BGP. Basicamente, o mecanismo de roteamento atualmente em uso na Internet não é capaz de beneficiar-se de propriedades intrínsecas da topologia *Power-law* da Internet. Conforme amplamente discutido na literatura [43, 44, 45], topologias *Power-law* oferecem um conjunto de propriedades ideais para o roteamento, contribuindo para melhorar o desempenho dos mecanismos de roteamento. Sendo assim, as investigações feitas no cenário da Internet neste trabalho são motivadas pela atual pesquisa relacionada aos conceitos de *small world* e navegabilidade de redes complexas [46, 47], nas quais a influência da estrutura física da rede sobre o nível de eficiência obtido pelo mecanismo de roteamento é investigada.

A instanciação do mecanismo de roteamento plano baseado em XOR no cenário de roteamento

entre domínios da Internet tem o objetivo de prover o máximo de entrega de tráfego possível, priorizando a escalabilidade do sistema como um todo em termos de sinalização e tamanho das tabelas de roteamento. A idéia é desenvolver um mecanismo de roteamento no qual a entrega de tráfego ocorre devido à integração entre a solução de roteamento e a infraestrutura de rede. Neste contexto, uma outra contribuição para o cenário Internet é a separação entre conectividade física e alcançabilidade, propondo a criação de um serviço de conectividade e um de alcançabilidade.

No serviço de conectividade proposto, ASs possuem total liberdade para comprar sua conectividade física de acordo com suas preferências, por exemplo, a partir de ASs que ofereçam melhores custos e/ou melhores recursos de rede. Uma vez que a solução proposta baseia-se em um espaço de identificação plano, não há a necessidade de usar a estrutura hierárquica atual da rede para garantir a atribuição topológica de endereços, provendo um natural suporte a novas demandas como *multi-homing*, evitar *nodes renumbering* e mobilidade.

O *serviço de alcançabilidade*, por sua vez, é um serviço complementar à solução de roteamento plano baseada em XOR, oferecendo um mecanismo para garantir alcançabilidade em escala mundial para os ASs, uma vez que no cenário proposto o mecanismo XOR não é responsável por garantir 100% da entrega de tráfego. O *serviço de alcançabilidade* cria um novo mercado na Internet, no qual redes portadoras (possivelmente ASs *tier 1* apresentando cobertura mundial), podem oferecer tal serviço como uma alternativa ao mercado atual de transporte de tráfego em longa distância. Sendo assim, o mecanismo de roteamento baseado em XOR garante um percentual de toda a navegabilidade da rede, priorizando uma convergência escalável do sistema de roteamento, e o restante da navegabilidade é obtida usando o *serviço de alcançabilidade* proposto.

Por último, foi desenvolvida uma ferramenta de emulação que contém uma implementação completa do mecanismo de roteamento plano baseado em XOR. Nesta ferramenta, *threads* independentes são instanciadas para agir como nós individuais, sendo capazes de trocar as mensagens de sinalização necessárias, construindo suas próprias tabelas de roteamento e encaminhando tráfego entre os nós de acordo com a especificação do protocolo. A ferramenta foi utilizada nos cenários investigados de redes de *data center* e roteamento entre domínios da Internet, tendo o mecanismo baseado em XOR recebido as extensões necessárias para operar em cada cenário em específico.

1.3 Estrutura da Tese

O Capítulo 2 detalha o IBR [29], apresentando suas instanciações no VRR (*Virtual Ring Routing*) [30] e ROFL (*Routing on Flat Labels*) [31], com o intuito de introduzir o conceito de roteamento efetuado diretamente sobre identificadores planos. Na sequência, este capítulo introduz o UIP (*Unmanaged Internet Protocol*) [17, 18], que emprega operações de XOR para entregar tráfego em

um cenário de redes sobrepostas. O cenário XOR do UIP é a base para a estrutura de rede em malha proposta neste trabalho como uma alternativa à estrutura de anel virtual do IBR.

O Capítulo 3 detalha o mecanismo de roteamento plano baseado em XOR proposto neste trabalho, descrevendo o princípio de roteamento baseado em XOR, como construir as tabelas de roteamento e como encaminhar tráfego. Além disso, este capítulo introduz o conceito de *visibilidade local*, detalhando como a utilização deste conceito contribui para a convergência do sistema como um todo e como este conceito está relacionado ao encaminhamento de tráfego.

O Capítulo 4 apresenta a primeira instanciação do mecanismo de roteamento proposto em um cenário de redes de *data center*, descrevendo a arquitetura baseada em cubo que contribui para a eficácia do conceito de *visibilidade local* proposto e ajuda na construção das tabelas de roteamento baseadas em XOR.

O Capítulo 5 traz a segunda instanciação do mecanismo proposto no cenário de redes veiculares *ad hoc* (VANETs). Resultados obtidos utilizando propostas disponíveis na literatura indicam que embora nós possuam informação sobre toda a topologia, eles são incapazes de efetuar a entrega de tráfego em 100% dos casos. Neste contexto, o aspecto de *visibilidade local* do mecanismo proposto introduz um cenário alternativo, no qual os resultados inicialmente obtidos apresentam um atraso fim-a-fim e uma taxa de entrega de pacotes adequados.

O Capítulo 6 descreve a terceira instanciação do mecanismo proposto no cenário de roteamento entre domínios da Internet. Este capítulo detalha o uso do mecanismo baseado em XOR efetuando encaminhamento de tráfego entre domínios, considerando os atuais identificadores de ASs como identificadores planos. O capítulo também detalha o uso do mecanismo baseado em XOR em conjunto com o *serviço de alcançabilidade* desenvolvido usando nós denominados *Landmarks* [48]. Além disso, o capítulo brevemente descreve uma proposta de arquitetura de Internet do futuro e avalia o cenário proposto usando a topologia entre domínios real da Internet, disponível no CAIDA [49] e composta por aproximadamente 33.000 ASs.

O Capítulo 7 conclui este trabalho, destacando as principais contribuições e os resultados obtidos, apontando questões importantes deixadas para serem investigadas em trabalhos futuros.

Finalmente, as avaliações efetuas nos Capítulos 4 e 6 foram efetuadas utilizando uma ferramenta de emulação que desenvolvemos. Esta ferramenta é detalhada no Apêndice A.

Introduction

Routing is one of the main functions of computer networks, being responsible for traffic forwarding between the entire set of source/destination pairs of nodes. Normally, the routing structure of a network is composed of a set of routers, which using a routing protocol exchange information regarding the destinations available in the network to build the routing tables. Based on such routing tables, traffic can be forwarded between nodes, and it is also responsibility of the routing protocol to keep the routing tables up-to-date, representing the most recent network condition after changes in its structure, in order to assure traffic delivery.

The routing protocols in operation in the vast majority of the networks worldwide are organized in three classes: 1) link-state, 2) distance-vector and 3) path-vector. Classical examples of protocols in each one of the three classes include the link-state Open Shortest Path First (OSPF) [2] protocol, the distance-vector Routing Information Protocol (RIP) [3] and the path-vector Border Gateway Protocol (BGP) [4]. Basically, the routing principle of the above mentioned protocols requires that routers composing the network have information about all the destinations available in order to assure the correct traffic delivery.

In this way, building large scale networks using such routing principle is widely regarded as non scalable [2, 5]. For example, the usage of OSPF in large scale autonomous systems (ASes) requires one hierarchical network organization to become scalable. The entire AS is divided in smaller regions called OSPF areas, connected through a central region called backbone. In this scenario, routers only have the entire topology map of their own area, and disseminate a resumed version of the topology in the backbone. Consequently, changes in the AS are isolated inside areas, not disturbing the entire AS with the dissemination of updates.

Another example where scalability is achieved using an hierarchical organization is the current IP-based inter-domain Internet routing mechanism. Originally, the IP address space was allocated using the concept of IP classes [6], and the initial inter-domain Internet routing system was considered a flat routing solution, where the scalability was not a problem due to the reduced number of existent routing information (the allocated IP classes). However, with the popularization of the Internet in the early 90's, the IP address space became scarce due to the inefficient allocation of IPs resultant of the

fixed IP classes.

Historically, the Internet was designed to operate in a scenario composed of a small set of networks [7, 8], providing communication between a controllable number of devices. Nonetheless, the real scenario faced by the Internet was totally contrary, achieving the number of five billion devices connected to the network in roughly seven years after its popularization [9]. In this way, such explosive growth led the inefficient address allocation system to almost collapse, causing a premature lack of IP addresses, and forcing the introduction of patches in the Internet routing mechanism to amend such problem [10, 11, 12].

In a simplistic way, the main alternative was the Classless Inter-Domain Routing (CIDR) [10] proposal, which removed the concept of fixed IP classes, introducing the concept of variable size IP prefixes. Consequently, the allocation of IP addresses became more efficient because networks started to receive blocks of IPs more appropriate to their real needs, avoiding the waste of addresses. However, the original flat inter-domain Internet routing mechanism became hierarchical, creating a scenario where a network tree organization was used to allow an efficient aggregation of IP addresses into IP prefixes. Basically, the aggregation was essential for worldwide scalability, since the CIDR-based routing mechanism still required information about all the destinations (global knowledge) present in the network in order to assure traffic delivery. The intrinsic problem of this global knowledge scenario is the fact that routing tables follow the growth of the routing information present in the network.

Conversely, there are routing mechanisms available in the literature which require just a fraction of the overall routing information present in the network. The main characteristic of such routing mechanisms is related to their better control for the rate at which the routing tables grow, once it does not follow the amount of routing information available in the network. Examples of routing mechanisms included in this class are mainly DHT (Distributed Hash Table) solutions, such as Chord [13], Pastry [14], Tapestry [15], Kademlia [16] and UIP (Unmanaged Internet Protocol) [17, 18], normally used in peer-to-peer communications. In such scenarios, a flat identity space is used to uniquely refer to nodes present in the network, and neighborhood relations at the flat identity space are established through an overlay network, built on top of a substrate network providing the direct communication between overlay nodes, such as an IP network.

The main requirement of these proposals is to assure the overlay network correctness, with all nodes establishing neighborhood relations with the proper nodes. Basically, inconsistencies introduced in the overlay network affect the traffic delivery. However, maintaining the correctness of the overlay network structure may be challenging, since nodes can change their attachment points at the substrate network, resulting in changes on their current addresses (IPs), and requiring mechanisms to keep active the association between the overlay network and the substrate network. Afterwards,

the required neighbors at the overlay network can be nodes that are physically distant in the underlay substrate. Hence, the relations established at the flat identity space of such proposals are oblivious to the physical network structure.

In this context, this work proposes a flat routing mechanism aimed at integrating the flat identity space with the physical network structure, presenting a flat routing mechanism which purely routes using flat IDs, i.e., there is no need to use a substrate network to forward traffic between nodes. Consequently, the proposed mechanism generates its routing tables not only considering the relations that nodes have at the flat identity space, but also considering the relations that nodes have at the physical network structure, introducing the aspect of locality to the (location-free by nature) flat identity space. Such locality aspect of the proposed flat routing mechanism is a concept called *local visibility* in this work, and it is related to the ability of the proposed flat routing mechanism to deliver traffic without a global knowledge about the network.

Motivation

In 2006, the IETF Routing Research Group (RRG) organized a meeting of its Internet Architecture Board (IAB) aimed at investigating and pointing the main factors causing scalability problems in the current inter-domain Internet routing mechanism. As a result of this meeting, the IETF published a report [5] in which a set of important questions were raised, having the questions related to the IP semantics overload received a special attention. Fundamentally, the IP has two functions in the current Internet routing architecture, acting as identifier and locator of nodes, making difficult the use of new demands related to mobility, nodes renumbering and multi-homing [19, 20].

For example, the use of multi-homing dissolutes the aggregation of IP addresses into IP prefixes, compromising the operation of CIDR and leading to an accelerated growth of the routing tables present in the core region of Internet (called as Default Free Zone - DFZ). The growing rate of the DFZ routing tables is depicted in Figure 1.2. Basically, such growth is stressing the DFZ structure due to the global knowledge approach adopted in the Internet routing mechanism, which forces the routing tables to follow the network growth.

In this way, the RRG defined a set of possibles requirements for a future Internet routing architecture [21], motivating the research on new routing paradigms. In this context, a set of new proposals emerged aimed at solving the IP semantics overload through the separation between identifiers and locators. However, these proposals usually perform the ID/Loc separation by inserting a new identity layer at the protocol stack of end hosts [22, 23, 24, 25, 26, 27], and restricting the current IP layer to the function of locators. Consequently, such approach relies in the existence of new mapping systems, where information regarding the flat identifiers of nodes are associated to their

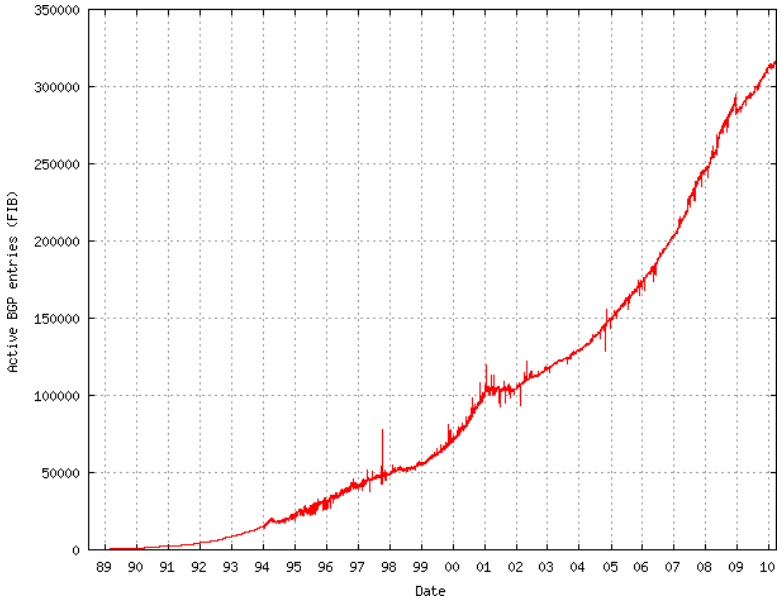


Figure 1.2: Curve representing the growth of the BGP routing tables composing the Internet DFZ. Extracted from the BGP reports [1].

respective locators.

As a consequence, the bottleneck currently located in the IP routing system is alleviated, but it is entirely transfered to the new mapping systems. Basically, the IP structure become fixed, recovering the strong IP aggregation and, consequently, reducing the size of the routing tables. At the same time, it becomes responsibility of the mapping systems to provide the correct information about the current location of all nodes in the Internet. So, the mapping systems must be designed to support dynamic conditions, since the envisioned scenario considers new demands such as mobility, avoidance of nodes renumbering and multi-homing.

As an alternative to the challenges imposed by the maintenance of the mapping systems [28], the recent Identity-Based Routing (IBR) work [29, 30, 31] moves a step further and removes the need for locators, proposing to perform routing directly on top of flat identifiers. The IBR organizes the network in a virtual ring structure, where nodes establish neighborhood relations with successor and predecessor nodes to allow traffic forwarding. However, although it offers better scalability levels related to the amount of information present in the routing tables, it is necessary to consider questions related to the randomness for assigning the flat identifiers to nodes and, also, questions related to the dynamism of the network. Essentially, such questions spread the required routing information through the entire network, making difficult the creation of the virtual ring structure in large scale scenarios. Afterwards, the dynamism of the network can lead to the partition of the virtual ring, avoiding the traffic forwarding and requiring auxiliary mechanisms to merge the broken virtual ring.

In this context, this work proposes the usage of an alternative network organization, where the relation between nodes are established using the bitwise exclusive or (XOR) metric, leading to the creation of a mesh network structure which helps to solve the inherent problems of the virtual ring. In short, this work is aligned with the points raised by the IETF RRG in [5], considering the ID/Loc separation essential to support the new architectural demands of a future routing mechanism, but opts to the routing paradigm introduced in IBR, where locators are totally removed from the routing system. We consider that the costs associated to the maintenance of the mapping systems compromise the overall system scalability.

Contributions of this Thesis

The main contribution of this work is the proposed XOR-based flat routing mechanism capable of performing traffic forwarding using purely flat identifiers, avoiding the usage of a substrate network of locators. Consequently, the proposed routing mechanism eliminates the use of mapping systems to keep track of flat identifiers and their respective locators. As mentioned before, the proposed flat routing mechanism organizes the network in a mesh structure, as opposed to the virtual ring approach of IBR, extending other proposals available in the literature [16, 17, 18] used to create overlay networks. This work inherits the routing tables organization and the routing function available in the literature, both based in the XOR operation, proposing a new routing system which propitiates the integration between the flat identity space and the physical network structure.

In this way, one important contribution of this work is the mechanism for building the routing tables, which allows the creation of the XOR-based routing tables under the concept of *local visibility*. The proposed mechanism is totally dynamic and distributed, introducing a mechanism whose implementation in real networks is feasible, as opposed to the centralized and theoretical models investigated in the compact routing research [32, 33]. This work presents the entire protocol specification, including details about the signaling used to generate the routing tables.

The concept of *local visibility* aims to integrate the flat identity space with the physical network structure, providing the fundamental basis for the development of a routing mechanism aware about the real network condition. In this context, the mechanism for building the routing tables is designed to prioritize the insertion of neighbor nodes physically near (in number of hops), creating a scenario where the convergence of the routing system becomes simpler, avoiding the dissemination of signaling messages across the entire network.

Besides the integration between the flat identity space and the physical network structure offered by the proposed routing mechanism, it presents interesting properties related to the size of the routing tables, the amount of signaling messages needed to converge the routing system, and the quality of

the obtained paths across the network. Consequently, the flat routing mechanism was instantiated in three different scenarios: 1) data center networks, 2) vehicular *ad hoc* networks (VANETs) and 3) the inter-domain Internet routing system.

In the data center scenario, the main challenge is to handle the huge amount of servers present in the network. Normally, proposals available in the literature adopt solutions like VLANs, tunneling and/or source routing, creating a scenario where the servers are totally isolated from the network infrastructure to achieve scalability. The XOR-based flat routing mechanism, as far as we know, is the first flat routing solution which totally integrates the servers inside the data center with the network structure using a random distribution of flat IDs, propitiating the creation of scalable routing tables using information regarding the flat identifiers of servers composing the data center.

This work also details the proposed data center architecture, developed to extract the maximum benefits from the XOR-based flat routing mechanism. Basically, servers in the architecture are organized in a cube-based topology, where the physical distance (in number of hops) between servers is reduced due to the established links in order to leverage the *local visibility* concept. Usually, cube-based proposals [34, 35, 36] rely on rigid addressing schemes where the position of nodes are fully represented on their addresses, requiring complex mechanisms to correctly assign such addresses. On the other hand, the proposed solution only requires uniqueness to assign the flat identifiers to servers, creating a scenario where a total random distribution of the identifiers is ideal to the instantiation of the flat routing mechanism, simplifying the deployment of data center networks.

In the *ad hoc* vehicular network scenario, the main challenge is to handle the high level of dynamism resultant of the frequent change of the vehicles' position in the road and/or streets. Proposals available in the literature rely on the global knowledge approach, requiring that nodes present in the network have information regarding the entire network topology and/or the current position of nodes available in the network [37, 38, 39]. Hence, keeping such global knowledge information requires frequent signaling exchange, compromising the usability of such solutions. In this way, the *local visibility* aspect of the XOR-based flat routing mechanism constitutes an interesting alternative, where nodes prioritize the insertion of physically near nodes in the routing tables in order to forward traffic.

Two different versions of the routing mechanism are presented for the VANETs' scenario, the first version purely relies on the XOR-based flat routing mechanism, and the second version extends the XOR-based mechanism to operate in conjunction with a proposal to identify stable connections [40, 41, 42] in the network. The entire VANET work was developed in conjunction with Prof. Dr. Rodolfo Oliveira of Universidade Nova de Lisboa, Portugal, and it is still in the initial development phase. Basically, initial results obtained through simulations are presented in this work.

In the inter-domain Internet scenario, the objective is to contribute with the scalability problems

related to the explosive growth of the routing tables composing the Internet DFZ, and also related to the signaling overhead resultant of changes in the network, which leads to the frequent dissemination of BGP updates. Basically, the current routing mechanism adopted in the Internet does not benefit from the intrinsic properties of the Power-law topology of the Internet. As widely discussed in the literature [43, 44, 45], Power-law topologies offer a set of properties ideal for routing, contributing to increase the performance of the routing mechanisms. In this way, the investigations performed for the Internet scenario in this work are motivated by the current research related to both small world and navigability of complex networks concepts [46, 47], where the influence of the network structure over the level of efficiency achieved by the routing mechanism is investigated.

The instantiation of the XOR-based flat routing mechanism in the inter-domain Internet routing scenario is aimed at providing the maximum traffic delivery as possible, prioritizing the overall system scalability in terms of signaling and routing tables' size. The rationale is to develop a routing mechanism where traffic delivery occurs due to the integration between the routing solution and the network infrastructure. In this context, another contribution for the Internet scenario is the decoupling between physical connectivity and reachability, proposing the creation of a connectivity and a *reachability service*.

In the connectivity service, ASes are totally free to purchase their physical connectivity according to their preferences, for example, from ASes offering better costs and/or better network resources. As the proposed solution relies on a flat identity space, there is no need to use the current hierarchical network structure in order to assure correct topological addresses assignment, providing natural support for new demands such as multi-homing, avoidance of nodes renumbering and mobility.

The *reachability service*, in its turn, is a complementary service to the XOR-based flat routing solution, offering a mechanism to assure worldwide reachability to ASes, since in the proposed scenario the XOR mechanism is not responsible for assuring 100% traffic delivery. The *reachability service* creates a new business in the Internet, where carriers (possible tier 1 ASes with networks presenting worldwide coverage), can offer such service as an alternative to the current long distance traffic forwarding business. In this way, the XOR-based routing mechanism assures a percentage of the overall network navigability, prioritizing a scalable routing system convergence, and the remainder network navigability is achieved using the proposed *reachability service*.

Finally, we developed an emulation tool including an entire implementation of the proposed XOR-based flat routing mechanism. In this tool, independent threads are instantiated to act as individual nodes, being able to exchange the required signaling messages, building their own routing tables and forwarding traffic between nodes according to the protocol specification. The tool was used in the investigated scenarios of data centers and inter-domain Internet routing, with the XOR-based mechanism receiving the required extensions to operate in each specific case.

Thesis Structure

Chapter 2 details the IBR [29] proposal, and its instantiations in the Virtual Ring Routing (VRR) [30] and Routing on Flat Labels (ROFL) [31], aimed at introducing the concept of routing directly on top of flat identifiers. In the sequence, it introduces the Unmanaged Internet Protocol (UIP) [17, 18], which uses the XOR operation to deliver traffic in an overlay network scenario. The XOR scenario of the UIP proposal is the basis for the mesh network structure proposed in this work as an alternative to the virtual ring structure of IBR.

Chapter 3 details the XOR-based flat routing mechanism proposed in this work, describing the XOR-based routing principle, how to build the routing tables, and how to forward traffic. Afterwards, this chapter introduces the concept of *local visibility*, detailing how the usage of such concept can contribute to the overall system convergence, and how it is related to traffic forwarding.

Chapter 4 presents the first instantiation of the XOR-based flat routing mechanism in the data center scenario, describing the cube-based architecture, which contributes for the effectiveness of the *local visibility* concept proposed and helps to create the XOR-based routing tables.

Chapter 5 brings the second instantiation of the routing mechanism in the *ad hoc* vehicular network (VANET) scenario. Results obtained using proposals available in the literature show that although nodes pose information regarding the entire topology, they are unable to deliver packets in 100% of the communication cases. In this context, the *local visibility* aspect of the XOR-based routing mechanism introduces an alternative scenario, where the initial results present adequate end-to-end delay and packet delivery ratio.

Chapter 6 describes the third instantiation of the mechanism in the inter-domain Internet routing system. This chapter details the usage of the XOR-based flat routing mechanism for inter-domain traffic forwarding, considering the current AS IDs as flat identifiers. This chapter details the usage of the XOR-based flat routing mechanism in conjunction with the *reachability service*, which is developed using nodes called Landmarks [48]. The chapter briefly describes a future Internet architecture proposal, and evaluates the proposed scenario using the real inter-domain Internet topology available at CAIDA [49], composed of approximately 33,000 ASes.

Chapter 7 concludes this work, highlighting the main contributions and the obtained results, pointing some important open issues left to be investigated as future work.

Finally, the evaluations performed in Chapters 4 and 6 were performed using the developed emulation tool, which is detailed in Appendix A.

Chapter 2

Related Work

Routing scalability is one of the main challenges in large-scale networks such as the Internet, being a frequent subject on computer networks research. Currently, practical routing solutions including link-state, distance-vector and path-vector protocols [3, 2, 4] are able to offer shortest paths, but at the cost of maintaining $\Omega(x)$ routing information in the network elements of a network of size x, and requiring elevated signaling overhead to generate and maintain the routing tables, compromising their operation in large-scale scenarios.

In general, the main solution used to provide scalability in large-scale networks relies on hierarchical network structures, where aggregation-based mechanisms reduce the amount of information present in the routing tables. The main characteristic of such hierarchical scenario is the use of name-dependent addresses [50, 33, 51], where information regarding the location of nodes is embedded on the addresses assigned to all nodes. Nevertheless, the use of name-dependent addressing schemes compromise the support for mobility and multi-homing.

As opposed to the name-dependent scenario, several routing mechanisms propose performing routing using name-independent flat names [52, 32, 53], adopting stable node identifiers at the network layer to serve the needs of the application layer. Benefits of such scenario include natural support for mobility, multi-homing and better security management, once such flat names are usually self-certifying identifiers permanently assigned to nodes. However, even tough the name-independent proposals build their routing tables using stable flat identifiers, the vast majority of the name-independent proposals available in the literature require the use of mapping mechanisms to translate flat identifiers into locators, which indicate the current position of nodes [24, 18, 22, 23], i.e., the flat identifiers are not effectively used to transport traffic across the network. At the same time, there are other name-independent proposals which rely on special addressing schemes to forward the initial packets through a structured network [54, 55], allowing source nodes to discover information (a routing hint) regarding the current position of the destination nodes, in order to allow the delivery

of subsequent packets using the flat network structure.

In this context, one important contribution in the name-independent routing research was introduced by IBR (Identity-Based Routing) [29, 30, 31], which is pioneer in performing routing using purely flat names, i.e., IBR requires neither mapping systems to translate from identifiers to locators, nor structured networks to deliver the first packets. As mentioned before, IBR relies on a virtual ring network structure to coordinate the establishment of neighborhood relations and to perform traffic forwarding. However, the virtual ring structure presents challenges related to the overall ring correctness, and the occurrence of failures in the network may lead to the virtual ring partition.

In this way, this work leverages the XOR-based mechanism used in UIP (Unmanaged Internet Protocol) [17, 18], where XOR operations are used to build an overlay network aimed at providing communication through network discontinues (such as firewall and NAT). Essentially, this work extends the UIP proposal in order to create a mesh network structure where flat routing is performed directly on top of flat identifiers. As opposed to the virtual ring structure, the XOR-based mesh network approach offers mechanisms to fully support the proposed *local visibility* concept, allowing traffic forwarding in a scenario where the creation of routing tables prioritizes the insertion of physically near nodes.

The remainder of this chapter is organized as follows: Section 2.1 details the IBR, briefly presenting its instantiations in the *ad hoc* sensor networks and in the Internet scenario, called as VRR (Virtual Ring Routing) and ROFL (Routing on Flat Labels), respectively. Section 2.2 presents UIP, detailing the overlay network scenario where the XOR-based mechanism is deployed. Section 2.3 summarizes this chapter.

2.1 Identity-Based Routing

This section details IBR, pioneer on the concept of routing directly on top of flat identifiers, and referred as the first scalable flat routing mechanism [30]. The entire protocol description contained in this section, including the exemplification figures, was extracted from [29, 30, 31]. In short, IBR organizes nodes into a virtual ring structure, where the position of a certain node in the ring is determined by its flat identifier. In this context, IBR assures that every node can make progress towards all the destinations available in the virtual ring if pointers to immediately adjacent (successor and predecessor) nodes are correctly maintained. Such traffic forwarding is based on numerical progress towards the destination node identifiers, and is purely performed on top of the flat identifiers, i.e., there is no underlay network structure providing such communication.

According to the IBR scenario presented in Figure 2.1, there are three main information

maintained at each node: 1) the node's identifier, 2) a collection of virtual pointers (vset) to nodes adjacent in the identity space and 3) forwarding pointers used to forward traffic between nodes connected by a virtual pointer. Regarding the node's identifier, IBR assumes that each node has a b-bit globally-unique numeric identity. In Figure 2.1 the identifiers are depicted using the hexadecimal notation for $b = 12$.

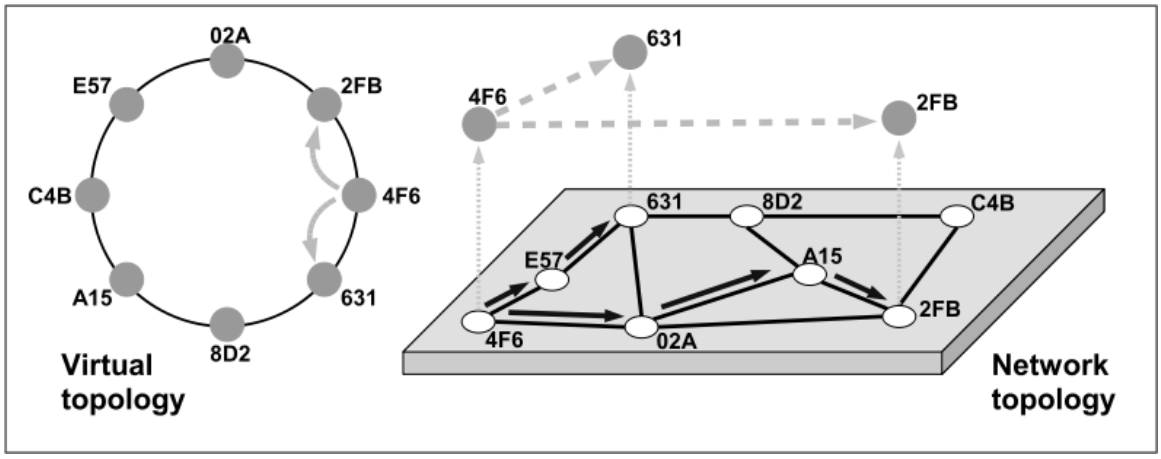


Figure 2.1: Virtual and network-level topologies.

Note in the network topology shown in the right side of Figure 2.1 that node identifiers are unrelated to their position in the physical network. Conversely, the identifiers are ordered in the virtual (ring) topology shown in the left side of the figure. Each node x is responsible for maintaining virtual pointers (vset) to r virtual neighbors, where $r/2$ are successors and $r/2$ are predecessors. Basically, each virtual pointer consists of a pair of endpoints, where x is one endpoint and the virtual neighbor pointed by x is the other endpoint. In Figure 2.1, node $4F6$ maintains virtual pointers to nodes 631 (successor) and $2FB$ (predecessor).

Afterwards, each node x maintains a path vector for each virtual pointer, which corresponds to a sequence of hops, originating at x and terminating at x's virtual neighbor. Instead of storing the list of hops locally, each hop on the list maintains a forwarding pointer to the next hop in the list, and by traversing this sequence of forwarding pointers, x can forward traffic to its virtual neighbors. Figure 2.2 exemplifies the forwarding table maintained at node 631, where several pieces of information are maintained for each pointer, including the pointer endpoints, the next hops used to reach a given endpoint, and a path identifier used to uniquely identify the path.

Based on such information, nodes are able to forward traffic across the virtual ring structure. In this way, a given source node s sends a packet to the virtual pointer p that is numerically closest to the destination node d, i.e., s computes the numeric distance along the ring between d and each of its virtual pointers, and selects the virtual pointer with smallest numeric distance. By doing this, s

Forwarding table at node 631

Endpoint	Next-hop	Endpoint	Next-hop	Path-id
E57	E57	C4B	8D2	1
631	631	4F6	E57	1
631	631	8D2	8D2	2
02A	02A	E57	E57	1
4F6	02A	2FB	8D2	1

Figure 2.2: Forwarding table.

maximizes progress through the name space. At each intermediate hop between s and p, forwarding pointers are used to make progress towards p, and when p receives the packet, if it is not the final destination, it repeats the process by looking up its virtual pointer closest to d. An example is shown in Figure 2.3, where node 631 sends a packet to node $2FB$.

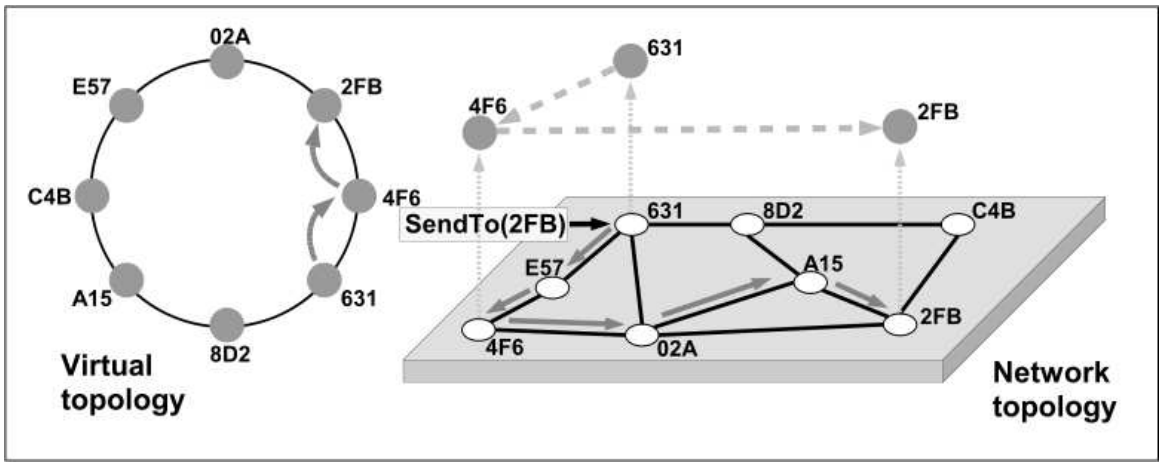


Figure 2.3: Example: forwarding a packet.

Basically, node 631 does not have a path vector to $2FB$ in its forwarding table, and hence cannot forward the packet directly. However, 631 poses a virtual pointer to node $4F6$ (its predecessor), which is closer in the name space to the destination $2FB$. Another option is 631 forward the packet to $8D2$ (its successor), but the numeric progress is smaller in this case. Consequently, when $4F6$ receives the packet, it performs a similar procedure to locate the next virtual hop that maximizes progress towards the final destination, and as $4F6$ has a pointer to $2FB$, it forwards the packet directly to $2FB$.

One important benefit of performing routing directly on top of flat identifiers is the occurrence of path optimizations. Figure 2.4 presents a communication case between nodes $4F6$ and $C4B$, where the shortcutting optimization of IBR occurs. First, node $4F6$ selects its virtual neighbor

that maximizes progress along the ring, which is node $2FB$, forwarding the packet towards node $2FB$. Ordinarily, the packet would traverse the path $(4F6, 02A, 2FB)$. However, $02A$ maintains a virtual pointer to $E57$, which is closer in the name space to the final destination $C4B$ than is $2FB$. Hence when the packet reaches $02A$, instead of naively forwarding the packet towards $2FB$, $02A$ will forward the packet towards $E57$. Afterwards, the shortcutting is performed again before the packet reaches node $E57$. In particular, note that $E57$ maintains a path to its virtual neighbor $C4B$, which traverses $E57 - 631 - 8D2 - C4B$. Consequently, 631 deviates the packet towards node $C4B$.

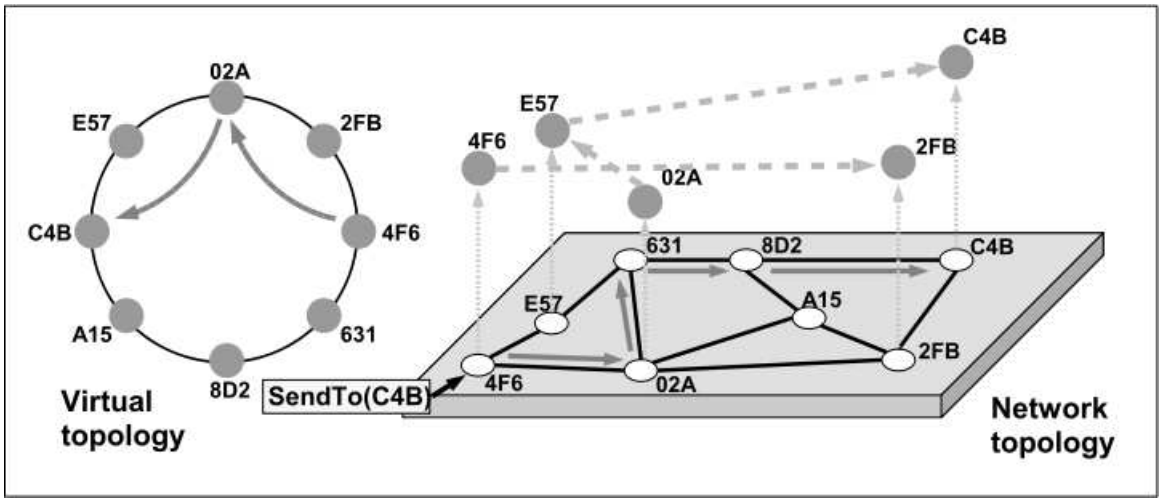


Figure 2.4: Example: forwarding a packet using the shortcutting optimization.

After describing the state maintained by each node in IBR, and how to route traffic using such state, the next step is to describe how to insert new nodes in the virtual ring. There are two main steps. First, the new node needs to discover its virtual neighbors. Next, it must build path vectors to ensure it can directly reach each of its virtual neighbors. The key challenge in performing these tasks is that the joining node J cannot use the IBR routing process, since J has not yet joined the ring. Consequently, J relies in its physical neighbor R, using R as a proxy to send and receive packets. In particular, J joins by using R to forward messages of join request destined to J's identifier. Since J does not yet exist in the network, the join message will be delivered to J's predecessor P in the virtual ring. When P receives the join request, it constructs a set containing its own identifiers, and the identifiers of all its virtual neighbors, and returns this set back to J via R. In the sequence, J can determine its virtual neighbors (vset), selecting the $r/2$ closest neighbors clockwise (successors) and $r/2$ counter clockwise (predecessors).

After such process, J is aware about the identifiers of its virtual neighbors but has no way to forward packets to them. Hence, J builds paths to each of its virtual neighbors using path setup messages. Each network-level hop that the path setup message traverses adds an entry to its

forwarding table regarding the new path being established. An example is given in Figure 2.5, where node $1C5$ wants to join the virtual ring.

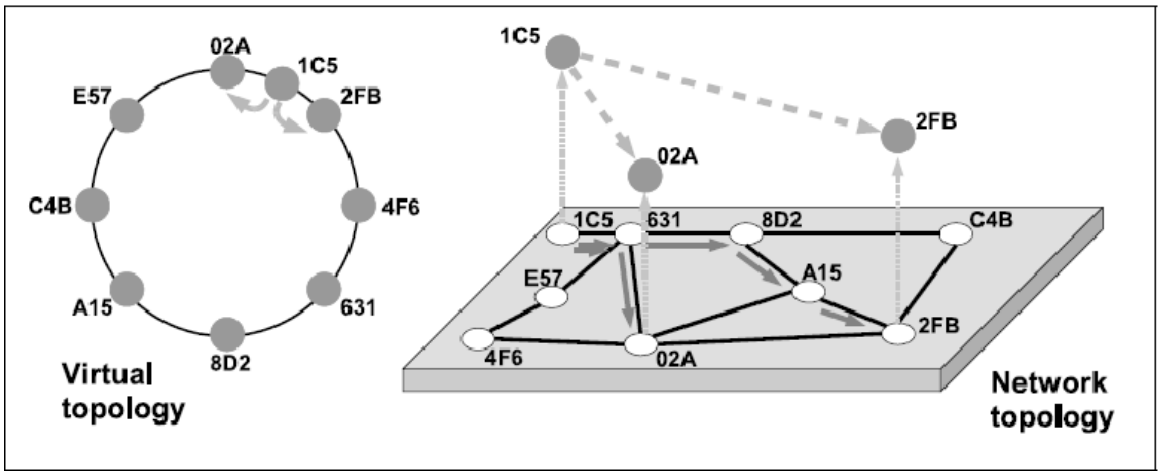


Figure 2.5: Examples: a new node joins the network.

The joining process starts with node $1C5$ sending a path setup message with the destination set to its own identifier. Since initially $1C5$ has no virtual neighbors, it forwards the message using node 631 as a proxy. The message is routed using normal IBR-style forwarding until it reaches $1C5$'s predecessor $02A$. In the sequence, $02A$ constructs the set $E57, 02A, 2FB$ containing its own identifier and the identifiers of its virtual neighbors, and sends the set back to $1C5$ using 631 as proxy. Consequently, $1C5$ selects $02A, 2FB$ as its virtual neighbors, and sends a path setup message to $2FB$ (its successor). As the message is forwarded, each intermediate hop adds a forwarding table entry pointing to the next hop along the path. Such process is then repeated to build a path between $1C5$ and its virtual neighbor $02A$ (its predecessor).

Regarding the virtual ring maintenance, IBR runs a ring maintenance protocol aimed at assuring that each node x eventually converges to point to its $r/2$ successor and $r/2$ predecessor nodes in the ring. At a first glance, this may seem like a simple problem. However, without the maintenance protocol, the ring may converge to an incorrect state during churn. Moreover, it is worth noting that previous work on ensuring consistency of DHTs does not address the IBR problem, even though IBR is inspired in DHTs. Basically, in a traditional DHT, a variety of network layer failure modes are masked by IP, whereas IBR is exposed to and must deal directly with them. Figure 2.6 presents two common problems that can occur in the absence of the ring maintenance protocol.

The first problem is the creation of loop cycles resultant of the join process. As can be seen in Figure 2.6(a), each individual node is correctly ordered between its predecessor and successor nodes. However, several nodes do not have their correct global successors, like node $02A$. Afterwards, network level partitions can cause the virtual ring topology to break in multiple rings, as exemplified

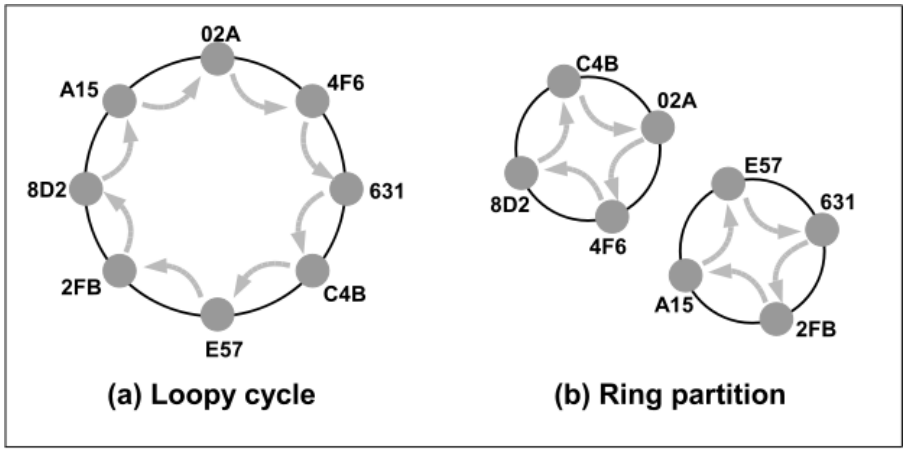


Figure 2.6: Examples: ring mis-convergence.

in Figure 2.6(b). The IBR approach for maintaining the correctness of the virtual ring leverages the FloodMin algorithm [56], used to determine the vertex with the smallest identifier present in a given graph $G = (V, E)$. Such node with the smallest ID is elected as the representative node for each ring partition created after failures. In short, every vertex maintains a record of the minimum ID observed so far, which is initialized on startup to the vertex's own identifier. During each round of the FloodMin algorithm, the minimum identifier is propagated to each of its neighbors. Considering a network of diameter d and composed of n nodes, after d rounds (and nd messages) the records at all nodes are equal to the minimum ID in the system.

This section is not aimed at describing the IBR maintenance mechanism in details, but it is important to remark that achieving the global correctness of the virtual ring after failures and/or churn is essential for traffic forwarding, since problems in the virtual ring organization avoid packets forwarding. Basically, the virtual ring structure does not support operating under the *local visibility* concept proposed in this work, since it would involve the existence of some nodes in the virtual ring without successor and/or predecessor nodes, in order to prioritize the insertion of physically near nodes in the forwarding tables. In short, operating IBR under such *local visibility* scenario leads to the occurrence of ring partitions. Finally, IBR was instantiated in two scenarios, and the main modifications required in each specific scenario are briefly presented in the next sections.

2.1.1 Virtual Ring Routing

VRR extends the IBR protocol to operate in the context of *ad hoc* wireless networks. The primary goals of VRR are to quickly recover from outages with minimal churn, and to forward packets with low probability of loss. In this way, VRR extends IBR in three ways:

1. Asymmetric link detection: Correct operation of IBR depends heavily on reliable communication. Wireless networks are particularly vulnerable to message loss, and wireless anomalies such as asymmetric links[1]. To deal with this, VRR leverages a failure detection scheme based on the neighbor discovery procedure present in OLSR [57], which handles this problem by having each node propagating its physical neighbors in hello messages. If node n observes a message from physical neighbor p that does not contain n's identifier, n concludes np is an asymmetric link.

2. Link estimation: In wireless networks, the presence or absence of a link in the topology is not a binary notion, as the quality of communication channels can vary significantly over time and location. However, VRR treats connectivity as a binary relation. To deal with this, VRR uses the link-estimation scheme described in [58], which computes the probability of successful communication by observing the loss rate of probes between physical neighbors. Basically, if such probability is lower than a threshold, VRR removes the link from the graph, and do not allow it to be used for communication.

3. Representative selection: Wireless networks are particularly prone to partitioning and high churn. Hence in these networks it is critical that ring maintenance perform quickly and efficiently. To reduce control overhead, VRR modifies the ring maintenance protocol of IBR to piggyback routes on hello packets destined to the representative nodes[2] selected using the FloodMin algorithm [56]. Essentially, in order to reduce overhead during partitions, representatives wait for a timeout to expire before triggering ring recovery. To reduce update overhead, VRR propagates path costs to representatives rather than the entire path. Finally, to reduce sensitivity to transient loops, VRR propagates a destination sequence in a manner similar to DSDV [59].

2.1.2 Routing on Flat Labels

ROFL is an instantiation of the IBR protocol to operate in the Internet scenario. Basically, it extends the virtual ring structure of IBR to operate in a hierarchical DHT structure similar to Canon [60], leveraging the concept of Autonomous Systems (ASes) by establishing virtual pointers with successor and predecessor nodes located internally and externally of ASes, as shown in Figure 2.7. In this way, ROFL considers routing in both intra-domain and inter-domain levels.

[1] An asymmetric link comprises the case where a given node n has a physical neighbor node p, but node p does not have node n as its physical neighbor.

[2] A representative node is the node with the smallest numerical ID present in each ring partition, which is responsible for the merging process coordination.

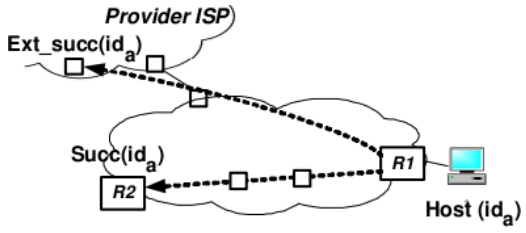


Figure 2.7: A host with id_a has pointers to an internal successor, Succ(id_a), and an external successor, Ext_succ(id_a).

ROFL assumes self-certifying identifiers assigned to all nodes and all routers, which are tied to a public-private key pair. Each node in the network is associated to a gateway, called hosting router, which is responsible for maintaining a set of resident node IDs, and establishing the successor and predecessor pointers on behalf of the resident nodes. Afterwards, ROFL assumes three classes of nodes in the system: routers, stable hosts (e.g., server and stable desktop machines) and ephemeral hosts (which are intermittently connected at a particular location, either because of mobility or frequent shut-downs, e.g., laptops and home PCs), where the decision about whether a node is stable or ephemeral is made by the authority who administers the router at which it is resident. Specially in the case of ephemeral hosts, they cannot serve as successor or predecessor to other IDs; they merely establish a path between themselves and their predecessor, which keeps a source-route to the ephemeral hosts. When other nodes route to this ephemeral ID, the packet will travel to the predecessor's hosting router, and then be forwarded to the ephemeral host.

In order to detect failures in the physical network connecting hosting routers, ROFL assumes the existence of an underlaying OSPF-like protocol that provides a network map, and is responsible for identifying physical network failures. Such OSPF-like protocol finds paths between hosting routers present inside the same AS (intra-domain level), and maintains routes to external border routers whom the internal hosting routers have pointers to (inter-domain level).

In ROFL, each AS X runs its own ROFL-ring (RR), RR_X, creating the intra-domain virtual ring. In order to assure that hosts within its RR are reachable from other domains, RR_X needs to be merged with the RRs of other domains. This is done in two phases. First, AS X discovers its up-hierarchy graph G_X, which consists of all ASes "above" X in the AS hierarchy of Internet. Next, X performs a Canon-style recursive merging protocol that constructs additional successors to RRs in other ASes, as depicted in Figure 2.8. This is done by merging X's RR with RRs in the domains at or below X in the AS graph.

The merging process requires an isolation property to assure that when a host in domain X sends a packet to a host in domain Y, the data path will stay within the subtree rooted at the earliest common ancestor of these two domains. Furthermore, if a host within domain X sends a packet to another host

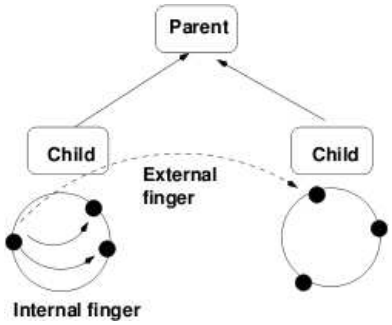


Figure 2.8: Merging rings.

in the same domain, no external pointers can be used. For example, Figure 2.9 shows the internal and external routing state for a router hosting an identifier 8 residing in AS 4. The hosting router has an internal successor pointer to the router hosting identifier 20 and external successor pointers to hosting routers residing in ASes 5 and 3. The join protocol of ROFL discovers the external successor at each level of the joining node's up-hierarchy. For instance, the hosting router for 8 maintains an external successor to 16 at the level of AS 2, and an external successor to 14 at the level of AS 1. In order to exemplify the required isolation property, if the identifier in AS 5 were 12 instead of 16, 8 would not maintain 14 as a successor, since it violates the isolation property.

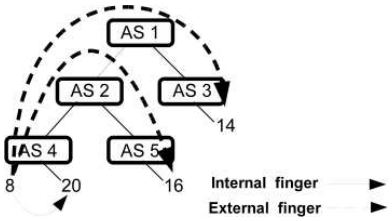


Figure 2.9: Routing state for virtual node with identifier 8.

2.2 Unmanaged Internet Protocol

This section details the Unmanaged Internet Protocol (UIP) proposal, which adopts an XOR mechanism to provide communication through network discontinuities, like firewall and NAT, by building an overlay network. All the details presented in this section, including the protocol specification and the exemplification figures, were extracted from [17, 18].

UIP is an identity-based inter-networking protocol designed to fill the connectivity gaps left by address-based protocols such as IP. UIP stitches together multiple address-based layer 2 and layer 3 networks into one large "layer 3.5" internetwork, in which nodes use topology-free (name-independent) identifiers in a flat name space instead of hierarchical addresses. All UIP nodes

act as self-configuring routers, enabling directly or indirectly connected UIP nodes to communicate via paths that may cross any number of address domains. As exemplified in Figure 2.10, UIP provides an overlay routing layer that operates on top of the Internet's existing routing layer to provide robust peer-to-peer connectivity between personal devices even when those devices are mobile and/or behind firewalls or NATs.

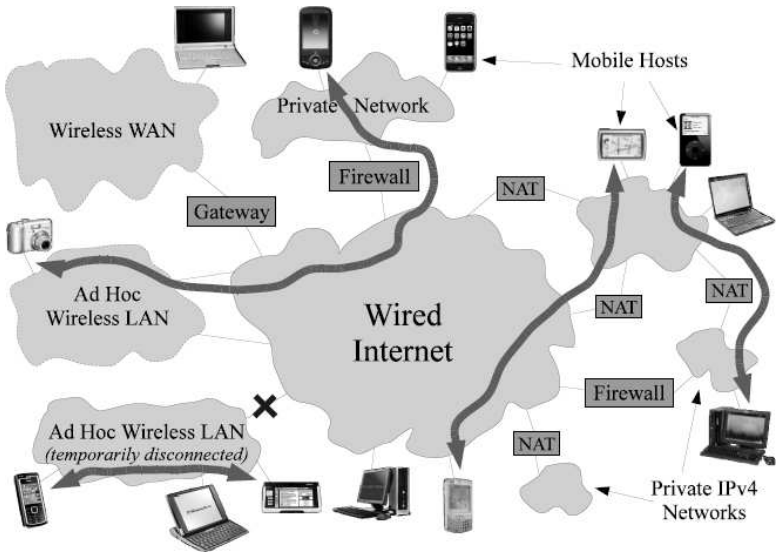


Figure 2.10: Global connectivity challenges for the UIA overlay routing layer.

Each node in UIP network maintains a neighbor table, in which the node records information about all the other UIP nodes with which it is actively communicating at a given point in time, or with which it has recently communicated. The nodes listed in the neighbor table of a node A are termed A's neighbor, and are not necessarily "near" to A in either geographic, topological, or node identity space; the presence of a neighbor relationship merely reflects ongoing or recent pairwise communication.

As part of each entry in a node's neighbor table, the UIP maintains whatever information it needs to send packets to that particular neighbor. This information describes a link between the node and its neighbor. A link between two nodes A and B may be either physical or virtual. A physical link is a link for which connectivity is provided directly by some underlaying protocol. For example, if A and B are both well-connected nodes on the Internet and can successfully communicate via their public IP addresses, then AB is a physical link from the perspective of the UIP layer, even though this communication path may in reality involve many hops at the IP layer and even more hops at the link layer. A virtual link, in contrast, is a link between two nodes that can only communicate by forwarding packets through one or more intermediaries at the UIP level. In Figure 2.11, for example, virtual link AC builds on physical links AB and BC, and virtual link AD in turn builds on virtual

link AC and physical link CD. Once these virtual links are established, node A has nodes B, C and D in its neighbor table. Node D only has nodes C and A as its neighbors; D does not necessarily need to know about B in order to use virtual link AC.

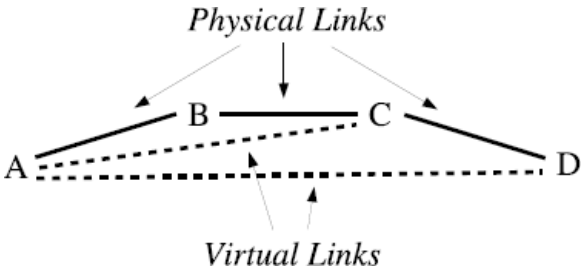


Figure 2.11: Forwarding via virtual links.

UIP treats node identifiers as opaque l-bit binary bit strings, where the longest common prefix (lcp) of two nodes n_1 and n_2, writen $lcp(n_1, n_2)$, is the longest bit string prefix common to their respective UIP identifiers. The proximity of two nodes $prox(n_1, n_2)$ is the length of $lcp(n_1, n_2)$, i.e., the number of contiguous bits their identifiers have in common starting from the left. As illustrated in Figure 2.12, each node n divides its neighbor table into l buckets, and places each of its neighbors n_i into bucket $b_1 = prox(n, n_i)$ corresponding to that neighbor's proximity to n, which is the XOR distance between both n and n_i identifiers.

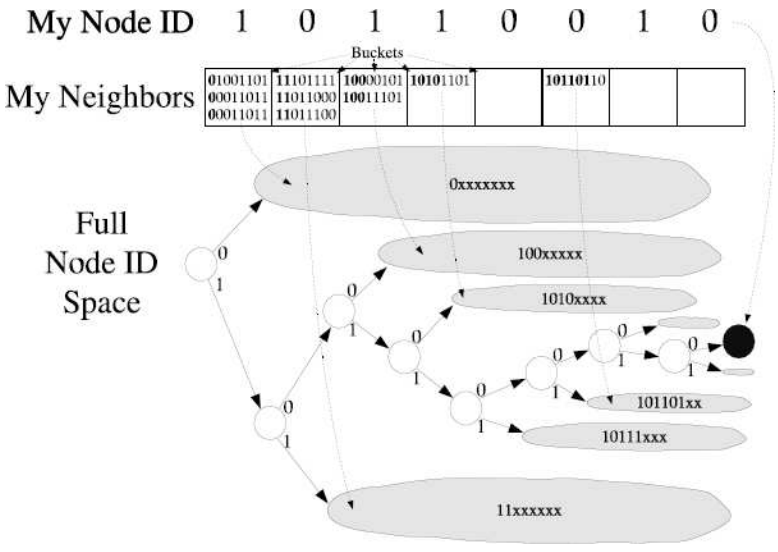
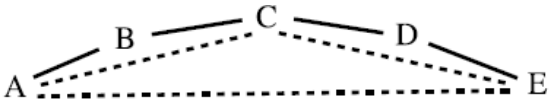


Figure 2.12: Neighbor tables, buckets, and node ID space.

In order for an UIP network to be fully functional, the network must satisfy a connectivity invariant, where each node x perpetually maintains an active connection with at least one neighbor in every bucket b, as long as a reachable node exists anywhere in the network that could fit into bucket b.

Based on such connectivity invariant, UIP nodes are able to forward traffic across the entire network, and it explores two methods for traffic forwarding: one based on source routing, and the other one based on recursive tunneling.

With source routing, each entry in a node's neighbor table contains a complete source route to the target node. The source route lists the UIP identifiers of a sequence of nodes, starting with the origin node and ending with the target node, such that each adjacent pair in the sequence has a working physical link between them. Considering the example depicted in Figure 2.13, in which five nodes A, B, C, D, E are connected by a chain of physical links. Nodes A and C have established a virtual link AC by building two-hop source route via their mutual neighbor B, and nodes C and E have similarly established a virtual link CE via D. Suppose node A subsequently learns about E from C and desires to create a virtual link AE via C. Node A contacts C requesting C's source route to E, and then appends C's source route for $CE(C, D, E)$ to A's existing source route for $AC(A, B, C)$, yielding the complete physical route A, B, C, D, E. To send a packet to E, node A includes in the packet's UIP header the complete source route for the virtual link AE stored in its neighbor table entry for E.



Source_Routing

A's Neighbor Table:

B	Phys	123.45.67.89
C	Virt	A→B→C
E	Virt	A→B→C→D→E

"→" Refers to *physical links only*

Tunneling

A's Neighbor Table:

B	Phys	123.45.67.89
C	Virt	A⇨B⇨C
E	Virt	A⇨C⇨E

"⇨" Refers to *physical or virtual links*

Figure 2.13: Source routing versus recursive tunneling.

In contrast with source routing, where each entry in a node's neighbor table for a virtual neighbor contains a complete, explicit route that depends only on physical links, recursive tunneling preserves the abstraction properties of neighbor relationships by allowing the forwarding path describing a virtual link to refer to both physical and virtual links. As a results, each neighbor table entry representing a virtual link only needs to hold two UIP identifiers: the identifier of the target node, and the identifier of the "waypoint" through which the virtual link was constructed. In the example in Figure 2.13, node A has constructed virtual link AC via B, and C has constructed virtual link CE via D, and as before, A learns about E from C and wants to construct a virtual link AE via

C. With recursive tunneling, A does not need to duplicate its route to C or ask C for information about its route to E. Instead, A merely depends on the knowledge that it already has to get to C, and that C has to get to E, constructing a neighbor table entry for E describing the "high-level" two-hop forwarding path A, C, E.

As illustrated in Figure 2.14, to send a packet to E using the recursive tunneling approach, node A wraps the packet data in three successive headers. First, it prepends an UIP tunneling header describing the "second-level" virtual path from A to E via C. Only nodes C and E will examine this header. Second, A prepends a second UIP tunneling header describing the "first-level" virtual path from A to C via B. Finally, A prepends the appropriate lower-layer protocol's header, such as an IP or Ethernet header, necessary to transmit the packet via the physical link from A to B.

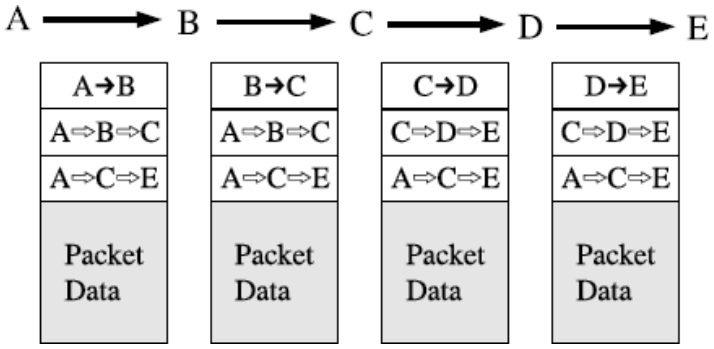


Figure 2.14: Forwarding by recursive tunneling.

When the packet reaches node B, it strips off the lower-layer protocol header, and looks in the first-level UIP tunneling header to find the UIP identifier of the next hop. B then looks up this identifier in its neighbor table, prepends the appropriate (new) lower-layer protocol header, and transmits the packet to C. When the packet reaches node C, it strips off both the lower-layer protocol header and the first-level UIP tunneling header, and examines the second-level tunneling header to find the final destination E. In the sequence, C prepends a new first-level tunneling header describing the route from C to E via D. Finally, C prepends the lower-layer protocol header for the physical link from C to D and forwards the packet, which will be delivered to E in a similar process performed in the path AC.

The UIP also supports some path optimizations. Basically, in the source routing approach, it is possible to combine two shorter paths into a longer one by checking for nodes that appear in both shorter paths. For example, if Figure 2.15(a), suppose that node A has established a virtual link AD via B with path A, B, C, D, by building on virtual link BD with path B, C, D. A virtual link also exists between D and F. A now learns about F through D and attempts to create a virtual link AF via D. Without path optimization, the resulting path will be A, B, C, D, C, B, F. The path can be trivially shortened to the optimal A, B, F. The same optimization shortens the path from A to F in

Figure 2.15(b) from A, B, C, D, C, E, F to the optimal A, B, C, E, F. This path optimization does not help in the case of Figure 2.15(c).

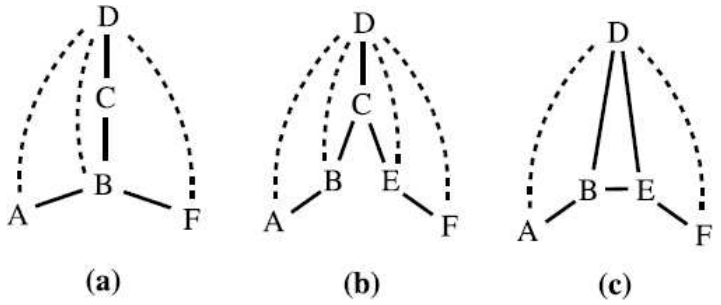


Figure 2.15: Path optimization opportunities on different topologies, when A builds a virtual link to F via D.

Conversely, path optimization is not as easy in forwarding by recursive tunneling, because the information needed to perform the optimization is more spread out through the network. For example, in Figure 2.15(a), node A knows that the first hop along virtual link AD is the physical link AB, but A does not necessarily know what type of link BD is and may not even know that node C exists.

The XOR metric used in UIP to create neighbor tables structured in l buckets offers the ideal scenario for using the proposed *local visibility* concept, since it establishes l neighborhood relationships, as opposed to the simple circular relations used in the virtual ring of IBR considering successor and predecessor nodes. Finally, the connectivity invariant of UIP is a strong condition for certain network scenarios, since the required neighbors for certain buckets can be "distant" at the network topology, causing elevated signaling overhead in order to discover such nodes. The proposed *local visibility* concept addresses such question, since it prioritizes the insertion of physically near nodes in the routing tables.

2.3 Summary

This chapter briefly described IBR and UIP, which constitute the main related work. Once again, it is important to emphasize that this chapter was entirely elaborated using the published content of IBR [29, 30, 31] and UIP [17, 18], including the protocols' specification and the figures presented.

From the IBR proposal, the main characteristic that heavily contributed to the development of this work is the IBR approach of performing flat routing directly on top of flat identifiers. Such scenario raises routing to a new condition, where the separation between identifiers and locators are performed by simple removing the locators from the network. Basically, IBR is the first scalable routing mechanism to operate using purely identifiers [30]. However, the virtual ring structure of IBR strictly requires global ring correctness, not supporting the proposed *local visibility* approach.

Conversely, the XOR metric used to provide communication between personal devices in the overlay network structure of UIP offers the fundamental basis for supporting the proposed *local visibility* concept. The use of l buckets to organize the neighbor tables based on the XOR metric improves the robustness for establishing neighborhood relationships in the network, as opposed to the simple successor and predecessor approach of IBR.

In [61], it is evaluated the impact of geometry in resilience and proximity for overlay DHTs, indicating that the ring structure and the XOR structure are very similar in several aspects, even tough the ring offers a slight better level of flexibility for selecting neighbor nodes and routes to forward packets. However, we argue in this work that operating the ring geometry directly on top of the physical network structure, instead of using an underlay network as analyzed in [61], compromises the flexibility offered by the ring, since maintaining the global ring correctness under such circumstances becomes challenging, once the connectivity problems can not be masked anymore by the IP network. In this way, the XOR geometry offers better flexibility for operating directly on top of the physical network structure, specially using the proposed *local visibility* scenario, due to the diversity of neighborhood relations spread in the buckets.

The next chapter brings the main contribution of this work, detailing the proposed XOR-based routing mechanism which performs flat routing directly on top of the physical network structure using the proposed *local visibility* concept.

Chapter 3

XOR-based Flat Routing with Local Visibility

Routing has been defined as the process in which packets are sent from source to destination using information related to the location of the nodes in the network. This concept is not only present in the name-dependent routing mechanisms, but also in the name-independent routing mechanisms, where although flat identifiers are used to uniquely refer to nodes in the network, mapping mechanisms are still required to translate flat node identifiers into addresses indicating the location of nodes (locators) in the network in order to forward traffic to them.

As mentioned before, an important contribution in the name-independent routing research was introduced by the Identity-Based Routing (IBR) [29, 30, 31], which is pioneer in the concept of routing directly on top of flat identifiers. It eliminates the need for an underlay network providing communication between nodes and, consequently, eliminates the need for mapping systems to translate from flat identifiers to locators. IBR uses unique name-independent node identifiers, and organizes nodes into a virtual ring in order of increasing identifiers, where each node maintains a virtual neighbor set (vset) of cardinality r containing the node identifiers of the $r/2$ closest neighbors clockwise (successors), and the $r/2$ closest neighbors counter clockwise (predecessors). Such r neighbors are crucial for traffic forwarding and for maintaining the integrity of the virtual ring structure, since failures in the vset may lead to network partitions, compromising the overall traffic forwarding.

In this way, the proposal in this work leverages the concept of routing directly on top of flat identifiers introduced by IBR, but opts for an alternative mesh network structure, where neighbors are selected in the network using XOR operations. The usage of the mesh structure increases the connectivity level between neighbor nodes and improves the robustness of the routing mechanism. The XOR approach was introduced in Kademlia [16] and UIP [18], being extended in this proposal

to operate without the usage of an underlay network. Essentially, the proposed approach brings the notion of locality to the (location-free by nature) flat identity space, and is mainly based on the mechanism for building the routing tables here proposed. Such notion of nodes' locality creates the concept of *local visibility* proposed in this work. The *local visibility* is essentially related to the ability of the proposed flat routing mechanism to deliver traffic using just a fraction of the overall routing information available, prioritizing the physically near information.

The remainder of this chapter is organized as follows. Section 3.1 details the XOR-based routing principle, explaining the XOR metric used to organize the routing tables and to create the mesh network structure. Section 3.2 presents the proposed mechanism for building the routing tables. Section 3.3 introduces the XOR-based routing process to forward packets using the XOR metric. Section 3.4 details the *local visibility* concept, explaining the occurrence of empty buckets (gaps in the routing tables) and how gaps interfere in the overall network navigability, i.e., how they interfere in the traffic forwarding. Section 3.5 explains how to handle dynamic scenarios such as insertion of nodes in the network, mobility and failures. Section 3.6 summarizes this chapter.

3.1 XOR-based Routing Principle

The mechanism here proposed uses n-bit flat identifiers to organize the routing tables in n columns and route packets through the mesh network structure. Its routing principle uses the bitwise exclusive or (XOR) operation between two flat node identifiers a and b as their distance, which is represented by $d(a,b) = a \oplus b$, being $d(a,a) = 0$ and $d(a,b) > 0$, $\forall a,b$. Given a packet originated by node x and destined to node z, and denoting $\mathbb{Y}$ as the set of identifiers contained on x's routing table, the XOR-based routing mechanism applied at node x selects the node $y \in \mathbb{Y}$ that minimizes the distance towards z, which is expressed by the following routing policy

$$\mathcal{R} = \underset{y \in \mathbb{Y}}{\operatorname{argmin}}\{d(y,z)\}. \tag{3.1}$$

Each node maintains a routing table in which its knowledge about neighbor nodes is spread into n columns called buckets and represented by $\beta_i, 0 \leq i \leq n-1$. Such organization of the buckets of the routing tables drives the creation of the mesh network structure and improves the granularity level in which neighbors are selected in the network, as opposed to the simple successor and predecessor approach of IBR. Table 3.1 presents an example of a routing table for node 0001 in an identity space where $n = 4$. Each time a node a knows a novel neighbor b, it stores the information regarding node b in the bucket β_{n-1-i} given the highest i that satisfies the following condition[1]

[1] div denotes the integer division operation on integers.

$$d(a,b) \operatorname{div} 2^i = 1, a \neq b, 0 \leq i \leq n-1. \tag{3.2}$$

For example, consider *a* = 0001 and *b* = 0010. The distance *d*(*a*,*b*) = 0011 and the highest *i* that satisfies the condition (3.2) is *i* = 1, concluding that the identifier *b* = 0010 must be stored in the bucket $\beta_{n-1-i} = \beta_2$. Basically, condition (3.2) denotes that node *a* stores node *b* in the bucket β_{n-1-i}, where *n*-1-*i* is the length of the longest common prefix (*lcp*) between both identifiers of nodes *a* and *b*. This can be observed in Table 3.1, where the buckets $\beta_0, \beta_1, \beta_2, \beta_3$ store the identifiers having *lcp* 0, 1, 2, 3 with node 0001.

Table 3.1: Hypothetic routing table for node 0001 with $n = 4$.

β_0	β_1	β_2	β_3
1000	**0**100	**00**10	**000**0
1001	**0**101	**00**11	
1010	**0**110		

The proposed routing mechanism also considers a *K* factor, which defines the amount of information required per bucket that will be sought in the network during the dynamic distributed process of building the routing tables (detailed in Section 3.2). Since a bucket β_i has at most 2^{n-1-i} entries, if $K > 2^{n-1-i}$ we limit K for that bucket to $K = 2^{n-1-i}, 0 \leq i \leq n-1$. Essentially, by varying the value of the *K* factor, nodes have wider or narrower knowledge about the network. Furthermore, the adjust of the *K* factor provides a controllable mechanism which helps to increase the robustness of the mesh network structure at the granularity of each individual bucket. In IBR, r neighbors constitute the vset, being $r/2$ successors and $r/2$ predecessors. In this proposal, the number of neighbors desirable in each routing table is

$$N = \sum_{i=0}^{n-1} \min(K, 2^{n-1-i}), \tag{3.3}$$

establishing *lcp*-based relations between neighbor nodes which are distributed in n independent buckets.

3.2 Building the Routing Tables

The XOR-based routing mechanisms available in the literature [16, 18] assure the communication between all pairs of nodes present in the network if at least one existing neighbor is present in

each individual bucket of the XOR-based routing tables (the connectivity invariant of UIP). In such proposals, the use of bigger values for the K factor contributes, for example, to stretch reduction. However, fulfilling all buckets with K information may lead nodes to search for neighbors in the entire network, since the required neighbor nodes can be physically distant (in number of hops) from each other. Therefore, fulfilling the routing tables in such proposals may create a scenario of elevated signaling overhead not only to converge, but also to maintain the routing mechanism operational.

As mentioned before, this work proposes the concept of *local visibility* as an alternative to address the scalability problems present in the proposals available in the literature due to the connectivity invariant. The *local visibility* concept is mainly present on the process of building the routing tables, which searches for neighbors according to the region of interest located in the intersection area of Figure 3.1. The proposals available in the literature [16, 18], on the other hand, only consider the relations that nodes have at the flat identity space (left circle of Figure 3.1) to build the routing tables. Section 3.4 details the concept of *local visibility*, explaining the occurrence of empty buckets (gaps) in the routing tables in order to provide scalability to the proposed routing mechanism. Furthermore, the process of building the routing tables operates using the pull model, where nodes search for the routing information using unicast messages addressed to the neighbors already present in the routing tables, avoiding the dissemination of information in the entire network inherent of the push model.

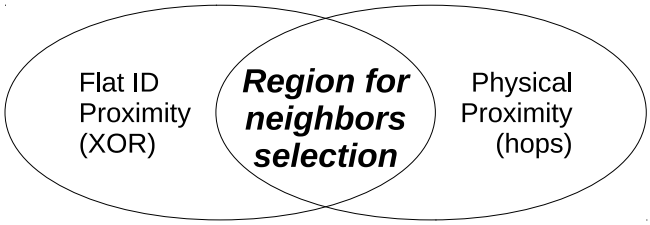


Figure 3.1: Region in which neighbors are selected during the process of building the routing tables.

Essentially, the relation at the flat identity level is defined by the XOR distance described in Section 3.1, and the relation at the physical level is obtained through the number of physical hops separating the nodes. During the process to build the routing tables, a node a can know a node b from two distinct ways: 1) a *discovery process* (Section 3.2.1), in which nodes actively search for neighbors to fill the buckets; and 2) a *learning process* (Section 3.2.2), where nodes use information contained in the signaling messages that cross them to add more information into their buckets in a costless passive fashion.

3.2.1 Discovery Process

In the *discovery process*, there are two types of neighbor nodes: 1) *physical neighbors*, which are the nodes directly/physically connected, also defined as 1-hop distance neighbors; and 2) *virtual neighbors*, which are the nodes not physically connected, where the physical distance is bigger than 1-hop. Essentially, in the proposed process of building the routing tables, a node always knows its *physical neighbors*, and all of them are stored in the bucket having the greatest *i* that solves the condition (3.2). Such insertion of *physical neighbors* in the buckets is not limited by the previously defined *K* factor, since their insertion in the routing tables is essential to provide the communication between nodes using the purely flat IDs approach.

The *discovery process* is constituted of three signaling messages: 1) `HELLO`, 2) `QUERY` and 3) `RESPONSE`. The `HELLO` message has the scope of local links, and is responsible for the discovery of the *physical neighbors*. The `HELLO` message is firstly generated when a given node detects a new active link with another node, and it only contains information regarding the flat ID of its originating node. The `QUERY` and `RESPONSE` messages, on the other hand, are used to discover *virtual neighbors*, and are always sent using the unicast model to the nodes already stored in the routing table. So, after inserting all *physical neighbors* in the routing table due to the exchange of `HELLO` messages, a node that still has buckets requiring information to reach the defined K value, actively searches for *virtual neighbors* that can fill such buckets according to the process depicted in Figure 3.2. Basically, the protocol defines that on each iteration of the *discovery process*, `QUERY` messages are sent to all neighbors stored in the routing table.

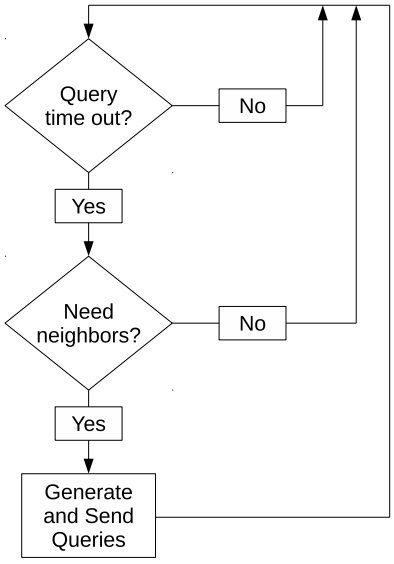


Figure 3.2: Diagram of the process used to generate and send `QUERY` messages.

The `QUERY` message is composed of four fields: 1) `DISTANCE`, 2) `SRC_ID`, 3) `QUERY_VECTOR`

and 4) KNOWN_NODES.

- DISTANCE: it is an incremental field used to indicate the number of physical hops from the originating node until the queried node. It is set to 0 by the source node, and incremented in one unit by each node on the path;

- SRC_ID: it corresponds to the flat ID of the node originating the QUERY message, and is essential to the node receiving the QUERY message to find the required neighbors;

- QUERY_VECTOR: it comprises a n-vector describing the number of neighbors required in each one of the n buckets. A position in the vector with value 0 indicates a bucket which does not require any information;

- KNOWN_NODES: it contains information about the neighbors present in the routing table of the node originating the QUERY message. This field is implemented using the probabilistic Bloom filter [62] structure, aimed at improving the quality of the queries to obtain better responses with reduced signaling. The Bloom filter is an array of *m* bits, initially all set to 0. The size *m* of the array and the number *t* of independent hash functions are defined according to the number of expected elements to be inserted in the Bloom filter and the acceptable *false positive* rate. For each element *x*, the bits $h_i(x)$ are set to 1 for $1 \leq i \leq t$. A location in the array can be set to 1 by multiple elements, resulting in *false positive* occurrences. To check if an item *y* is present in the Bloom filter, all the $h_i(y)$ positions are verified to see if they are set to 1. If not, *y* is not, for sure, a member of the Bloom filter. On the other side, if all $h_i(y)$ are set to 1, *y* is present in the Bloom filter with some probability of being a *false positive*.

The node which receives the QUERY message is able to define the ranges where neighbor nodes are required, and the amount of neighbors required in each range based on the information contained in the SRC_ID field and in the QUERY_VECTOR field, respectively. For example, consider an identity space where $n = 4$ and $K = 1$, and a QUERY message generated by node 0000 in order to fill its buckets β_1 and β_2 that are empty. The SRC_ID field carries the flat ID 0000 of the node, and the QUERY_VECTOR field is set to 0,1,1,0, since buckets β_0 and β_3 are already fulfilled and buckets β_1 and β_2 require one neighbor each to achieve the defined K factor. Based on the SRC_ID 0000, the node which receives this QUERY message defines the ranges: 1) from 1000 to 1111 for bucket β_0 (none bit in common); 2) from 0100 to 0111 for bucket β_1 (one bit in common); 3) from 0010 to 0011 for bucket β_2 (two bits in common); and 4) 0001 for bucket β_3 (three bits in common). Afterwards, it infers that one neighbor is required in the range of β_1 (from 0100 to 0111), and one neighbor is required in the range of β_2 (from 0010 to 0011) according to the QUERY_VECTOR field.

Based on such information, the queried node searches in its routing table for neighbors which fit in the specified ranges, giving priority to include in the RESPONSE message the neighbors which are physically closer, i.e., the nodes whose distance in number of hops is smaller. Such distance information is associated with the entries present in the routing table. Consequently, the process of building the routing tables not only considers the similarity that nodes have at the flat identity space, but also the existent physical distance (in number of hops) between nodes. Such project decision is the first characteristic of the *local visibility* concept proposed.

Before inserting the selected neighbors in the RESPONSE message, the queried node checks in the KNOWN_NODES field if the selected neighbors can be already present in the routing table of the requesting node. If the Bloom filter points that the neighbor is present in the routing table of the requesting node, another neighbor is selected if available. In the case where no other neighbor is available, no answer is provided to the current range. In this way, after processing the QUERY message, the queried node creates a RESPONSE message containing the selected neighbors and sends it to the requesting node. The RESPONSE message has two fields: 1) DISTANCE and 2) ANSWER.

- DISTANCE: it has the same functionality in both QUERY and RESPONSE messages, i.e., it starts set to 0 and is incremented by nodes present in the path, indicating the number of hops separating the source node from the destination node;
- ANSWER: it is composed of a set of tuples in the format <NEIGHBOR_ID, NEIGHBOR_DISTANCE>, where each tuple represents a neighbor to be inserted in the routing table of the requesting node.

When the RESPONSE message arrives at the requesting node, it knows its physical distance towards the queried node based on the DISTANCE field, and it starts to insert the neighbors contained in the ANSWER field. Basically, for each answer, it stores the NEIGHBOR_ID in the bucket having the greatest i that solves the condition (3.2), and associates to it the following information:

1. the total distance towards the *virtual neighbor* given by DISTANCE + NEIGHBOR_DISTANCE;
2. the *physical neighbor* (next hop) from where the RESPONSE message was received;
3. the local interface used to forward traffic to the next hop.

Finally, as shown in Figure 3.2, the proposed *discovery process* is interactive, and each node uses the information already contained in its routing table to search for missing *virtual neighbors*. So, in the first iteration of the *discovery process*, QUERY messages are sent to the *physical neighbors*, which

are the only information present in the routing table. As *virtual neighbors* are discovered, they are also queried in the next iterations (in each query time out) of the process if the node still requires neighbors to fulfill its routing table. Basically, on each iteration of the *discovery process*, the amount of neighbors requested in the `QUERY` messages is

$$\Delta N = \sum_{i=0}^{n-1} \left[\min(K, 2^{n-1-i}) - \min(K, v(\beta_i))\right], \tag{3.4}$$

where $v(\beta_i)$ represents the amount of neighbors v already contained in the bucket β_i.

3.2.2 Learning Process

In order to perform flat routing directly on top of flat identifiers, i.e., routing without using an underlay network of locators to forward traffic as available in the literature [16, 18], it is required to assure the existence of a path connecting two *virtual neighbor* nodes a and b. In this way, the proposed mechanism for building the routing tables has a *learning process* being passively executed by nodes in parallel to the *discovery process*. The first function of the *learning process* is to assure the creation of symmetric routing tables between neighbor nodes, as defined in Property (1).

Property 1 *If a given node a is present in the routing table of node b, then node b is present in the routing table of node a.*

Afterwards, the *learning process* assures the existence of a path connecting two *virtual neighbor* nodes a and b, as defined in Property (2).

Property 2 *All nodes present in the physical path between the virtual neighbor nodes a and b have information about both nodes on their routing tables.*

The *learning process* is mainly based on the `SRC_ID` field of the `QUERY` message and in the `ANSWER` field of the `RESPONSE` message. Essentially, the symmetry between the routing tables defined in Property (1), and the existence of a path between two *virtual neighbor* nodes defined in Property (2), are achieved through the exchange of complementary `QUERY` and `RESPONSE` messages. Figure 3.3 details the *learning process* from the perspective of node 0000, where nodes 0110, 0010 and 0001 suffice to reach the *N* required neighbors defined in (3.3), since the scenario depicted in the figure considers $K = 1$.

The example of Figure 3.3 starts with node 0000 sending a `QUERY` message to its *physical neighbor* 1000 already stored in its routing table (obtained through the `HELLO` message), describing its buckets which still require information. In the sequence, when node 1000 receives the `QUERY`

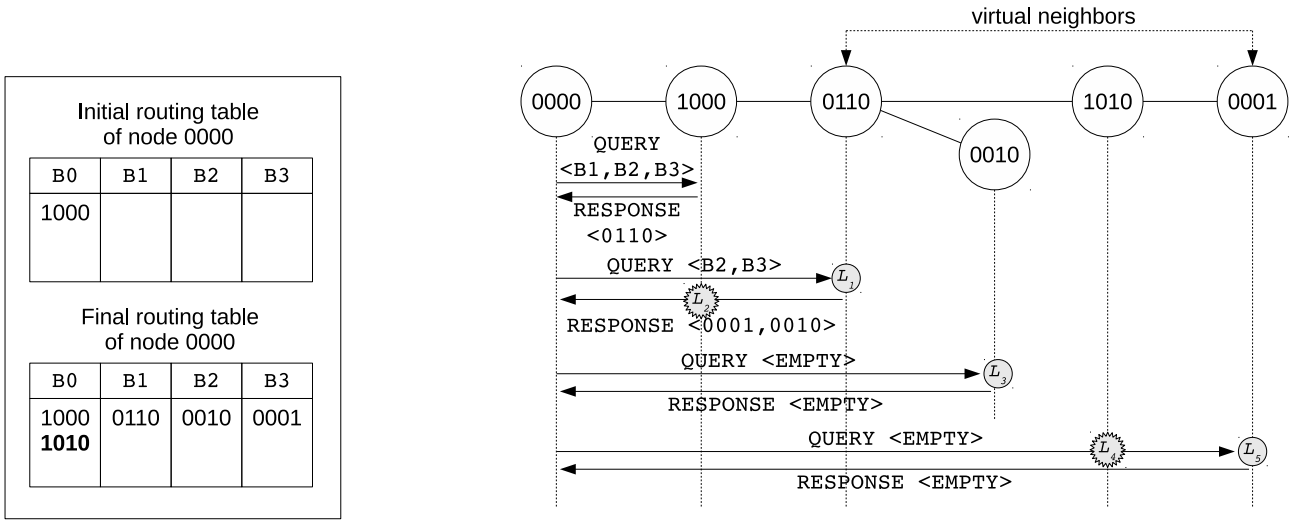


Figure 3.3: Exemplification scenario of the *learning process* from node 0000 perspective.

message, it processes the message according to the procedures described in Section 3.2.1, and sends back a RESPONSE message informing node 0000 about the existence of node 0110. Consequently, after node 0000 receives the RESPONSE message, it stores the discovered node 0110 in its routing table, concluding the first iteration of the *discovery process*.

In the second iteration of the process, as node 0000 still needs information in some of its buckets, it generates another QUERY and sends it to nodes 1000 and 0110, which are the nodes present in its routing table. For simplicity, Figure 3.3 focuses on the iteration with node 0110 recently discovered, assuming that node 1000 has no new interesting information to node 0000, case in which an empty RESPONSE message is returned from node 1000. In this way, at the instant that node 0110 receives the QUERY message from node 0000, the first learning L_1 occurs. Essentially, node 0110 uses the SRC_ID field of the QUERY message to learn about node 0000, consolidating the routing tables' symmetry between the *virtual neighbors* 0000 and 0110 defined in Property (1).

In the sequence, node 0110 sends back a RESPONSE message informing node 0000 about the existence of nodes 0001 and 0010, which on the way back to 0000 crosses node 1000, causing the second learning L_2. At this time, the *learning process* uses the ANSWER field of the RESPONSE message to insert nodes 0001 and 0010 in the routing table of node 1000, assuring the existence of a path between the *virtual neighbors*, where all nodes present in the path have information about the *virtual neighbors*, as defined in Property (2). Note that the information about node 1010 is not returned to node 0000, since it does not fit in the required flat ID ranges.

When node 0000 receives the RESPONSE message it fulfills all of its buckets, and the remaining iterations present in Figure 3.3 are triggered by the *learning process* to assure the requirements of Properties (1) and (2). First of all, node 0000 sends two empty QUERY messages to nodes 0001 and

0010 recently discovered. The empty QUERY messages do not require information for buckets, but they are aimed at informing nodes 0001 and 0010 about their insertion in nodes' 0000 routing table. As a consequence of these two empty QUERY messages, three other learnings L_3, L_4 and L_5 occur, all of them based on the SRC_ID field of the QUERY messages. In the case of L_3, node 0010 learns about node 0000, and in the case of L_5, node 0001 learns about node 0000.

Specially in the cases of L_2 and L_4, they trigger some complementary iterations of the *learning process* to assure the occurrence of both Properties (1) and (2), as detailed in Figure 3.4. In the case of L_2, node 1000 exchange two QUERY messages with nodes 0010 and 0001, both of them learned using the RESPONSE message which crossed it from node 0110 towards node 0000. For simplicity, the complementary QUERY messages are empty, but they could also include the requirements of node 1000, if some of its buckets need information. As a consequence of these iterations, three other learnings $L_{2.1}$, $L_{2.2}$ and $L_{2.3}$ occur, and in the specific case of $L_{2.2}$, it triggers the process which results in the $L_{2.2.1}$ occurrence, also detailed in the figure. Basically, node 1010 learns about node 1000 using the SRC_ID field of the QUERY message, and exchange the complementary messages to assure the Property (1) occurrence.

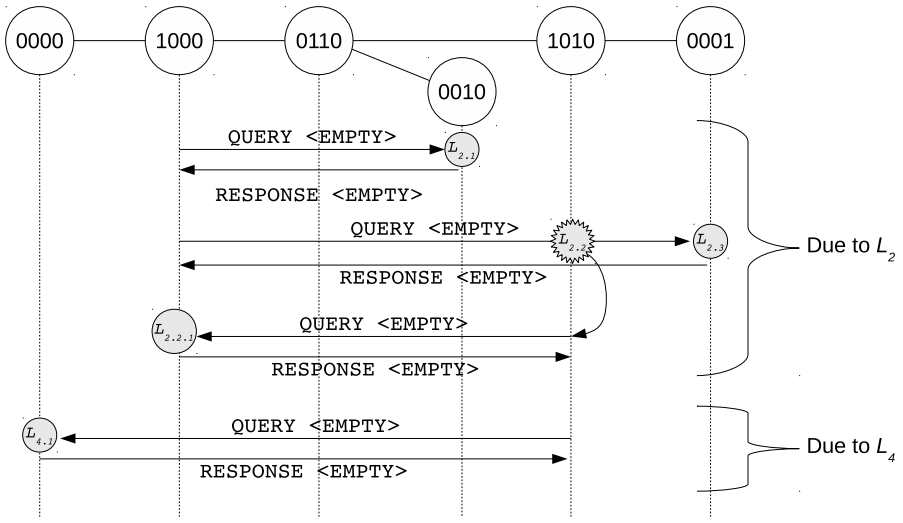


Figure 3.4: Complementary iterations of the *learning process*.

In the case of L_4, the QUERY message destined to node 0001 crosses node 1010, triggering another exchange of QUERY and RESPONSE messages between nodes 1010 and 0000, illustrated in the bottom part of Figure 3.4. This last exchange of messages lead to the occurrence of the learning $L_{4.1}$ at node 0000, resulting in the insertion of node 1010 in the bucket β_0 of node 0000. Obviously, the exemplification scenario of Figures 3.3 and 3.4 is detailed from the perspective of node 0000. In the realistic execution of the proposed mechanism for building the routing tables, nodes are performing both *discovery* and *learning process* in parallel, resulting in a different evolution of the signaling

exchange due to the individual requirements of nodes to fulfill their buckets.

Finally, the decision of stopping the process of building the routing tables is totally dependent of the scenario in which the proposed protocol is instantiated. In a scenario where an elevated level of dynamism is the dominant characteristic, the protocol may need to continuously exchange signaling messages. On the other hand, static scenarios may stop the search as soon as a given node sends signaling messages to all its neighbors, but no new information is discovered. In this case, the interval between each iteration of the process of building the routing tables can be exponentially incremented, for example.

3.3 Routing Process

This section is aimed at detailing how packets are forwarded through the network by using the XOR-based mechanism directly on top of flat identifiers. The XOR-based routing process starts by defining the bucket β_i in which the next hop will be selected to forward the packets. After defining the bucket, the neighbor contained in the bucket β_i which better reduces the XOR distance towards the destination node is selected as next hop. In this way, considering that node a generates/receives a packet destined to node b, it defines the bucket β_i, where the index i is obtained solving condition (3.2). Then, node a routes the packet to the node available in its bucket β_i that is closest to node b in the flat identity space. In other words, the packet is routed to n_R, which is selected by node a computing the following solution

$$n_R = \underset{id \in \beta_i}{\operatorname{argmin}} \{d(id, b)\}, \tag{3.5}$$

where id represents each one of the identifiers contained in the bucket β_i.

In the overlay proposals available in the literature [16, 18], the underlaying network is used as a tunnel providing direct communication between the overlay neighbors. In this way, packets routed by node a are directly delivered to the selected n_R, using an IP network structure for example, ignoring the existence of other nodes in the physical path between nodes a and n_R, as presented in Figure 3.5.

In this proposal, as routing is performed directly on top of flat identifiers, i.e., without the existence of an underlay network to deliver traffic, the n_R can be the identifier of a *virtual neighbor* $j-$hops away ($j > 1$). In this way, node a uses the next hop information associated to n_R entry in its routing table to forward the packets, which is resultant of the *discovery* and *learning processes* for building the routing tables. Figure 3.6 depicts a network used in two examples to detail how packets are forwarded in the proposed routing mechanism.

In the first example, node 0000 wants to communicate with node 0111. In this way, node 0000

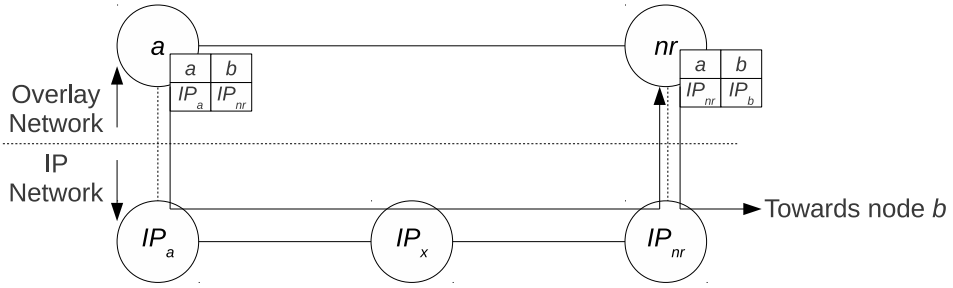


Figure 3.5: Forwarding packets in the overlay proposals available in the literature.

generates a packet destined to node 0111, solves condition (3.2) that returns bucket β_1 and, in the sequence, solves condition (3.5) that points node 0100 as the n_R node towards the destination node 0111. Consequently, node 0000 forwards the packet to its *physical neighbor* node 1000, the physical next hop towards node 0100. It is important to mention that the packet departs from node 0000 towards node 0111 using a single header, where the source identifier is 0000 and the destination identifier is 0111, i.e., there are no auxiliary headers to forward packets.

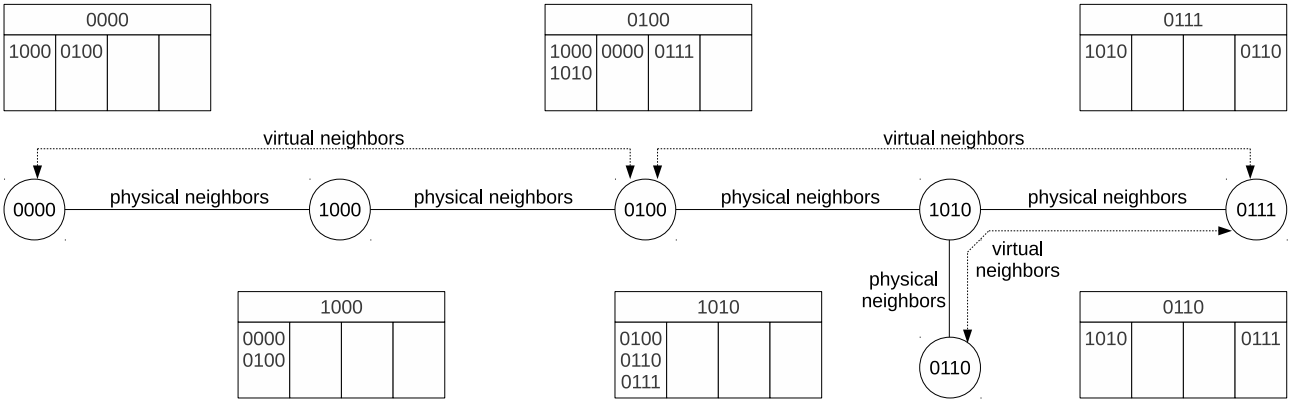


Figure 3.6: Exemplification scenario of the proposed XOR-based routing process.

In the next step, the packet arrives at node 1000, it solves condition (3.2) that returns bucket β_0, and solves condition (3.5) that also points node 0100 as the n_R node towards 0111. In the sequence, node 1000 delivers the packet to node 0100, since both nodes are *physical neighbors*. In the third step, the packet arrives at node 0100, it solves condition (3.2) that returns bucket β_2, and solves condition (3.5) that finds the destination node 0111 in its routing table, passing through its *physical neighbor* node 1010. Consequently, node 1010 repeats the routing process, and finds in its bucket β_0 the destination node 0111, delivering the packet.

In the second example, node 0000 wants to communicate with node 0110. Similarly to the previous example, node 0000 generates a packet addressed to node 0110, solves condition (3.2) that returns bucket β_1, and solves condition (3.5) that points node 0100 as the n_R node towards 0110,

passing through its *physical neighbor* node 1000. After some steps, the packet arrives at node 0100, it solves condition (3.2) that returns bucket β_2, and solves condition (3.5) that points node 0111 as the n_R node towards the destination node 0110. Consequently, node 0100 forwards the packet to its *physical neighbor* node 1010.

However, in this case, when node 1010 solves condition (3.2) it returns bucket β_0, where node 1010 has information about nodes 0100, 0110 and 0111. In this way, at the moment that node 1010 solves condition (3.5), it finds $d(0100, 0110) = 0010$, $d(0110, 0110) = 0000$ and $d(0111, 0110) = 0001$, forwarding the packet to node 0110 due to the smaller XOR distance value which was obtained. Note that the routing decision taken by node 1010 optimizes the packet delivery, deviating the packet that was previously being forwarded towards node 0111, directly delivering it to node 0110. Such behavior is only achievable due to the routing directly on top of flat identifiers paradigm, which allows that nodes on the path from source to destination inspect the single packet header.

A practical benefit of such scenario is route stretch reduction, since all the unnecessary segments of the path are avoided. In the overlay proposals available in the literature, when the packet forwarded in the last example departs from node 0100 towards the n_R node 0111, it crosses node 1010 using the IP underlay network, which is prevented of checking the destination node identifier present in the inner packet header. In this way, the packet is delivered to node 0111, which finds the destination node 0110 in its routing table, and forwards the packet, once again, through node 1010. The comparison between the path used to deliver the packet in the proposed scenario and in the overlay scenario is depicted in Figure 3.7, represented by the thick arrows. The segment of network in the left side of the figure represents the proposed XOR-based flat routing mechanism, and the right side of the figure represents the overlay solutions, where IP_1, IP_2, IP_3 and IP_4 are used as locators of nodes 0100, 1010, 0111 and 0110, respectively.

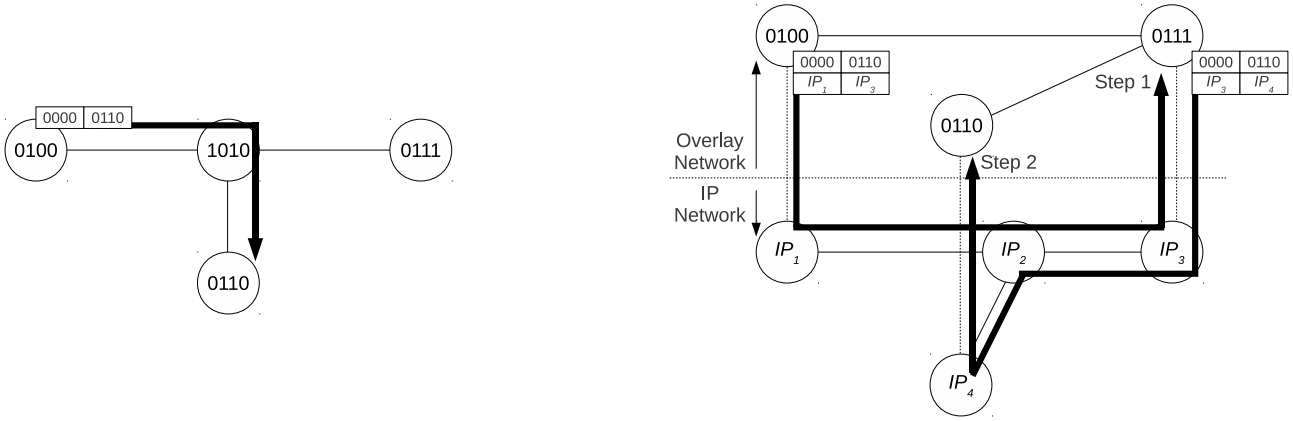


Figure 3.7: Comparison of the obtained paths in the proposed and in the overlay scenarios.

3.4 Concept of Local Visibility

The main objective of the proposed concept of *local visibility* is to provide scalability for the routing system, offering mechanisms to control the exchange of signaling messages required to create the routing tables and avoiding the exchange of signaling messages through the entire network. In this way, the first characteristic of the proposed *local visibility* concept is to prioritize the insertion of physically near nodes in the routing tables. Hence, a given node which receives a `QUERY` message during the proposed process of building the routing tables, prioritizes the insertion of neighbor nodes in the `RESPONSE` message that not only fit in the flat ID ranges specified by the `QUERY_VECTOR` field, but also are closer in number of hops, as detailed in Section 3.2.

In order to control the exchange of signaling messages during the process of building the routing tables, this work introduces the concept of gaps in the XOR-based routing tables. The principle is to allow the occurrence of buckets in the routing tables, where the defined K factor is not satisfied when the required neighbors are physically distant. Consequently, it is also possible the occurrence of totally empty buckets (gaps), when a given node already asked all of its neighbors, but none of them have information to help completing its routing table.

For example, in the virtual ring approach of IBR, it is possible that some nodes have less neighbors on their vsets than the specified r value. However, in IBR it is impossible to allow nodes to have none relations with successors and/or predecessors nodes, since it means breaking the virtual ring, avoiding traffic forwarding in the network. Conversely, the proposed mesh network structure, built using the XOR-based routing tables, allows the occurrence of some empty buckets (gaps), and even in this case traffic can be forwarded in the network. Basically, only the fraction of the traffic which is destined to those nodes related to the empty buckets is affected, as detailed in the sequence.

Considering n-bit flat identifiers, there are 2^n possible nodes in the network, distributed in n buckets of the XOR-based routing tables. From the overall 2^n possible nodes, each bucket $\beta_i, 0 \leq i \leq n-1$ can be filled with 2^{n-1-i} nodes. Consequently, each individual bucket β_i represents $\frac{1}{2^{i+1}}$ nodes, $0 \leq i \leq n-1$, from the entire set of nodes available in the network. For example, in an identity space where $n = 3$, there are $2^3 = 8$ possible nodes available in the network, and the amount of nodes which can be inserted in each individual bucket is $\beta_0 = 4$, $\beta_1 = 2$ and $\beta_2 = 1$, representing 50%, 25% and 12.5% of the nodes available in the network, respectively.

In this way, if bucket β_0 of a given node a is left empty after the process to build the routing tables, it means that node a is avoided to forward traffic to 50% of the nodes available in the network. Nevertheless, considering a fully random distribution of flat IDs in the network, the concentration of nodes in the network which fit in the bucket β_0 is higher than the concentration of nodes which fit in the bucket β_{n-1}. Moreover, the much longer the flat identity space used (the higher the n value), the lower the impact of the occurrence of empty buckets in the right positions (less significant bits) of

the routing tables. In an identity space where $n = 3$, the bucket β_{n-1} represents 12.5% of the overall nodes. But, in an identity space where $n = 32$, all the 26 less significant buckets in conjunction represent only 1.56% of the nodes available in the network. Hence, the proposed *local visibility* concept in conjunction with the XOR-based routing tables propitiates a scenario where traffic tends to be delivered in most of the cases, specially in large-scale networks.

Afterwards, in the virtual ring approach, as traffic moves towards the destination, all nodes that receive the packet to forward may have r possible next hops, as the rationale is to reduce the numerical distance towards the destination node. On the other hand, as traffic is pushed across the network using the proposed flat routing mechanism, the amount of information available in the buckets to forward packets exponentially decreases, since the XOR-based routing mechanism makes progress in the flat identity space, shifting in the buckets from left-to-right direction.

Based on the proposed scenario, the main novelty of such approach is the overall system convergence which become simpler, contributing for the scalability. So, if a node required in the routing table is not physically close, the proposed action is to simple give up about finding such neighbor during the process of building the routing tables, allowing a gap for that missing neighbor. At the same time, it is important to mention the possibility of naturally occurring cases, where the adopted scenario of random distribution of flat IDs propitiates the creation of long-distance links between nodes during the process of building the routing tables. Such long-distance links contribute for the routing mechanism efficiency, similarly to the long-distance relations present in the small world research [45]. Essentially, the *local visibility* concept leads to the creation of routing tables which leverages the network structure, resulting in a scenario where the traffic delivery relies on the integration between the flat identity space and the network structure. Furthermore, other indirect benefits include the reduction of the routing tables and the simplicity to handle insertion/departure of nodes in the network, mobility and failures (detailed in Section 3.5).

As mentioned before, such project decision is inspired in the current research on the concepts of small world and navigability of complex networks [46, 47], which highlights the existence of networks totally navigable, even tough none of their nodes have information about the entire network topology. In this way, one of the main objectives of this work is to evaluate the navigability of some network scenarios, focusing in the development of solutions in which the routing mechanism is fully integrated with the network structure, trying to extract the maximum benefits of the reduced routing tables, and the reduced signaling overhead in order to assure the overall system scalability. Essentially, the objective is to check what happens if the routing mechanism is not in charge of always assuring 100% of the traffic delivery. How efficient (navigable) the networks adopting such solution can be?

Figure 3.8 brings a network composed of eight nodes, using an identity space where $n = 3$,

to exemplify the navigability of a network using the proposed *local visibility* concept. The figure presents the routing tables resultant of the mechanism for building the routing tables using $K = 1$, and highlights the region in which node 000 participates in the network. As can be seen in Figure 3.8, there are four gaps at bucket β_2 of nodes 000, 001, 100 and 101. According to the current condition of the network such gaps are permanent, since none of the neighbors already present in the routing tables of the nodes with gaps have information to fill them.

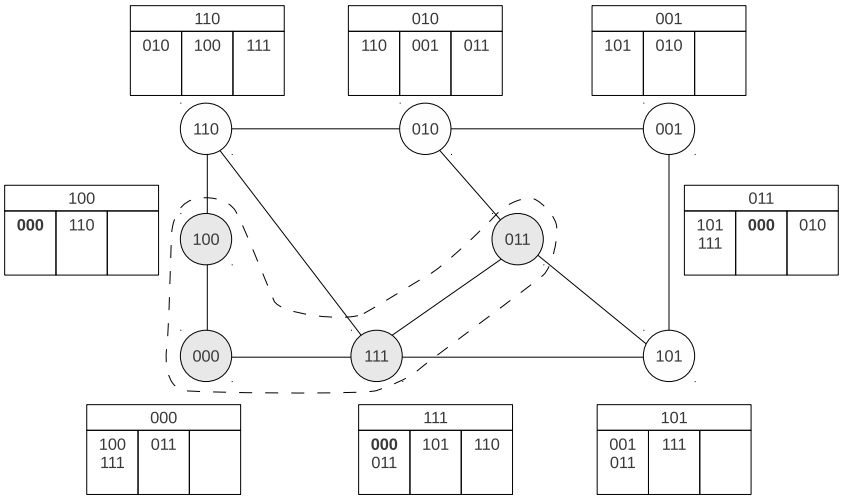


Figure 3.8: Exemplification scenario of the local visibility concept.

In order to analyze the navigability rate of the exemplification network, it is required to ask nodes present in the network to generate packets destined to other nodes available in the network. In this way, traffic is pushed across the network, and the communication between a given pair of nodes is said to be successful in the cases where the packets are delivered. The developed emulation tool (detailed in Appendix A) plays an important role not only in the analyzes of the proposed flat routing mechanism, but also in the development of the mechanism. Basically, the tool is able to load network topologies, instantiating individual threads to act as real nodes of the network topologies. Such nodes have an entire implementation of the proposed flat routing mechanism, exchanging the signaling messages to build their own routing tables according to the protocol specifications. Moreover, the tool allows traffic exchange between nodes available in the network, gathering information about several statistics related to the proposed flat routing mechanism.

Related to the navigability of the exemplification scenario, there are eight nodes available in the network, resulting in a combination of 56 paths in total. The combination is calculated considering that each one of the eight nodes will try to communicate to the other seven nodes present in the network. In this way, as a consequence of the four gaps present in the routing tables of nodes 000, 001, 100 and 101, the current network presents a navigability rate of 71.43%, where 16 paths are not

operational. In order to exemplify the traffic forwarding, Table 3.2 details the paths resultant from the communication initiated at node 000 towards the other nodes available in the network.

Table 3.2: Obtained paths from source node 000 towards all other nodes present in the network.

SRC/DST node IDs	Traversed paths
$000 \rightarrow 001$	GAP
$000 \rightarrow 010$	$111, 011, 010$
$000 \rightarrow 011$	$111, 011$
$000 \rightarrow 100$	100
$000 \rightarrow 101$	$100, GAP$
$000 \rightarrow 110$	$100, 110$
$000 \rightarrow 111$	111

In the results presented in Table 3.2, it is possible to check the occurrence of two gap cases, and five cases of successful paths. Basically, the packets depart from node 000 and are pushed across the network using the information contained in the routing tables of the neighbor nodes, according to the routing process described in Section 3.3. In the cases where the required buckets are empty, the packets are discarded and one gap case is computed. Finally, Table 3.3 brings information regarding all the 16 gap cases obtained in the evaluated scenario.

To conclude this section, the rationale of the proposed scenario is to remove from the routing mechanism the obligation of always delivering traffic, prioritizing the overall system scalability. For the cases where 100% traffic delivery is required, this work proposes the development of complementary services and/or techniques to overcome the gaps occurrence. Obviously, such complementary functions are dependent of the specific scenario where the protocol is instantiated, and one objective of this work is to study how intrinsic aspects of different networks can contribute to traffic delivery.

3.5 Maintenance

As mentioned in Section 3.4, the usage of the proposed *local visibility* concept simplifies the system convergence and, consequently, simplifies the maintenance of the routing mechanism after changes in the network structure. Basically, the insertion and/or removal of nodes due to mobility or failures affect only the region of the network in which nodes are contained.

The insertion of nodes in the network is trivial. First of all, once the node physically connects to the network, it inserts the *physical neighbors* in its routing table, and starts the process of building the routing tables as detailed in Section 3.2. The main characteristic of the proposed mechanism is that

Table 3.3: Gap cases obtained in the exemplification network.

SRC/DST node IDs	Traversed paths
$000 \rightarrow 001$	GAP
$000 \rightarrow 101$	$100, GAP$
$001 \rightarrow 000$	GAP
$001 \rightarrow 100$	$101, GAP$
$010 \rightarrow 000$	$001, GAP$
$010 \rightarrow 101$	$110, 100, GAP$
$011 \rightarrow 001$	$111, 000, GAP$
$011 \rightarrow 100$	$101, GAP$
$100 \rightarrow 001$	$000, GAP$
$100 \rightarrow 101$	GAP
$101 \rightarrow 001$	$001, GAP$
$101 \rightarrow 100$	GAP
$110 \rightarrow 000$	$010, 001, GAP$
$110 \rightarrow 101$	$100, GAP$
$111 \rightarrow 001$	$000, GAP$
$111 \rightarrow 100$	$101, GAP$

the insertion of nodes in the network does not lead to the dissemination of such event in the entire network, preserving the stability of the network due to the unicast nature of the proposed pull-based mechanism. Basically, there is no global correctness requirement, as occurs in the virtual ring.

The removal of nodes, either for mobility or failures reasons, is isolated inside the scope of the region in which the node was located, represented by the nodes present in the routing table of the node which is departing. In this case, one proactive solution, usually feasible in the mobility cases, requires that the node departing from the network sends tear down messages to its neighbors, asking for the removal of the information regarding it from their routing tables. On the other hand, the reactive solution relies on the *physical neighbors* to detect the broken link after the output event. Such detection is based on the periodically exchange of `HELLO` messages to check if the physical connections are still alive. At the instant that the failure is detected, the *physical neighbors* send tear down messages through out their interfaces, in order to remove the deprecated state. The proposed reactive solution is depicted in Figure 3.9.

As detailed in Figure 3.9, the required actions from the nodes which receive the tear down messages and have the node that left from the network on their routing tables include: 1) removing the information regarding the node from the routing table; and 2) forwarding the tear down messages in all the interfaces, except the interface in which the messages arrived. At the same time, the behavior

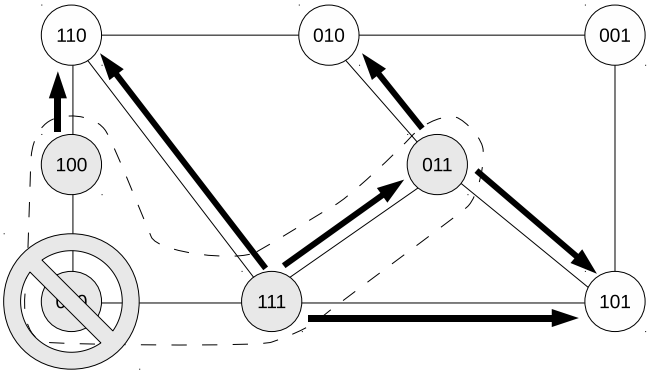


Figure 3.9: Reactive removal of deprecated state in the network.

of nodes which receive the tear down message, but the node that left from the network is not present on their routing tables is to simple discard the tear down messages. In this way, the propagation of the tear down messages achieves the nodes located inside the area where the departing node was located, plus their *physical neighbors*.

After the removal of the state associated with the node that left the network, a second phase is started to remove the remaining deprecated state from the network, which corresponds to the paths that were created passing through the node that left. To this aim, the *physical neighbors* seek on their buckets for entries where the next hop information associated to them match with the node which was removed from the network. For each positive case, tear down messages are also sent according to the procedures detailed in Figure 3.9. Note that although such process is required to avoid incorrectness while forwarding traffic, it is maintained in the network region where the node was located, i.e., it does not affect the entire network as in the virtual ring.

3.6 Summary

This chapter presented the XOR-based flat routing mechanism designed to operate under the *local visibility* concept, the main contribution of this work. This chapter also presented the proposed mechanism for building the routing tables, essential to allow packets forwarding directly on top of flat identifiers, avoiding the usage of underlay networks.

Moreover, the proposed mechanism for building the routing tables, in conjunction with the proposed *local visibility* concept contributes to the overall system convergence. Basically, it prioritizes the insertion in the routing tables of neighbor nodes which are physically near, controlling the exchange of signaling messages. Afterwards, such scenario leads to the occurrence of some buckets with less information than the defined K factor, or even empty at all, causing some gaps in the routing tables. At the same time, the adopted random distribution of flat IDs naturally leads to the

creation of long-distance links between nodes, improving the efficiency of the proposed flat routing mechanism.

The occurrence of gaps influences the overall network navigability, and one objective of this work is to evaluate how the protocol behaves in different network scenarios. In this way, three different network scenarios were investigated and the results are detailed in the following three chapters. The first scenario considers the internal data center networks (Chapter 4), the second scenario considers the *ad hoc* vehicular networks (Chapter 5) and the third scenario considers the inter-domain routing scenario of Internet (Chapter 6).

Chapter 4

Flat Routing in Data Center Networks

Nowadays, innumerable applications demanding massive storage, memory and processing support are executed in large-scale Data Centers (DCs) deployed by companies such as Microsoft, Amazon, Google and Yahoo. To cope with this scale and lower total ownership costs, these DCs use custom software such as BigTable [63], Google File System (GFS) [64], Map-Reduce [65], and customized hardware encompassing servers [66, 67], racks and power suppliers. However, the only component that has not really changed yet is the network, which usually leverages the current available Internet-related routing technologies.

As detailed in [68], there have been a number of network-centric proposals for redesigning DC networking architectures, many of them focusing on fat-tree topologies for scaling IP and/or Ethernet forwarding to large-scale DCs (ten of thousands of servers) [69, 70]. Nevertheless, the "routing" solutions present in these proposals are basically mapping services, in which the IP and/or MAC addresses of the servers are associated to the MAC addresses of ingress, intermediary and egress switches through the fat-tree topology. This scenario achieves scalability by creating a black box totally unaware about the servers inside the DC, delegating the maintenance costs related to changes in the structure of servers to the mapping service, since it is impossible to maintain information about every server in the forwarding tables of switches. In this way, servers and network infrastructure maintain an oblivious relationship in network-centric proposals, increasing the complexity to deploy services in the DCs, and requiring frequent signaling to keep the mapping services updated.

On the other hand, there are server-centric proposals in which servers are equipped with multiple network interface cards (NICs) [34, 35, 36], and routing is performed in the NIC itself in order to provide direct server-to-server communication, i.e., packets are forwarded directly from server NIC to server NIC using commercially available NICs, capable of running computing capabilities [71]. Such proposals make usage of a cube-based topology that offers better path diversity in the network. By having such path diversity, the bisection bandwidth is increased, fault-tolerance is obtained, wiring

complexity and the total cost are reduced, and load balance can be better exploited, potentially providing energy savings [68]. However, the routing mechanism adopted in these proposals rely on a structured addressing scheme, representing the exactly position of all servers inside the cube-based topology in order to forward traffic inside the DC. This approach requires complex configuration mechanisms to set-up all servers, in which the topological organization of the servers is required to assure the proper configuration of the entire DC.

This chapter presents a server-centric DC architecture where the proposed XOR-based flat routing mechanism is used to provide direct server-to-server communication, as an alternative to the scalability problems faced in other network-centric and server-centric proposals. In the proposed scenario, a 3-cube topology is used to connect the servers due to its higher path redundancy, simplicity for wiring and fault resilience. The proposed approach, using the XOR-based flat routing mechanism in conjunction with the 3-cube topology, removes the dependence on traditional switches and/or routers for server-to-server traffic forwarding inside the DC, closing the gap between servers and network infrastructure, and creating a server-aware DC for the deployment of services. Essentially, the main contribution of the proposed XOR-based flat routing mechanism is its capability of performing routing with a small knowledge about the entire network inherent of the *local visibility* concept, providing the required scalability for integrating servers and DC networking infrastructure. Afterwards, the proposed XOR-based flat routing mechanism only requires uniqueness for the flat identifiers (IDs) of servers, being a totally random distribution of IDs the ideal scenario for its operation, i.e., no topological, structured addressing, or semantical IDs organization is required to configure the servers, as occur in other cube-based proposals [34, 35, 36]. In this way, this chapter is focused on describing the internal server-centric DC architecture, detailing the internal server-to-server communication using the XOR-based flat routing mechanism with *local visibility*.

The evaluations presented in this chapter consider 3-cube topologies with 64, 128, 256, 512, 1024 and 2048 servers, all of them were evaluated using the developed emulation tool (details in Appendix A). The results depict the number of signaling messages, the number of entries required in each routing table, the route stretch and the load distribution among the servers in order to forward traffic through the entire DC. Essentially, the results reveal a flexible and scalable routing mechanism, where the number of signaling messages and the size of the routing tables do not grow linearly with the size of the DC network. Such results indicate the feasibility of using this proposal in large-scale DCs (ten of thousands of servers). Furthermore, it also presents a small route stretch and efficient load distribution among the servers in order to perform the task of forwarding packets from server-to-server.

This chapter is organized as follows. Section 4.1 details the proposed 3-cube server-centric DC networking architecture. Section 4.2 introduces the required changes in the XOR-based mechanism

presented in Chapter 3 in order to operate inside DCs. Section 4.3 presents the evaluations of the XOR-based flat routing mechanism running in the proposed DC architecture. Section 4.4 summarizes this chapter.

4.1 Proposed 3-cube server-centric DC networking architecture

The proposed DC networking architecture [72] is composed of a set of servers organized in a 3-cube topology, where servers are distributed in three axis (x, y and z). In such server-centric scenario, the server is the fundamental element in the DC, not relying on the usage of other network elements, such as routers and/or switches in order to provide traffic forwarding between servers, i.e., the servers' NICs can be directly connected. To this aim, each server comprises a general purpose multi-core processor, memory, persistent storage and six NICs named from `eth0` to `eth5`. In order to be settled in the 3-cube topology, each server uses the NICs `eth0` and `eth1` in the x axis, the NICs `eth2` and `eth3` in the y axis, and the NICs `eth4` and `eth5` in the z axis, as illustrated in Figure 4.1.

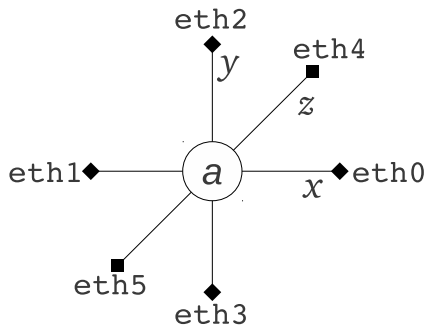


Figure 4.1: How to settle the NICs of the server a in the 3-cube topology.

Figure 4.2 details the wiring process of the proposed DC network, where there are three servers (a, b and c) located in the same row of the DC along the x axis. All the required connections in a certain axis are established using the two interfaces assigned to it, also considering the servers located in the edges as neighbors. According to these procedures, it is possible to check the desired result in Figure 4.2, where `eth0` of server a is connected to `eth1` of server b, `eth0` of server b is connected to `eth1` of server c, and `eth0` of server c is connected to `eth1` of server a, wrapping the x axis to establish this last link. The same pattern is used to wire the y and z axis, where `eth2` is always connected to `eth3` and `eth4` is always connected to `eth5`, respectively.

When compared to other topologies, the 3-cube topology not only offers simpler wiring and higher path redundancy, but also reduces to half the maximum distance (the radius) between the servers. Such significant reduction in the maximum distance is obtained by wrapping the links between the

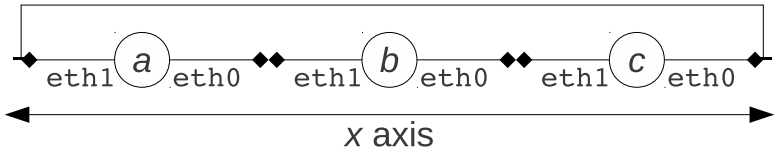


Figure 4.2: Wiring servers *a*, *b* and *c* in the *x* axis.

edge servers, and it contributes to the convergence of the XOR-based process of building the routing tables under the *local visibility* approach. In order to exemplify the resultant wiring between servers, Figure 4.3 presents a 3-cube topology composed of 27 servers. Note in this figure the existence of six NICs in all servers, the linkage between servers in the three axis, and the wrapping connections between edge servers. Such wrapping connections raise the servers located in the surface (edge servers) of the 3-cube to a special condition in the DC, since they are able to invert traffic in the extremities of the axis.

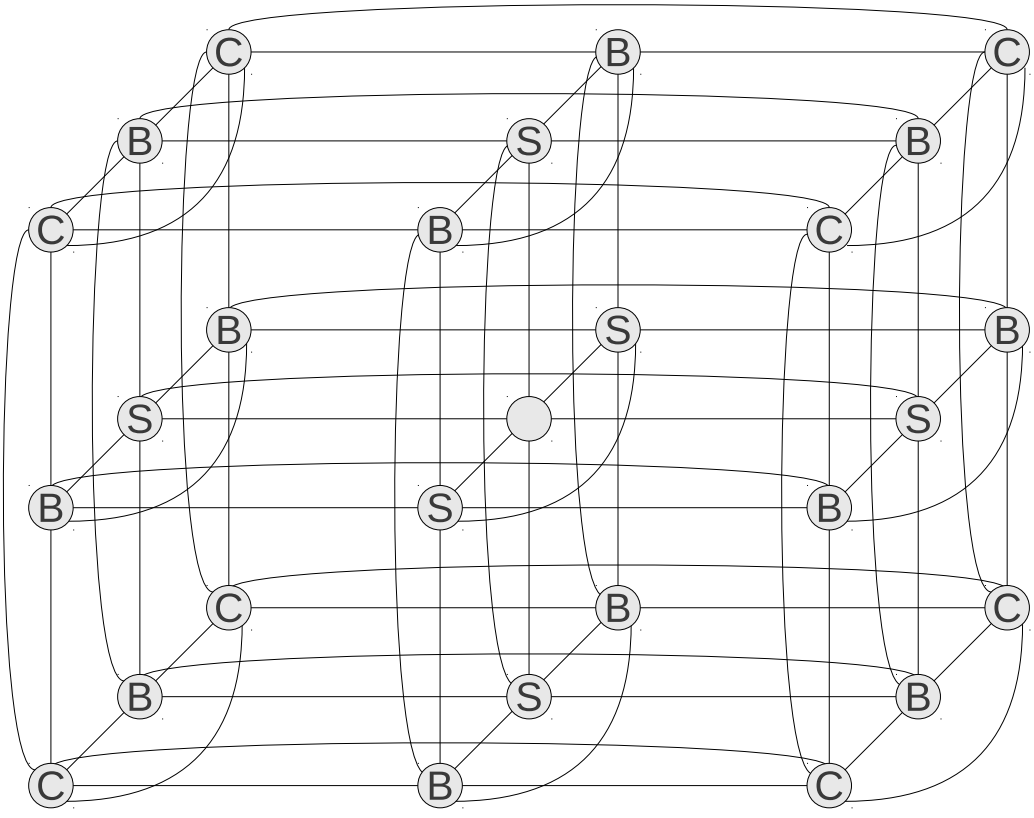


Figure 4.3: Example of a 3-cube topology where the axis are *x=3*, *y=3* and *z=3*.

Essentially, in the server-centric 3-cube topology, there are three different levels of edge servers as seen in Figure 4.3. Such levels are relative to the ability each edge server has to invert traffic along the three axis used in the topology. In the first level are the edge servers located in the inner parts of the

3-cube surface (labeled as S servers in the figure). These servers are able to perform traffic inversion in a single axis. After, in the second level are the edge servers located in the borders (labeled as B servers in the figure) of the 3-cube. These servers are able to perform traffic inversion in two axis. Finally, in the third level are the eight edge servers located in the corners (labeled as C servers in the figure) of the 3-cube. These eight servers are able to perform traffic inversion in the three axis. In this way, when compared to other topologies, the 3-cube topology also offers higher resilience. For example, the DCell [73] network can be partitioned even if less than 5% of the servers or links fail. The server-centric 3-cube topology is able to operate even if nearly to 50% of the servers or links fail. The symmetry and redundancy allow failures in some regions of the topology, without impacting the performance of the remaining servers.

Regarding the DC configuration, there are available in the literature several techniques to bootstrap the configuration of elements comprising the DC, such as servers, routers and switches. For example, Portland [74] uses a hierarchical addressing scheme to assign *Pseudo MACs* to servers according to their position in the DC. It also uses a *Location Discovery Protocol* (LDP) to allow the switches present in the fat tree topology to realize their role in the network, whether they are *edge*, *aggregation* or *core* switches. In this work, due to the proposed XOR-based flat routing mechanism, uniqueness is the only requirement for the flat identifiers used by servers, i.e., none topological adherence, structured addressing scheme, or semantic organization is required for the IDs.

Actually, a total random distribution of IDs suffices for routing with the proposed XOR-based mechanism. Furthermore, switches and routers are not required to provide server-to-server communication, avoiding the usage of special protocols to configure these equipments. However, if such equipments are used, they are intended to transparently operate, offering a network bus connecting servers and reducing the wiring complexity. In this way, one possible solution to automatically configure all servers inside the DC, and to assure the required uniqueness and randomness is to allow each server to choose the smallest MAC address (among its six NICs) to be its flat ID.

In order to provide server-to-server traffic forwarding, this proposal adopts MAC-in-MAC encapsulation. Besides the transparency offered to the applications running on upper layers of the network stack, the option for performing routing at the layer 2 in conjunction with MAC-in-MAC technology has the advantage of fixing the usage of two headers from source to destination, simplifying the definition of the maximum payload size and avoiding fragmentation of packets on the entire path traversed inside the DC. Figure 4.4 exemplifies the traffic forwarding from server *a* towards server *e*. In this figure, it is possible to verify that the inner header carries the server-to-server information (SRC Flat ID and DST Flat ID). This header is preserved during the entire packet transmission and, essentially, it is used by the servers on the path to perform the XOR-based routing

(detailed in Section 4.2). On the other hand, the outer header has only local link scope, being changed in all the links from source to destination. This header carries the MAC address of the NIC through which the server currently routing the packet will forward it, and the MAC address of the NIC present in the next server (next hop) on the path towards the destination, i.e., it is used to transfer traffic between two adjacent servers.

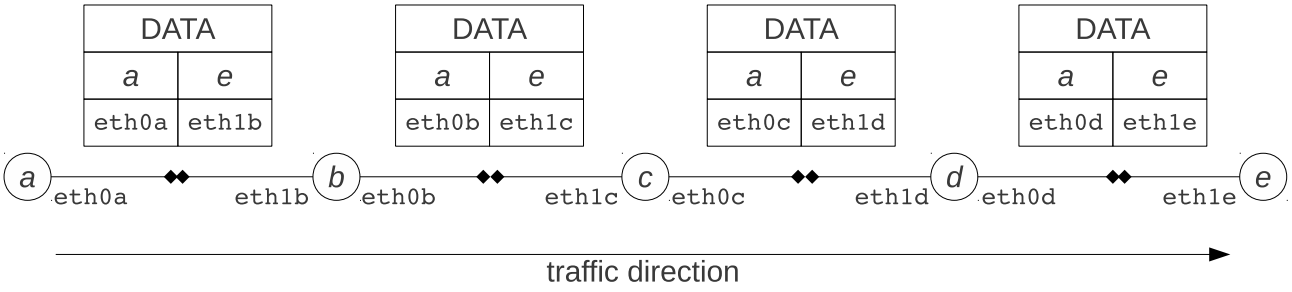


Figure 4.4: Traffic forwarding from server *a* to server *e* using MAC-in-MAC.

Finally, the proposed DC networking architecture predicts that among the servers present in the DC, a fraction of the servers have one extra NIC to forward traffic in and out of the DC (e.g., to/from the Internet). For example, one possible solution for placing these nodes considers the edge servers, once they are able to invert traffic along the axis of the 3-cube topology. In this case, the designer of the DC could start using the servers located in the corners (C servers - 3 axis inversion), after the servers located in the borders (B servers - 2 axis inversion) and, finally, the servers located in the surface (S servers - 1 axis inversion) of the 3-cube topology depicted in Figure 4.3.

4.2 XOR-based Flat Routing Mechanism

This section details the extensions required in the XOR-based flat routing mechanism to operate inside the proposed server-centric DC architecture. Conceptually, the only difference present in the mechanism designed to operate inside the DC, from the mechanism described in Chapter 3, is the need to assure 100% network navigability. Basically, the deployment of services inside the DC requires a routing mechanism capable of providing communication between all pairs of servers present inside the DC. In this way, the *discovery process* in this section leverages the reduced distance between servers offered by the 3-cube topology and is adapted to assure the entire DC network navigability.

In the original protocol, a given node a generates a `QUERY` message describing the amount of information required in each individual bucket using the `QUERY_VECTOR` field, and sends it to its neighbor node b. When neighbor node b receives the `QUERY` message, it is processed and a `RESPONSE` message is returned to node a. However, if node b has no interesting information

to return to node a, it sends an empty RESPONSE message. Conversely, in the DC version of the XOR mechanism proposed in this chapter, the QUERY message receives a new field called DISCOVERY_EXPANSION. This new field carries an integer value defining the number of neighbors that node b is allowed to try to insert in the RESPONSE message, if it is originally empty (no answer that fit in the QUERY_VECTOR specified).

In this way, when node b receives the QUERY message and has no information to return to node a, it checks the value contained in the DISCOVERY_EXPANSION field, and searches in its routing table for neighbors that fit in the buckets of node a where the QUERY_VECTOR field is set to 0 (already fulfilled buckets). The search in the QUERY_VECTOR field follows from right to left side, i.e., it starts from the buckets where the number of servers available in the network is smaller. Essentially, the new *discovery process* gradually expands the visibility of servers requiring neighbors, at the same time that it continues to prioritize the insertion of physically near servers in the routing tables proposed by the *local visibility* concept. Basically, the modification is aimed at introducing at least one existing neighbor per bucket, assuring 100% navigability as done in the UIP connectivity invariant.

Another required expansion is due to the usage of the MAC-in-MAC technology to provide server-to-server communication. In this case, the HELLO message also receives a new field, called MAC_ADDRESS. This field carries the MAC address of the NIC present in the server originating the HELLO message, through which the HELLO message departed towards the *physical neighbor*. Consequently, the following information are associated with each entry of the routing table:

1. the physical distance in number of hops towards the neighbors;
2. the MAC address of the local NIC used to establish the link with the *physical neighbor* (next hop);
3. the Flat ID of the *physical neighbor* (next hop);
4. the MAC address of the NIC present in the *physical neighbor* (next hop).

Furthermore, if server *a* discovers server *b* through different NICs, it can store such redundant information for path diversity purposes. Finally, there is no change in the *learning process* and in the routing process described in Chapter 3 to support the DC scenario.

4.3 Evaluations

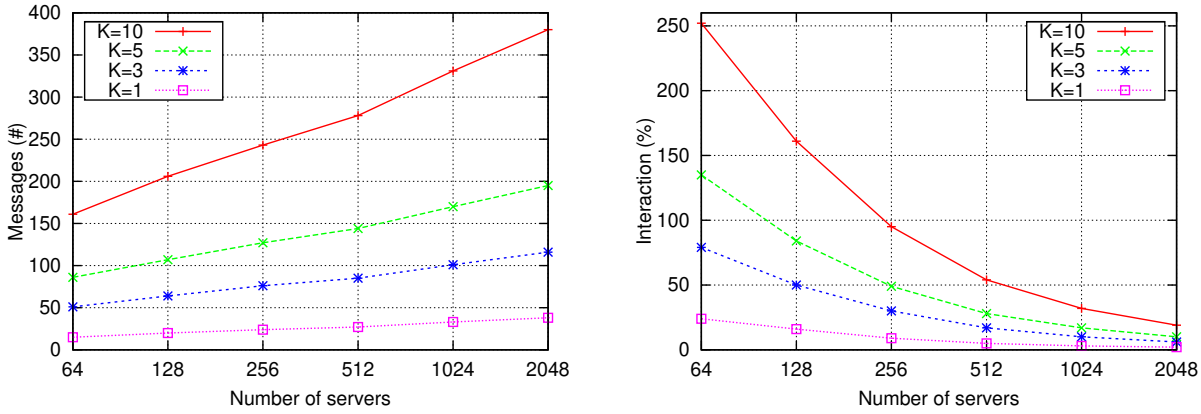
This section presents the evaluations of the proposed XOR-based flat routing mechanism running in the 3-cube server-centric DC networking architecture. The main objective of this section is to

demonstrate the level of scalability achieved by the XOR-based routing mechanism in terms of number of entries in the routing tables. Such information is essential to demonstrate the feasibility of integrating servers and networking, creating a server-aware DC architecture for the deployment of services. As mentioned before, in order to oppose the oblivious approach found in the literature, it is required a routing mechanism able to operate using a reduced number of information about the servers present in the DC. This section also details the required signaling to converge the flat routing mechanism, presents the route stretch incurred in such scenario of reduced routing tables, and depicts the load distribution among servers during the task of forwarding traffic inside the 3-cube DC network.

The evaluations are presented for six networks of different sizes – 64, 128, 256, 512, 1024 and 2048 servers – varying the K factor among the values of 1, 3, 5 and 10, and using the `DISCOVERY_EXPANSION` field always set to 1. The thorough evaluation considers more than 22 million computed paths, i.e., it was computed the full combination of source/destination paths. The developed emulation tool (detailed in Appendix A) has an important contribution in the evaluations, since it offers the possibility of emulating individual servers provided with a full implementation of the XOR-based flat routing mechanism. Basically, each server inside the 3-cube DC is instantiated as an individual thread, being able to perform all the required interactions between the servers, exchanging signaling messages and building routing tables according to the protocol specification. Afterwards, the threads (the emulated servers) are able to transmit traffic inside the 3-cube DC, contributing with the analyzes related to route stretch and load distribution. The tool also includes a topology generator capable of creating 3-cube topologies, considering the required links between servers and assuring the uniqueness and the randomness of the flat IDs of servers. The evaluations presented in this chapter were partially performed in the Amazon EC2 (Elastic Computing Cloud) service [75], running the bigger evaluated topologies (1024 and 2048 nodes) using one instance of a 64-bit machine with 15 GB of memory and eight cores. Basically, the emulation tool is developed to naturally support the remote execution of evaluations.

The first results are presented in Figure 4.5, where it is possible to find information regarding the signaling complexity required to converge the XOR-based flat routing mechanism. Figure 4.5(a) describes the average number of signaling messages generated per server in order to create the routing tables for all the six evaluated topologies. This number represents the exchange of `QUERY` and `RESPONSE` messages, from server-to-server, during the *discovery process*. Such numbers show that as the size of the topologies increases the number of signaling messages also increases.

However, analyzing the percentage of interaction between the pairs of servers available in the topologies, it is possible to verify that the percentage of interaction between the servers does not increase linearly with the size of the topology, as shown in Figure 4.5(b). This behavior has origin



(a) Average number of signaling messages generated per server.

(b) Average percentage of interaction per server in the overall signaling.

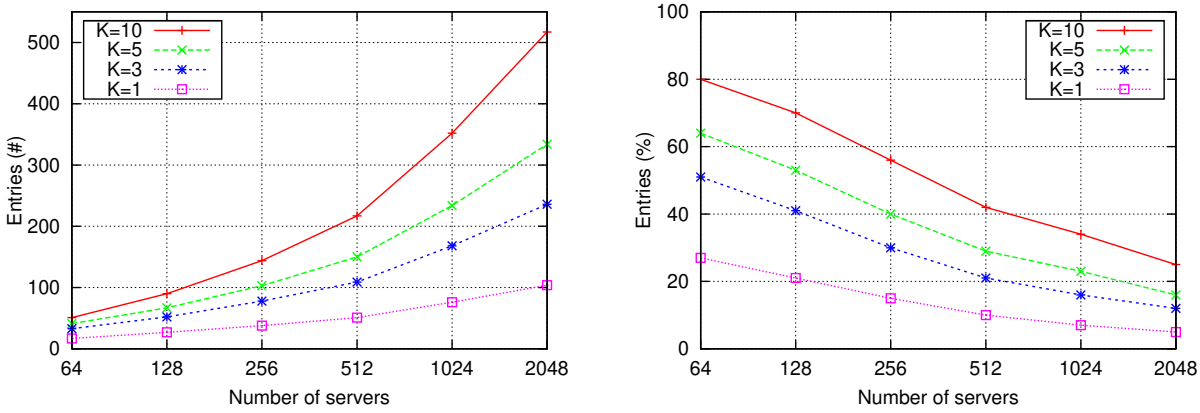
Figure 4.5: Required signaling to converge the proposed XOR-based routing mechanism.

in the proposed *discovery* and *learning processes* designed to extract the maximum information from the signaling messages, and prioritizing the exchange of messages with servers located physically near due to the *local visibility* approach.

The next result detailed in Figure 4.6 presents the average number of entries required in the routing tables of servers. Similar to the results obtained for the signaling messages, the bigger the network the higher the number of entries in the routing tables, as seen in Figure 4.6(a). At the same time, the percentage of routing information required in the routing tables also decreases as the network topologies increase, as presented in Figure 4.6(b). The main reason for this behavior is the proposed XOR-based routing tables which require only a small fraction of the entire routing information available in the network in order to assure 100% traffic delivery. Actually, the presence of one neighbor per bucket suffices for assuring 100% traffic delivery.

The results presented in Figure 4.6 not only demonstrate that the proposed flat routing mechanism in conjunction with the *local visibility* concept is able to operate with a fraction of the overall routing information, offering better control for the rate in which the routing tables grow, but also confirm its ability to integrate servers and networking in a scalable manner, offering the required infrastructure for inserting in the routing tables the information regarding servers. Such condition is ideal for the development of new services totally integrated with the available DC network structure.

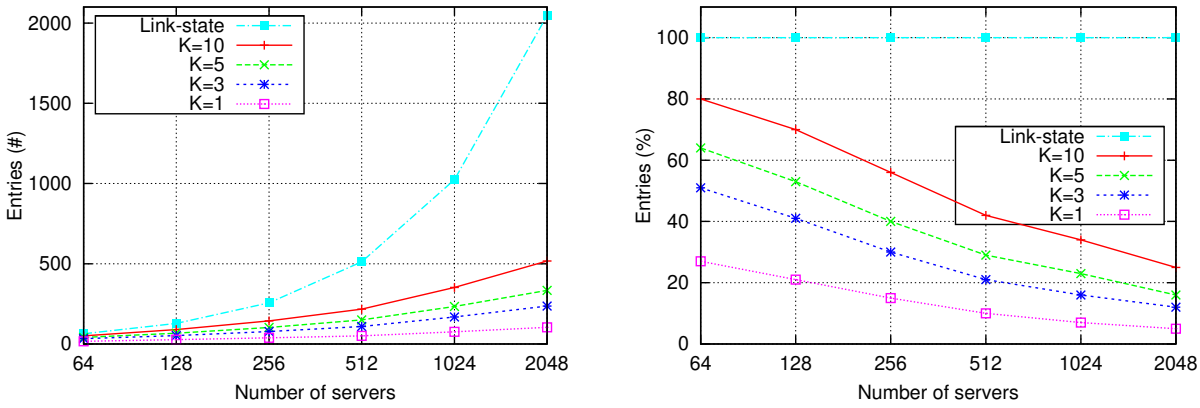
For example, current data center (DC) networks rely on solutions like VLANs, tunneling and/or source routing, creating a scenario where the physical DC network is oblivious to the servers present inside it. In a simplistic way, it is impossible to insert information regarding the totality of servers in the forwarding tables of equipments like switches, i.e., the memory existent in such equipments does not support the entire information. In this way, the achieved results in the proposed DC scenario



(a) Average number of entries in the routing tables. (b) Average percentage of network knowledge per server.

Figure 4.6: Routing tables generated by the proposed XOR-based routing mechanism.

presents an important contribution for the DC networks, fully supporting the integration between servers and the DC network structure. Figure 4.7 compares the obtained routing table results with a link-state routing mechanism, where $\Omega(x)$ routing information is constantly present in each routing table for a network composed of x nodes. In both Figures 4.7(a) and 4.7(b), the most relevant information is related to the rate at which the proposed routing tables grow; it does not linearly follow the amount of routing information available in the network, as occur in the link-state mechanisms.



(a) Average number of entries in the routing tables. (b) Average percentage of network knowledge per node.

Figure 4.7: Comparison of the XOR-based routing tables with a link-state mechanism in the investigated DC scenario.

Normally, based on the small number of entries present in the routing tables, it is expected that the routing mechanism does not find the shortest paths for all the server-to-server communication cases. Basically, the route stretch is associated with the number of entries present in the routing tables, where the higher the number of entries present in routing tables, the smaller the expected route

stretch. Nonetheless, Figure 4.8 details the obtained route stretch values for the proposed XOR-based flat routing mechanism, indicating values close to optimal (stretch 1), despite the reduced size of the routing tables.

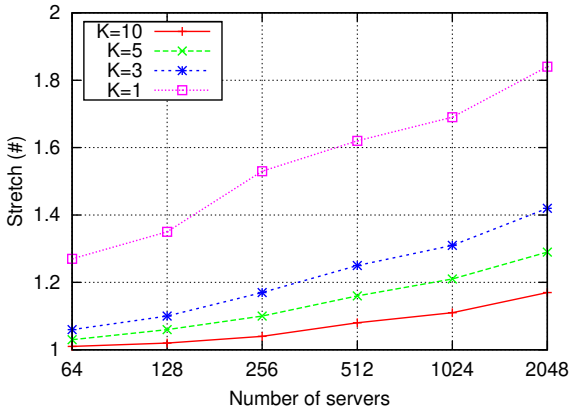
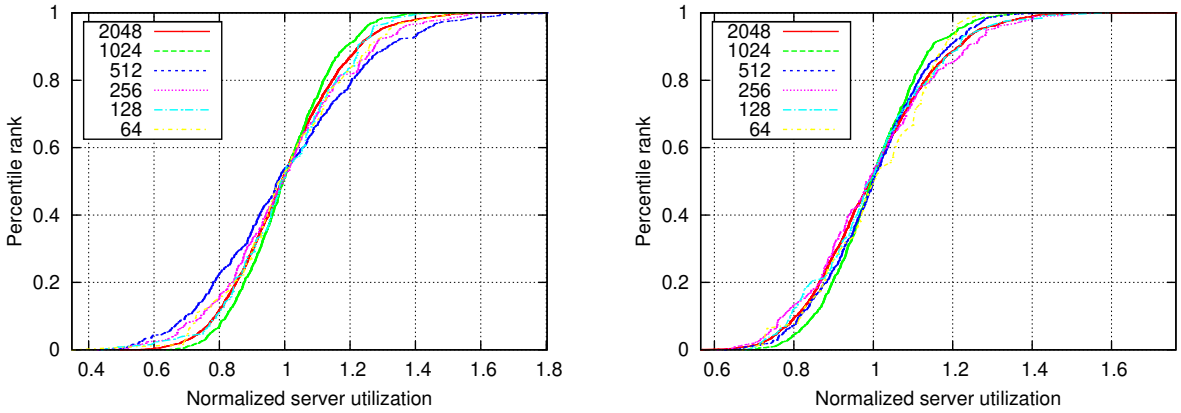


Figure 4.8: Route stretch of the proposed XOR-based routing mechanism.

For example, considering the results obtained for the topology with 2048 servers using $K = 1$, the servers exchange about 40 signaling messages on average (see Figure 4.5(a)), corresponding to the interaction with 2% of the servers, to create routing tables with approximately 100 entries (see Figure 4.6(a)). Such number of entries represents a knowledge about 5% of the entire network, and even with this reduced number of entries in the routing tables, the protocol is able to deliver traffic with an average route stretch around 1.85, not even doubling the length of the shortest paths available. Furthermore, comparing the $K = 1$ results with the obtained $K = 10$ results for the same 2048 servers topology, it is possible to check the routing tables containing about 25% of the entire routing information (approximately 512 entries on each node), which required a signaling exchange of about 380 messages to be created, and contributed to route stretch reduction to the value of 1.17. These numbers of route stretch prove the flexibility for operating with different K values of the proposed XOR-based flat routing mechanism, and its effectiveness to optimize the forwarding process of packets. Such optimization in the forwarding of packets is mainly possible due to the routing directly on top of flat identifiers scenario, which is propitiated by the proposed process of building the routing tables.

The next results demonstrate the load distribution among the servers in order to perform traffic forwarding. Figures 4.9(a), 4.9(b), 4.9(c) and 4.9(d) detail the load for the evaluated topologies using the K factor set to 1, 3, 5 and 10, respectively. The results indicate that approximately 80% of the servers operate at a load level ranging from 0.8 to 1.2, independently of the number of servers present in the DC and of the K factor used, i.e., most of the servers deviate only 20% from the average traffic forwarding load in the analyzed cases. Such numbers indicate the elevated path diversity of

the 3-cube topology, revealing a topology capable of providing not only an efficient scenario for load distribution, but also capable of supporting fault resilience. Afterwards, such numbers prove the convenient coupling between the 3-cube topology and the proposed XOR-based flat routing mechanism, indicating that the scenario of totally random distributed flat IDs is adequate to spread the forwarding load among the servers, not even requiring the usage of other load balancing mechanisms, such as VLB (Valiant Load Balance) and/or ECMP (Equal Cost Multi-Path).



(a) Traffic forwarding load for K=1.

(b) Traffic forwarding load for K=3.

(c) Traffic forwarding load for K=5.

(d) Traffic forwarding load for K=10.

Figure 4.9: Load distribution among all servers using the proposed XOR-based routing mechanism.

Essentially, the XOR-based organization of the routing tables in buckets, where nodes are inserted in the buckets according to the longest common prefix (*lcp*) existent between the flat IDs of servers, provides the fundamental basis for the creation of a load distribution mechanism. The random distribution of IDs naturally balances the insertion of nodes in the buckets and, consequently, it balances the traffic forwarding inside the DC. At the same time, such natural mechanism is enhanced by the proposed process to build the routing tables, since packets are gradually forwarded towards the destination server, and are constantly being deviated through optimized paths, i.e., there is no

occurrence of paths where packets are forwarded through unnecessary servers or passing through the same server twice.

4.4 Summary

This chapter presented a new server-centric DC architecture where the servers are organized in a 3-cube topology. In the proposed approach, the servers are the main elements of the DC, eliminating the need for traditional network equipments such as switches and routers in order to provide server-to-server communication. The main novelty is the integration of servers and networking, done by the scalable XOR-based flat routing mechanism with *local visibility*.

Among the benefits of the 3-cube topology are the higher path redundancy and the simpler wiring. The establishment of the wrapped links between the edge servers significantly reduces the maximum distance between two servers, contributing to the convergence of the proposed XOR-based flat routing mechanism with *local visibility*, and increasing the resilience of the entire DC network. The 3-cube topology, in conjunction with the XOR-based flat routing mechanism, creates an efficient scenario for load distribution of traffic forwarding among the servers in the DC, independent of the size of the topology and of the K factor used.

Once the proposed routing mechanism requires only a fraction of the entire routing information, it provides the total integration between servers and networking, eliminating the concept of a black box network. As presented in the evaluations, as the DC network grows in number of servers, the required signaling and the number of entries in the routing tables do not linearly follow this growth. At the same time, although the routing tables have just a small percentage of information, the paths from source to destination present a small route stretch penalization.

The next chapter presents the instantiation of the proposed XOR-based flat routing mechanism in the scenario of vehicular *ad hoc* networks (VANETs), where the main characteristic is the elevated level of nodes' mobility.

Chapter 5

Flat Routing in Vehicular *ad hoc* Networks

Vehicular *ad hoc* Networks (VANETs) constitute a very active research field, rapidly becoming a reality. Some research programs include the project on cooperative systems within the eSafety framework of the European Union [76], the Intellidrive initiative in the US [77], Smartway, DSSS (Driving Safety Support System) and ASV (Advanced Safety Vehicle) in Japan, simTD in Germany [78] and SCORE@F in France [79]. Standardization has being developed with the activities worldwide in ISO TC204 and IEEE (802.11p and 1609.x), SAE J2735 in the US, ETSI TC ITS and CEN WG278 in Europe and ARIB T-75 in Japan.

Basically, VANETs demand for scalable routing protocols able to provide high packet delivery ratio and low end-to-end packet delivery delay, in order to enable a variety of applications such as safety, traffic efficiency and infotainment. However, VANETs are self-organized networks where the level of nodes mobility is generally high, posing several scalability challenges for the routing mechanism in order to assure traffic delivery [80].

Generally, the routing protocols proposed for VANETs are classified into two groups [38]: 1) topology-based; and 2) position-based protocols. The topology-based class of protocols includes the well known Ad hoc On-Demand Distance Vector Routing (AODV) [81], Optimized Link State Routing (OLSR) [57] and others (DSDV [59], DSR [82], TORA [83] and FSR [84]). These protocols present high bandwidth consumption due to elevated signaling overhead for convergence, specially when the concentration of nodes is elevated, since nodes require information about the most updated network topology to perform traffic forward [85, 86]. On the other hand, the position-based class of protocols includes Greedy Perimeter Stateless Routing (GPSR) [87], Geographic Source Routing (GSR) [88] and others (A-STAR [89] and CAR [90]). Although nodes using these protocols do not need to know the entire network topology [39], they use the location of their neighbors and the location of the destination nodes to determine how to forward traffic. Therefore, these routing schemes require information about the current position of the nodes, which is also a drawback because

of the cost of disseminating this information across all VANET nodes.

In this way, the XOR-based flat routing mechanism proposed in this work provides an interesting alternative for the VANET's scenario, since it relies neither on information about the network topology, nor on the position of the nodes. As described in Chapter 3, it uses a metric based on the logical XOR operation between the flat identifiers of the network nodes, and prioritizes the concept of locality for selecting physically near nodes to insert in the routing tables. This approach has the advantage of requiring reduced routing information for traffic forwarding, contributing to solve the scalability problems faced by VANETs.

This chapter details the instantiation of the proposed XOR-based routing protocol in the high mobility application scenario of highways [91, 92], aimed at providing the support to the deployment of comfort applications (e.g. on board games and video/music file sharing). In order to support such scenario, minor changes to handle the frequent mobility of nodes are incorporated to the XOR-based protocol described in Chapter 3. This modified version of the protocol is referred in this chapter as XOR_1. Afterwards, a second improvement modifies the process of building the routing tables in order to reduce the required signaling overhead and better accommodate the dynamic nature of VANET's topology, being called XOR_2. In this way, this chapter presents initial evaluations of both XOR_1 and XOR_2 protocols with other topology-based routing protocols, characterizing their performance through the comparison of the packet delivery ratio, end-to-end path delay and average number of path hops. The evaluations presented in this chapter were performed using the ns-2 simulator, in a cooperation study with Prof. Dr. Rodolfo Oliveira of Universidade Nova de Lisboa, Portugal.

The remainder of this chapter is organized as follows. Section 5.1 presents the XOR_1 version, describing the adaptations needed in the vanilla version of the XOR-based mechanism in order to handle node's mobility. Section 5.2 details the modifications performed in the process of building the routing tables used in the XOR_2 version of the protocol. Section 5.3 presents and discusses the experimental results. Section 5.4 summarizes this chapter.

5.1 The XOR_1 version

This section introduces the XOR_1 version of the protocol designed for the VANET's scenario. As mentioned before, this version leverages all the details of the proposed protocol described in Chapter 3, introducing minor changes to adapt it to the wireless *ad hoc* scenario of VANETs, where nodes present an elevated level of mobility. In this way, this section mainly introduces the concepts of 1) wireless coverage area, 2) query-range and 3) time-limited entries to build the routing tables.

In wireless networks there is a coverage area in which nodes are able to communicate. According to the *ad hoc* nomenclature, nodes located inside such coverage area compose the set of nodes located

at 1-hop radius. Figure 5.1 exemplifies such scenario, where it is possible to check the coverage area of node a (the dashed circle), with five 1-hop neighbors (black nodes), and three other nodes out of the coverage area of node *a* (light gray nodes). In the *discovery process* designed for the VANET's scenario, a node always knows the nodes located at 1-hop radius, corresponding to the set of *physical neighbors*. Such knowledge about *physical neighbors* is obtained by each node through the broadcast of `HELLO` messages at frequency f_H Hz.

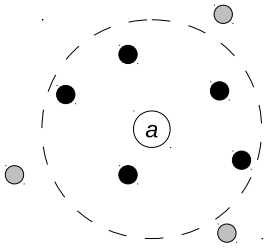


Figure 5.1: The wireless coverage area of node a, and its *physical neighbors* at 1-hop radius.

After introducing the *physical neighbors* in the routing table, the *discovery process* is started to satisfy the defined K value for those nodes that still require some information on their buckets. Similarly to the process described in Chapter 3, the discovered nodes are called *virtual neighbors*, and in the VANETs they are represented by nodes located at 2 or more hops of distance, i.e., *virtual neighbors* are nodes located out of the wireless coverage area of a given node a.

In order to achieve better performance, the *discovery process* was extended to consider a new query-range mechanism which controls the number of hops where `QUERY` messages can be sent. Essentially, `QUERY` messages are limited in range to H hops away from the query's originating node. Figure 5.2 illustrates the query-range approach in which node a is allowed to send query messages to other nodes located at most at 3-hops of distance.

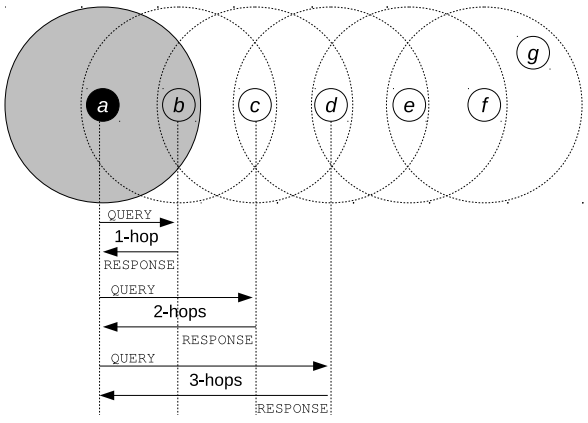


Figure 5.2: The query-range of node a with $H = 3$.

Although the query-range mechanism controls the exchange of signaling messages during the process of building the routing tables, it does not control the distance of neighbor nodes inserted in the routing tables. For example, in Figure 5.2, node a is allowed to send `QUERY` messages at most to node d, but it is possible that node a receives information regarding nodes e, f and g located out of its query-range scope, inserting them on its routing table.

Afterwards, the VANET's scenario requires a time-limited mechanism to control the entries present in the routing tables due to the dynamics of the network structure. In this way, associated with each entry in the routing tables there is a time stamp indicating its moment of insertion. Basically, neighbors are deleted from the routing table after T_β seconds, and if the deletion of neighbor nodes after T_β seconds lead to the occurrence of empty buckets, new `QUERY` messages are sent to try to fill such empty buckets. When the queries sent to all neighbors do not originate any response during the *discovery process*, a node repeats the `QUERY` sending process after T_Q seconds.

Finally, there is no change in the *learning process* and in the routing process described in Chapter 3 in order to support the VANET's scenario.

5.2 The XOR$_2$ version

This section presents the XOR$_2$ version, which is an improvement of the XOR$_1$ version presented in the last section. XOR$_2$ is more suited for high mobile networks, such as VANETs. In the first step of the vanilla process of building the routing tables, a node starts to query all its *physical neighbors* about the required information to fill its buckets. In the following steps, it also queries the discovered *virtual neighbors* if needed, respecting the query-range proposed in the previous section. This approach may decrease the network link capacity, specially in cases where the concentration of nodes is elevated due to the signaling overhead. In this way, the XOR$_2$ proposes the usage of the concept of network stability (detailed in Section 5.2.1) in order to achieve better performance. Basically, the XOR$_2$ maintains the same rationale as the XOR$_1$ algorithm while nodes are considered unstable (detailed in Section 5.2.1), however, it proposes the following change to the process of building the routing tables for nodes that achieve stability:

- once a node achieves stability, it stops querying all *physical* and *virtual neighbors* present in its routing table, and starts queries only a special node called Broadcast Group Leader (BGL), which is selected using a distributed mechanism (detailed in Section 5.2.2).

The diagram of the XOR$_2$ *discovery process* is depicted in Figure 5.3. This practice decreases the network load, specially if BGL nodes are properly selected according to the network motion. Essentially, the BGL nodes are expected to be frequently interrogated during the process of building

the routing tables (if compared to other common nodes), and due to the *learning process*, the probability of BGLs storing more information on their buckets increases. In this way, it is desirable to select nodes which tend to be BGLs for the longest time as possible, in order to increase the performance of their usage. This is explained by the fact that if the selected BGL nodes are often changing, they are interrogated less times during the process of building the routing tables, and consequently have less information present on their routing tables.

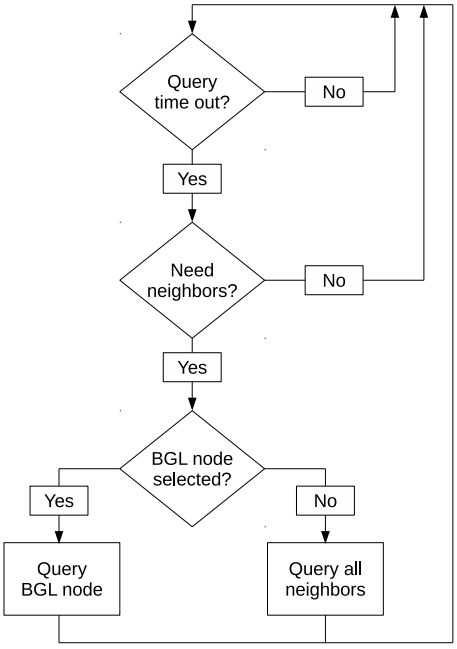


Figure 5.3: Diagram of the process used to generate and send `QUERY` messages in XOR$_2$.

The process for selecting BGL nodes [40, 41, 42] is organized in two main phases: 1) identifying the stabler links available among neighbor nodes in the VANET; and 2) selecting one of the stable nodes to be the BGL. Section 5.2.1 details the mechanism used to characterize the XOR$_2$ concept of stability and Section 5.2.2 presents the algorithm for selecting the BGL nodes.

5.2.1 XOR$_2$ Stability in VANETs

In the time instant $t_i(n_y)$, when the node n_a firstly receives a `HELLO` message from its neighbor node n_y, an unidirectional link is created between the nodes. The link is maintained since n_y periodically transmits `HELLO` messages with period $T_B = 1/f_H$. The duration of the link can be quantified by its stability value: the stability $\eta(n_y)$ measures the duration of the link between the nodes n_a and n_y in number of `HELLO` messages. Since the `HELLO` messages can be lost due to collisions, $\eta(n_y)$ is computed by the node n_a at instant t by applying the following expression[1]

[1] div denotes the integer division operation on integers.

$$\eta(n_y) = 1 + (t - t_i(n_y)) \text{ div } T_B, t > t_i. \tag{5.1}$$

Nodes keep a table of HELLO messages, where the *physical neighbors* are stored and associated with the following information:

- flat identifier n_y of the node originating the HELLO message;
- the stability value $\eta(n_y)$ between both nodes, refreshed at each received HELLO message;
- the flat identifier of the already selected BGL node $\xi(n_y)$ informed by the node originating the HELLO message. This information is contained in the payload of the HELLO message, which is extended to carry a new field called BGL_ID in the XOR_2 version;
- the time instant $t_i(n_y)$ when the first HELLO message was received;
- the time interval $T_O(n_y)$ in which this neighbor is still valid, even though no HELLO messages are received.

When n_a receives the first HELLO message from n_y at instant t_i, the stability value $\eta(n_y) = 1$. After that instant, the stability value computed at instant t is given by the number of T_B periods already elapsed since the instant $t_i(n_y)$, which indicates when the first HELLO message was received. The link is considered broken if no new HELLO message is received within the timeout interval $T_O(n_y)$. On the other hand, a link is considered stable if

$$\eta(n_y) \geq k_{stab}, \tag{5.2}$$

where k_{stab} corresponds to a lower-bound stability value.

In the VANET's scenario, the $k_{stab} = 50$ suffices to identify stability among a set of nodes [93]. In this way, the XOR_2 considers links with $\eta(n_y) < 50$ unstable, and nodes without stable links are considered unstable nodes, do not selecting a BGL node, i.e., they only use the XOR_1 version of the process of building the routing tables. On the other hand, nodes posing at least one stable link are considered stable, selecting their BGL nodes according to the algorithm described in Section 5.2.2. Once the BGL is selected, these nodes stop using the XOR_1 version of the mechanism for building the routing tables and start using the XOR_2 version, reducing the exchange of signaling messages to converge the routing system.

5.2.2 Algorithm for Selecting the Broadcast Group Leader

Representing the *physical neighbors* of n_a by N_a, and admitting that n_a knows the BGL nodes selected by its *physical neighbors*, n_a selects its own BGL $\xi(n_a)$ by applying the Algorithm 1.

```
Input  : N_a, η(n_y) (∀n_y ∈ N_a), ξ(n_y) (∀n_y ∈ N_a), t_i(n_y) (∀n_y ∈ N_a)
Output : ξ(n_a)
1  η_max ⇐ return_max_η_from_hello_table()
2  address ⇐ MAX_INT
3  ξ_aux ⇐ -1
4  transient_threshold ⇐ 1
5  if stable_node(n_a) then                                   /* R1 - if this node is stable */
6      for each neighbor n_y ∈ N_a do
7          insert_sorted(ξ(n_y), list_BGL)                  /* lower addresses in the head of the list */
8      end
9      if (n_a is BGL) then
10         insert_sorted(n_a, list_BGL)
11     end
12     for each ξ_y ∈ list_BGL do                           /* ξ_y is removed from the head of the list */
13         for each neighbor n_y ∈ N_a do
14             if (n_y = ξ_y) and stable_node(n_y) then     /* R4 - select a neighbor that already is BGL */
15                 ξ_aux ⇐ n_y
16             end
17         end
18         if (ξ_aux ≠ −1) then                                  /* BGL already selected */
19             break
20         end
21         if (n_a = ξ_y) then                                    /* R3 - auto-selection */
22             ξ_aux ⇐ n_a
23             break
24         end
25     end
26     if (ξ_aux = -1) then                                 /* R2 - its neighbor becomes a new BGL */
27         for each neighbor n_y ∈ N_a do
28             if (η_max − η(n_y) − transient_threshold ≤ 0) and (n_y < address) then
29                 address ⇐ n_y
30                 ξ_aux ⇐ n_y
31             end
32         end
33     end
34 end
35 ξ(n_a) = ξ_aux
```

Algorithm 1: Algorithm used by the generic node n_a to select its BGL $\xi(n_a)$.

The algorithm rules `R1`-`R4` are summarized as follows:

`R1` - when n_a is unstable (meaning that n_a does not have stable neighbors) it does not select any BGL;

`R2` - when none of n_a's neighbors ($n_y \in N_a$) had previously selected a BGL, n_a selects the neighbor having the smallest address from the set of the neighbors with which n_a has the biggest stability value;

`R3` - n_a selects itself as a BGL when n_a is already a BGL node (previously selected by a neighbor) and the other neighbors' BGL have higher addresses than n_a;

`R4` - when n_a is not selected BGL by its neighbors and there exists at least one neighbor n_y that is already a BGL, n_a selects the node n_y as its own BGL; ties are broken by choosing the smallest neighbor address;

The first BGL node selected in the network is justified by the application of the rule `R2`. The rule `R4` is defined to merge several BGLs selected by different nodes at 1-hop radius. The rule `R3` is also used to merge several BGL in the special case when n_a must select itself as a BGL.

5.3 Simulations

The VANET scenarios simulated in this section were obtained using the Trans tool [94], which integrates the SUMO traffic simulator [95]. The simulations consider a segment of a straight line highway with 3 lanes in each direction, and start with the vehicles moving in both sides of the highway as seen in Figure 5.4. During the simulation more vehicles are launched to maintain a constant density of nodes in the network. The highway segment is 10 km long, which limits the minimum number of network hops to cover the full highway segment to 9, as all vehicles are assumed to have a radio with a coverage area of 1000 meters.

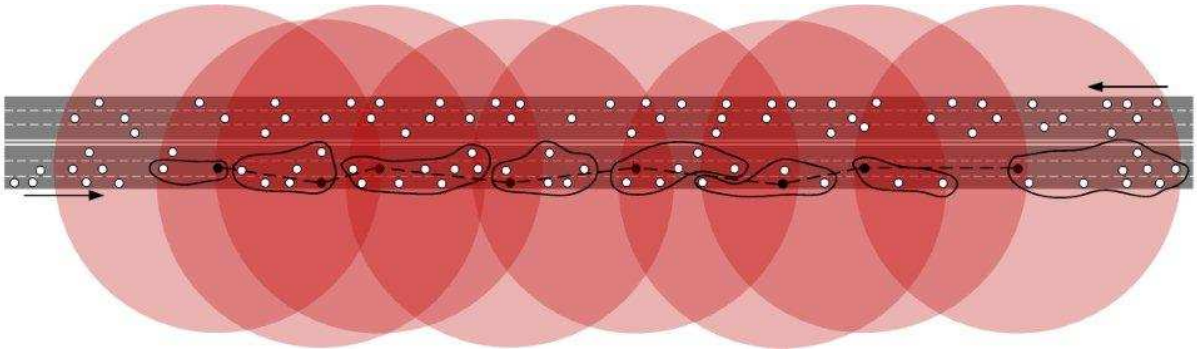


Figure 5.4: Segment of highway used in the simulations.

Three different classes of vehicles were used according to the information present in Table 5.1, and four scenarios were defined with different density of vehicles, denoted by ρ and described in Table 5.2. Independently of the density of the cars, 60% of the vehicles always belong to the class_1, representing the fastest cars; 25% of the vehicles belong to the class_2, representing slower medium sized cars, and 15% of vehicles are of class_3, which represents long sized vehicles such as trucks.

The simulations compare the performance of XOR_1 algorithm (described in Section 5.1) and XOR_2 (presented in Section 5.2) with OLSR, AODV and DSR routing protocols. The simulations used the ns-2 [96] simulator configured with the standard IEEE 802.11[2] [97]. The vehicles moving

[2] 11 Mbps and 2 Mbps were used to transmit unicast and broadcast traffic, respectively.

Table 5.1: Classes of vehicles considered in the simulations.

	vehicle's length (m)	V_{MAX} (m/s)	acceleration (m/s^2)	deceleration (m/s^2)	%
class$_1$	4	27.8 (100 km/h)	3.6	3.6	60
class$_2$	5	26.0 (94 km/h)	2.5	3.0	25
class$_3$	8	20 (72 km/h)	1.5	2.0	15

in the same left-to-right direction generate packets that are routed to another vehicle moving in the same direction. The generation of the packets is divided in two phases:

- in the first phase, a packet is randomly destined to one of the active mobile nodes. This is the packet responsible for the path creation if the routing protocol is reactive;
- in the second phase, after the path had been created, the source node periodically generates packets to the same destination defined in the first phase (the period was defined to 1s). The node stops the packet generation when the original path breaks.

Table 5.2: Vehicles' densities considered in the simulations.

	number of vehicles	average number of neighbors (ρ)	simulation time (s)
Scen$_4$	80	4.0	747
Scen$_6$	120	6.0	727
Scen$_8$	160	8.0	772
Scen$_{10}$	200	10.0	805

The vehicles moving from right-to-left do not generate packets but are able to forward them. The number of packets generated on each density scenario was maintained constant at approximately 3000 packets, and the period of packets generation used by each vehicle is sampled from an exponential distribution with averages 2, 3, 4 and 5 seconds for the scenarios Scen$_4$, Scen$_6$, Scen$_8$ and Scen$_{10}$, respectively. The parameterizations used in XOR$_1$ and XOR$_2$ algorithms were: $f_H = 1$Hz; $H = 5$hops; $T_\beta = 5$s; $T_Q = 5$s; $T_B = 1$s. The simulation results presented in this section were obtained guaranteeing a 95% confidence interval.

In Table 5.3, it is possible to check the packet delivery ratio, which represents the navigability level achieved by the protocols in the VANET's scenario. The obtained level of efficiency achieved by both the XOR$_1$ and XOR$_2$ proposals are comparable to the OLSR protocol, which is a pro-active

topology-based *ad hoc* protocol that requires a full knowledge about the network topology to forward traffic. In this way, the XOR_1 and XOR_2 results are prominent due to the reduced amount of information required in the routing tables, resultant of the proposed *local visibility* concept. Basically, nodes using the proposed XOR_1 mechanisms are controlled by the query-range during the process of building the routing tables, and stable nodes using the XOR_2 mechanism query the BGL nodes, avoiding the search for neighbor nodes in the whole network, and contributing for the system convergence with low overhead. The elevated packet delivery ratio achieved by AODV and DSR are obtained at the cost of flooding route request messages in the network, considerable consuming the overall network bandwidth. As mentioned before, the costs to maintain the most updated topology in all nodes may compromise the network usability in cases where the concentration of nodes is elevated.

Table 5.3: Packet delivery ratio [%].

ρ	XOR_1	XOR_2	OLSR	AODV	DSR
4	43.0	42.0	44.5	65.8	97.7
6	45.3	48.6	46.8	67.9	97.0
8	55.6	52.4	47.5	69.0	97.4
10	49.2	50.3	45.6	71.0	96.9

Table 5.4 presents the end-to-end delay incurred to forward packets from source to destination nodes. For this performance metric, the results evidence the benefits of using BGL nodes in the XOR_2 version, since it has an expressive result when compared to the XOR_1 version. Essentially, the BGL nodes pose more neighbors in their buckets, allowing the forwarding of the packets to nodes which better reduce the XOR distance towards the destination node, i.e., it is possible to find neighbors inside the buckets of the BGL nodes with a longest common prefix (*lcp*) match, optimizing the forwarding of packets by reducing the number of hops traversed form source to destination. The results obtained for XOR_2 protocol are comparable to the AODV protocol, and the smaller delay values of OLSR are justified by the usage of the MultiPoint Relay (MPR) optimization [98]. However, although the MPR optimization contributes for the end-to-end delay reduction, it incurs elevated signaling to update all nodes during topology changes.

Table 5.5 indicates the average length of the paths in number of hops, indirectly indicating the path duration, since the probability of path break increases with the number of hops. The results obtained for the XOR_1 and XOR_2 are comparable to the results obtained by the OLSR protocol. However, nodes using the XOR_1 and XOR_2 protocols have a reduced view about the network, prioritizing the insertion of physically near neighbors in the routing tables because of the *local visibility* concept,

Table 5.4: Path end-to-end delay [ms].

ρ	XOR_1	XOR_2	OLSR	AODV	DSR
4	32.6	5.0	2.7	5.8	180.8
6	20.7	5.0	3.2	8.3	103.8
8	108.4	9.3	3.7	9.3	80.0
10	215.4	18.9	4.3	10.8	107.5

and nodes using OLSR have an entire map of the network, allowing traffic forwarding through the shortest paths. In this way, the obtained results of XOR_1 and XOR_2 are expressive when compared with OLSR, and confirm that the XOR-based routing tables organized in buckets provide an efficient mechanism for traffic forwarding towards the destination node.

Table 5.5: Average Path length [hops].

ρ	XOR_1	XOR_2	OLSR	AODV	DSR
4	2.79	2.73	2.33	3.79	4.88
6	2.72	2.99	2.46	3.92	5.02
8	3.42	3.48	2.43	4.04	5.31
10	3.02	3.32	2.41	4.06	5.38

In general, considering the three performance metrics analyzed, the OLSR protocol obtained the better results, the XOR_2 is in the sequence, showing a slight better performance compared to the AODV protocol, and the XOR_1 obtained similar results to the DSR. In a comparison between XOR_1 and XOR_2, the results show that both protocols have approximately the same packet delivery ratio and path length, but the improvements introduced in XOR_2 considerable contributes to the end-to-end delay reduction. The evaluations presented in this section are still in the initial phase, and further work will explore new features capable of decreasing the end-to-end delay and increasing the packet delivery ratio of the XOR_2 version, such as the use of Bloom filters (used in [99, 100]), and novel schemes to fill and maintain the information in the buckets.

The current results indicate that the concept of *local visibility* proposed in this work constitute an interesting alternative for achieving scalability in the VANET's scenario. Basically, VANET's are extremely dynamic, where the topology is constantly changing, and mechanisms such as topology-based and position-based require an elevated number of signaling messages to allow traffic forwarding. At the same time, the evaluations prove that although nodes have an entire knowledge about the network condition in topology-based protocols, packets are not delivered in 100% of the

cases. In this way, the ideal scenario for VANETs' is to avoid unnecessary exchange of signaling messages in order to save the network bandwidth, trying to maximize the navigability of the networks using reduced routing tables, such as the scenario offered by the proposed XOR-based flat routing mechanism operating in conjunction with the *local visibility* concept.

5.4 Summary

This chapter presented the instantiation of the proposed XOR-based flat routing mechanism in the high mobility scenario of VANETs. Two different versions of the proposed mechanism, the XOR_1 and the XOR_2 versions, were designed to better operate in the challenging scenario of VANETs. The *local visibility* rationale of the proposed mechanism has an important contribution to the VANET's scenario, since it naturally operates prioritizing nodes physically near, avoiding the dissemination of routing information through the entire network. The results presented in this chapter show that while the assessed XOR-based schemes are not achieving the best performance in all the analyzed metrics, it can be efficiently applied to the high mobility conditions of VANETs, specially the XOR_2 version, since its performance is close enough to other analyzed protocols.

Finally, it is important to mention that the work presented in this chapter was developed in collaboration with Prof. Dr. Rodolfo Oliveira of Universidade Nova de Lisboa, Portugal, in the context of the master dissertation of André Gabriel Garrido [101]. As far as we know, this is the first work available in the literature that proposes the usage of XOR-based routing protocols in the VANET's scenario. This work continues to be cooperatively investigated.

The next chapter details the instantiation of the proposed XOR-based flat routing mechanism in the inter-domain scenario of Internet, a large-scale scenario facing important scalability problems, mainly related to the rate in which the routing tables are growing, and the required signaling overhead to converge the routing system.

Chapter 6

Flat Routing in the Internet

The current discussion around the harmful scenario of Internet has raised lots of important questions on the future of the global communication model. Historically, the Internet was conceived to deliver packets in small networks [7, 8], and not designed to carry this such huge amount of data available today, neither to connect about to 5 billion devices in less than 7 years [9]. By observing such growth, one of the first questions that emerges is related to the scalability of the Internet in terms of how efficient the routing system is to keep delivering packets.

Currently, one of the main concerns that has calling attention is the scalability in the Default Free Zone (DFZ), also known as the core of the Internet, due to the fast growing rate of the routing tables. Such growing is mainly due to traffic engineering, multi-homing and the adoption of Provider Independent (PI) addressing in the edge networks [5]. These approaches cause the deaggregation of IP prefixes, adding, consequently, more entries in the routing tables at DFZ.

Among several important points discussed in the IAB document [5] resultant of the IETF RRG meeting in 2006, one of the most relevant was the well known IP overload problem. The IP address acts as both locator and identifier, and makes mobility, nodes renumbering and multi-homing a challenge in the current Internet architecture. Several proposals have addressed such challenge by creating an identity layer for end-hosts [22, 23, 102, 103] as well as for domains [104]. At the same time, the IETF has been discussing some proposals, being LISP (Locator Identifier Separation Protocol) [24] one of the most prominent. However, most of the proposals are still tied to the IP network, heavily dependent of mapping mechanisms to keep track of identifiers and the associated locators.

In this context, this chapter presents the instantiation of the proposed XOR-based flat routing mechanism in the inter-domain Internet routing scenario [105]. To this aim, the proposed scenario considers the usage of flat domain (AS) identifiers to perform route at global scale as an alternative approach for the current IP-based Internet routing system. Nowadays, domains are the entities

responsible for the inter-domain routing mechanism, and also for practices such as multi-homing, traffic engineering and use of Provider Independent addresses. Actually, identifiers of 16-bits are currently assigned to the ASes and are used in the BGP routing mechanism to create the path-vectors towards the destinations (IP prefixes) available in the network. Recently, such AS identification space was extended to 32 bits [106] by the IETF in order to support the expansion in the number of ASes expected in the near future.

Essentially, this work considers domains as "first class citizens" in the Internet scenario. The adoption of the flat identity space to perform flat routing at the inter-domain level creates a scenario in which no topological adherence is required to assign addresses in the Internet. Consequently, the proposed scenario naturally supports new demands such as mobility, multi-homing and avoidance of nodes renumbering, which is currently required after changes in the block of IPs assigned to the ASes. For terminology definition, ASes are the vertex of the graphs used in this chapter, hence node is used to represent an AS (domain) and not an end-host.

As detailed in Chapter 3, in order to obtain the proposed *local visibility* scenario, the process of building the routing tables allows the occurrence of gaps in the XOR-based routing tables. Such gaps provide the fundamental basis for a scalable routing system convergence, removing from the routing system the responsibility of delivering 100% of the traffic. In this way, this chapter is aimed at investigating the navigability level achieved using the proposed XOR-based flat routing mechanism in the inter-domain Internet scenario.

Assuming that it is required to assure 100% traffic delivery in the Internet, the envisioned scenario considers the use of a *reachability service*, which is offered as a complementary service to the XOR-based flat routing system. Such service can be offered by big players, such as tier 1 carriers with global coverage networks and/or companies such as Google and Microsoft. Afterwards, the *reachability service* can be developed in several ways and, in this chapter, it is proposed a mechanism based on the usage of Landmarks [48, 107] and Bloom filters [99, 100] in order to validate the service concept and to demonstrate its integration with the proposed XOR-based mechanism. Basically, the envisioned scenario considers that the *reachability service* is used by those ASes which desire to assure worldwide reachability to their networks.

The proposed inter-domain routing scenario is evaluated in two steps. First, the developed emulation tool is used to evaluate five Internet-like topologies, ranging from 1024 to 16384 nodes (ASes). Such topologies were generated using BRITE [108], Boston university Representative Internet Topology gEnerator, developed at Boston University, and the thorough evaluation collected information of approximately 1,1 billion paths. It was computed 100% of the path combinations for all source/destination pairs of nodes present in the topologies, using the K factor set to 1, 2 and 3. Afterwards, in the second round of evaluations, the real AS-level topology of the Internet, which is

available at CAIDA [49], was evaluated using the developed emulation tool configured with $K = 1$. The evaluation gathers realistic information about the behavior of the proposed flat routing solution in the current Internet. The real topology has approximately 33,000 ASes and more than 10 million paths were computed using randomly selected pairs of source/destination nodes present in the topology.

To conclude this chapter, a brief description of the envisioned Internet scenario is presented, where the proposed routing solution is responsible for the worldwide communication. Although the routing mechanism is the focus of this work, this chapter also presents an alternative to migrate from the current Internet to the envisioned scenario.

The remainder of this chapter is organized as follows: Section 6.1 describes the extensions required in the XOR mechanism to operate in the inter-domain routing scenario of Internet. Section 6.2 introduces a proposal of a *reachability service* composed of Landmark nodes and Bloom filters. Section 6.3 brings the evaluations of the proposed mechanism, where Section 6.3.1 considers the five generated topologies, and 6.3.2 considers the real AS-level topology. Section 6.4 presents the envisioned Internet scenario. Section 6.5 summarizes this chapter.

6.1 Extensions to the XOR-based Flat Routing Mechanism

The version of the XOR-based mechanism used in the Internet scenario only differs in two aspects from the original mechanism detailed in Chapter 3. The first difference consists of an extension in the mechanism of building the routing tables, and the second difference is an adaptation in the routing process in order to cooperate with the proposed *reachability service*.

Regarding the mechanism of building the routing tables, the change encompasses the insertion of a new `PATH_VECTOR` field in the exchanged `QUERY` and `RESPONSE` messages. Basically, the new field is used to append the flat IDs of nodes present in the physical path existent between neighbor nodes during the process of building the routing tables. The information contained in the `PATH_VECTOR` field is associated with the entries present in the routing tables, and such extension enforces the support to current routing policies used in the Internet, which are applied considering the existent paths connecting ASes that are disseminated in the BGP updates.

In the case of the change related to the routing process, the modification is necessary to overcome the occurrence of gaps, supporting the use of the complementary *reachability service*. In short, such modification in the routing process is aimed at assuring 100% of traffic delivery. In this way, once a node forwarding a packet using the pure XOR-based mechanism finds a gap in its routing table, instead of discarding the packet, such node forwards the packet to one of the landmarks that makes part of the *reachability service* in order to continue the forwarding process.

6.2 The Reachability Service

After the execution of the process of building the routing tables by the nodes present in the network, the routing tables are ready to be used in the XOR-based mechanism, and due to the proposed *local visibility* approach, gaps may occur in the routing tables. In order to overcome such gaps, a *reachability service* is introduced as a complementary function to the XOR-based routing system. As mentioned before, such *reachability service* can be developed in several ways, and this section presents an alternative using the concept of Landmark nodes and Bloom filters.

The rationale is that ASes composing the Internet select a landmark present in the *reachability service* in order to assure their global reachability. The selection of which landmark to use can be performed considering several characteristics, such as physical distance and/or costs (e.g. financial costs) to use the landmark; it is also allowed to select more than one landmark. Basically, it is possible to extend the process of building the routing tables to automatically identify the presence of landmark nodes in the network, allowing landmark nodes to advertise their condition using a complementary information in the exchanged signaling messages.

Another option is to offer a list containing the landmarks available in the network in order to nodes select which landmark to use. In this chapter, it is assumed that nodes select only one landmark based on the list describing the landmark options, and the criteria used to select is physical distance, i.e., nodes select the landmark whose physical distance in number of hops is the smallest. In this way, after building the routing tables, nodes create a `REGISTRY` message destined to their landmarks, in order to join the *reachability service*. Such `REGISTRY` message contains the flat ID of the node joining the service and is sent through unicast communication.

Based on such scenario, when a gap is found in the XOR-based routing tables, traffic forwarding starts to be performed using the *reachability service*. To this aim, it is proposed the inclusion of a `LID` (Landmark ID) field and a *flag* in the packet header. Originally, the packet header was composed of two fields, the source ID (`SRC_ID`) and the destination ID (`DST_ID`). The option for inserting the `LID` field in the same header has the objective of preserving the occurrence of path optimizations, which is inherent to the routing on top of flat identifiers approach of this work. As a consequence, although packets start to be forwarded by the *reachability service*, they are not tunneled in an extra packet header.

In a given graph $G = (V, E)$, the landmarks are defined as a set $A \subseteq V$, and it is assumed that each landmark belonging to A has a routing entry to all other landmarks. Consequently, there is an inter-landmark communication level, where information about the nodes registered on each landmark is exchanged. Such exchange of information is done using Bloom filters, and the mechanism used to exchange the Bloom filters can be implemented using a landmark peering session protocol, or any other mechanism. In this chapter, such exchange is done using a `BF_ADVERTISEMENT` message,

which is sent to the landmarks in A via unicast communication. As a result of the Bloom filters exchange, the landmarks in A create a Landmark Information Base (LIB) where the Bloom filters of the set A are stored.

As already introduced in Chapter 3, Bloom filter is a probabilistic data structure which offers a simple interface capable of indicating whether a certain element is inserted on it or not. The main property of Bloom filters is the nonexistence of false negative cases, i.e., if a query for a certain element returns false, it is assured that the element is not a member of the Bloom filter. On the other hand, the probabilistic nature of Bloom filters leads to the occurrence of *false positives*, i.e., it is possible that a query for a certain element returns true, when in fact the element is not present in the Bloom filter.

As mentioned before, the XOR-based routing process needs to be extended in this chapter to operate in conjunction with the proposed *reachability service*. Basically, the modification requires the creation of two different routing functions, one used by common nodes and one specifically for the landmark nodes. Both routing functions are detailed in Algorithms 2 and 3, respectively. In short, the main modification requires that common and landmark nodes check whether the new `LID` field contains the identifier of a landmark or not.

```
    Input : packet
 1  DST_ID ⇐ packet.getDstID()
 2  LID ⇐ packet.getLID()
 3  nHop ⇐ NULL
 4  if (LID=NULL) then      /* packet has being forwarded, so far, using the XOR-based mechanism */
 5      nHop ⇐ routing_table.getNextHOP(DST_ID)
 6      if (nHop=NULL) then                    /* gap case, start to use the reachability service */
 7          packet.setLID(myLID)
 8          forward_packet(myLID)
 9      end
10      else                    /* packet is still forwarded using the pure XOR-based mechanism */
11          forward_packet(nHop)
12      end
13  end
14  else                        /* packet already being forwarded using the reachability service */
15      nHop ⇐ routing_table.findSpecificID(DST_ID)
16      if (nHop=NULL) then               /* forwarding to the landmark specified in the header */
17          forward_packet(LID)
18      end
19      else               /* optimization case, deviating the packet towards the destination */
20          packet.setLID(NULL)
21          forward_packet(nHop)
22      end
23  end
```

Algorithm 2: Algorithm used by a common node to forward a packet.

Figure 6.1 depicts an exemplification scenario used to explain the integration between the XOR-based mechanism and the *reachability service*, and it is also used to exemplify the operation of both Algorithms 2 and 3. In this figure there are three landmark nodes, represented by the light gray circles, whose LIDs are 001, 011 and 101. The other common nodes contained inside the squares

```
    Input  : packet
 1  DST_ID ⇐ packet.getDstID()
 2  LID ⇐ packet.getLID()
 3  nHop ⇐ NULL
 4  landmark_list ⇐ NULL
 5  if (LID=NULL) then      /* packet has being forwarded, so far, using the XOR-based mechanism */
 6      nHop ⇐ routing_table.getNextHOP(DST_ID)
 7      if (nHop=NULL) then                          /* gap case, start to use the reachability service */
 8          landmark_list ⇐ lib.getLIDs(DST_ID)
 9          for each landmark l ∈ landmark_list do
10              packet.setLID(l)
11              packet.setFlag(checked)
12              forward_packet(l)
13          end
14      end
15      else                      /* packet is still forwarded using the pure XOR-based mechanism */
16          forward_packet(nHop)
17      end
18  end
19  else                          /* packet already being forwarded using the reachability service */
20      if (LID=myID) then
21          nHop ⇐ routing_table.findSpecificID(DST_ID)
22          if (nHop=NULL) then
23              if (flag is set) then                          /* false positive case, packet discarded */
24                  packet.discard()
25              end
26              else                                            /* first landmark forwarding the packet */
27                  landmark_list ⇐ lib.getLIDs(DST_ID)
28                  for each landmark l ∈ landmark_list do
29                      packet.setLID(l)
30                      packet.setFlag(checked)
31                      forward_packet(l)
32                  end
33              end
34          end
35          else     /* packet arrived in a landmark where the destination node is registered */
36              packet.setLID(NULL)
37              forward_packet(nHop)
38          end
39      end
40      else
41          nHop ⇐ routing_table.findSpecificID(DST_ID)
42          if (nHop=NULL) then            /* forwarding to the landmark specified in the header */
43              forward_packet(LID)
44          end
45          else           /* optimization case, deviating the packet towards the destination */
46              packet.setLID(NULL)
47              forward_packet(nHop)
48          end
49      end
50  end
```

Algorithm 3: Algorithm used by a landmark node to forward a packet.

represent the nodes registered on each landmark. In the top-left side of Figure 6.1 the "hash function" used to create the Bloom filters is described. Basically, the eight flat identifiers were divided in three ranges which dictate a position in the Bloom filter that must be set to 1. For example, in the case of landmark 001 there are three identifiers registered on it (001, 010 and 100) which, according to the "hash function", results in the Bloom filter 1,1,0 (`Bf110`); the identifier 010 is responsible for

setting the first bit of the Bloom filter to 1 and both identifiers 001 and 100 are responsible for setting the second bit to 1. Note the collision at the second bit, such collisions may be responsible for *false positive* occurrences. In order to explain the operation of the *reachability service*, consider the communication between the following source/destination pairs of nodes: 1) 000 $\rightarrow$ 111; 2) 111 $\rightarrow$ 110; and 3) 110 $\rightarrow$ 100.

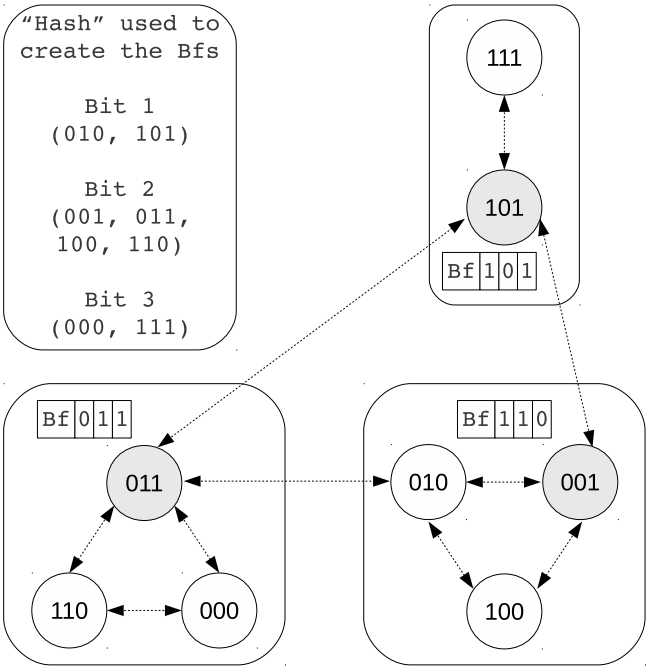


Figure 6.1: Exemplification scenario of the *reachability service*.

In the first example, node 000 generates a packet whose header is composed of `SRC_ID = 000`, `DST_ID = 111` and `LID = NULL`. In the sequence, node 000 starts to forward the packet using the pure XOR-based mechanism, and due to the XOR progress towards the destination node, the packet arrives at node 110. Assuming that node 110 finds a gap in its routing table at the bucket where node 111 should be located, node 110 forwards the packet to its landmark, changing the header to include `LID = 011`, as described in lines 6-9 of Algorithm 2. Once landmark 011 receives the packet, it checks if node 111 is one of the nodes registered on it (verifying if it is present in its routing table in line 21 of Algorithm 3) and, if not, it searches for the identifier 111 (the destination node ID) in the Bloom filters of other landmarks stored in its LIB (line 27 of Algorithm 3). In this case, landmark 011 discovers that node 111 is not registered at landmark 001, since the third bit of `Bf110` is set to 0, but it is possibly registered at landmark 101 due to the third position of `Bf101` which is set to 1. So, landmark 011 rewrites the packet header to `LID = 101`, set the *flag* to indicate that the packet was already forwarded by a landmark and forwards the packet towards landmark 101 (lines 28-32 of Algorithm 3) which, in its turn, delivers the packet to node 111 using lines 35-38 of Algorithm 3.

In the second example, node 111 generates a packet with `SRC_ID = 111`, `DST_ID = 110` and, assuming that node 111 has a gap where node 110 was supposed to be present, it creates the header with `LID = 101`. In the sequence, landmark 101 receives the packet and, as node 110 is not registered on it, landmark 101 checks the LIB and discovers the possibility of node 110 being registered in both landmarks 001 and 011, since the second bit of both Bloom filters (`Bf110` and `Bf011`, respectively) is set to 1. Consequently, landmark 101 forwards a copy of the packet to each one of the landmarks 001 and 011, rewriting the `LID` field and setting the *flag* in both packets. When landmark 001 receives the packet, it discovers that node 110 is not registered on it and immediately discards the packet using lines 23-25 of Algorithm 3; it is a *false positive* case. Note the use of the flag to indicate that the packet was previously forwarded by another landmark node, contributing to identify the *false positive* occurrences. Conversely, node 110 is effectively registered on landmark 011, which delivers the packet using lines 35-38 of Algorithm 3.

Regarding the *false positives* occurrence, it is adjustable by simple varying the length of the Bloom filter array in which the elements are inserted. For example, consider a scenario where 8192 nodes are equally distributed among four landmarks (2048 nodes registered per landmark), and each node identifier is 128-bits long. In the case where a non-compressible data structure is used, each landmark needs to disseminate 262,144 bits to propagate the 2048 identifiers of 128 bits to each one of the other three landmarks present in the network, i.e., the total number of bits propagated is 786,432. On the other hand, accepting the occurrence of 2% of *false positives*, it is possible to insert the same information about the 2048 identifiers of 128 bits in a Bloom filter array of 16,384 bits, representing an overhead compression factor of sixteen, i.e., the total number of bits propagated considering the three other landmarks is 49,152 bits.

It is important to mention three characteristics of the *false positive* occurrences in the proposed *reachability service*: 1) the *false positives* are limited to the portion of the traffic which requires the use of the *reachability service*, exempting the traffic forwarded using purely the XOR-based mechanism of *false positives*; 2) traffic forwarded by the *reachability service* uses the paths existent between the landmarks, limiting the occurrence of *false positives* to such links; and 3) the occurrence of *false positives* does not prevent traffic of being delivered in the proposed scenario; some packets are replicated but are immediately discarded by the landmarks which receive them in the *false positive* cases.

The third example is aimed at demonstrating the occurrence of path optimizations. In this case, node 110 generates a packet with `SRC_ID = 110`, `DST_ID = 100` and `LID = 011`. Once the packet arrives at landmark 011, it discovers in its LIB that the destination node might be registered at landmark 001, forwarding the packet towards it in the sequence. Note the presence of the common node 010 in the path between landmarks 011 and 001. According to the routing function detailed in

lines 14-23 of Algorithm 2, if a common node receives a packet to forward whose `LID` field is filled, and such common node has information regarding the exactly destination node present in its routing table, the common node is allowed to deviated the packet towards the destination node, even before it reaches the landmark specified in the packet header. Otherwise, the common node must forward the packet towards the landmark specified in the `LID` field. Consequently, assuming that node 010 has the information regarding node 100 in its routing table, it is allowed to set the `LID = NULL` and deviate the packet, optimizing its delivery.

6.3 Evaluations

The evaluations for the Internet scenario are organized in two sections. Section 6.3.1 presents results obtained with five different Internet-like topologies generated using the BRITE topology generator. The topologies have 1024, 2048, 4096, 8192 and 16384 nodes, and the thorough evaluation encompasses the full combination of all-to-all source/destination paths, with each topology being evaluated three times varying the K factor to 1, 2 and 3. In this way, the evaluations consider approximately 1,1 billion computed paths. These evaluations were performed at Ericsson Research in Stockholm, Sweden, during an internship of three months, and it was used two HP Proliant DL380 G5 servers with 32 GB of memory and four 3.0 Ghz Intel Xeon cores each.

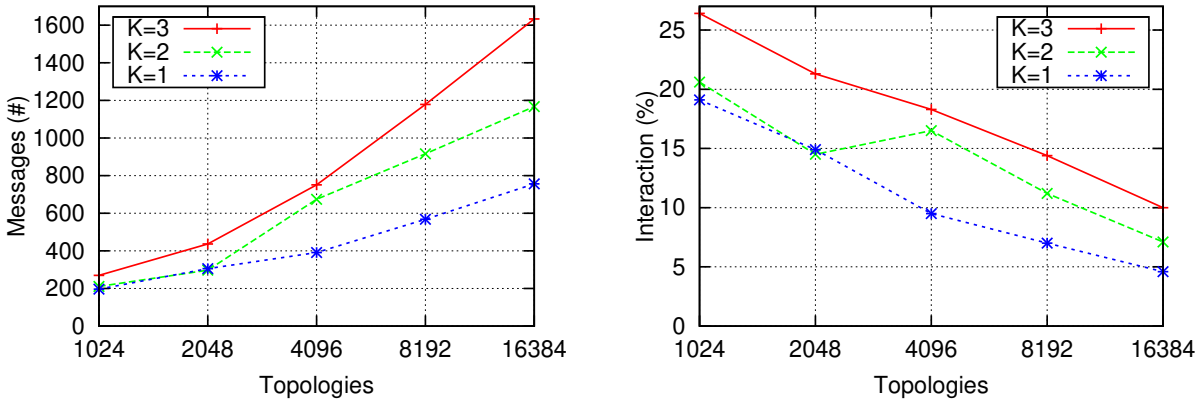
Section 6.3.2 presents the results obtained using the real inter-domain topology available at CAIDA, which is composed of approximately 33,000 nodes. The main objective of this evaluation is to gather realistic information regarding the instantiation of the proposed routing mechanism in the Internet scenario. In this case, it was computed approximately 10 million paths using a random selection of source/destination pairs with the emulation tool configured to use $K = 1$. The results were remotely collected running the developed emulation tool in the HP machines of Ericsson Research from Unicamp.

In both sections, the evaluations are focused in two aspects of the proposed scenario: 1) the operation of the mechanism for building the routing tables under the *local visibility* approach; and 2) the combination between the pure XOR-based mechanism and the *reachability service*. Consequently, the results firstly describe the overall system complexity in terms of signaling overhead, size of the routing tables and route stretch and, in the sequence, it includes results regarding the collaboration between both mechanisms. Specifically for the *reachability service*, the topologies were evaluated using ten landmarks, since we assume that ten represents a reasonable number of tier 1 carriers with global coverage networks offering such service. It is important to mention that the developed emulation tool (detailed in Appendix A) was extended to support the proposed *reachability service* with landmarks and Bloom filters.

6.3.1 Evaluations of the Generated Internet-like Topologies

This section presents the results obtained in the evaluations of the five Internet-like topologies generated using BRITE in conjunction with the developed emulation tool. This section includes results regarding the signaling messages used to build the routing tables, the number of entries present in the routing tables, route stretch and the collaboration between the XOR-based mechanism and the *reachability service* in order to provide 100% network navigability.

Figure 6.2(a) presents the average number of signaling messages exchanged per node in the five evaluated topologies. Such values correspond to the exchange of `QUERY` and `RESPONSE` messages per node during the process of building the routing tables. Essentially, the obtained results indicate that the number of signaling messages increases with the size of the networks and the K factor used.



(a) Average number of signaling messages generated per node.

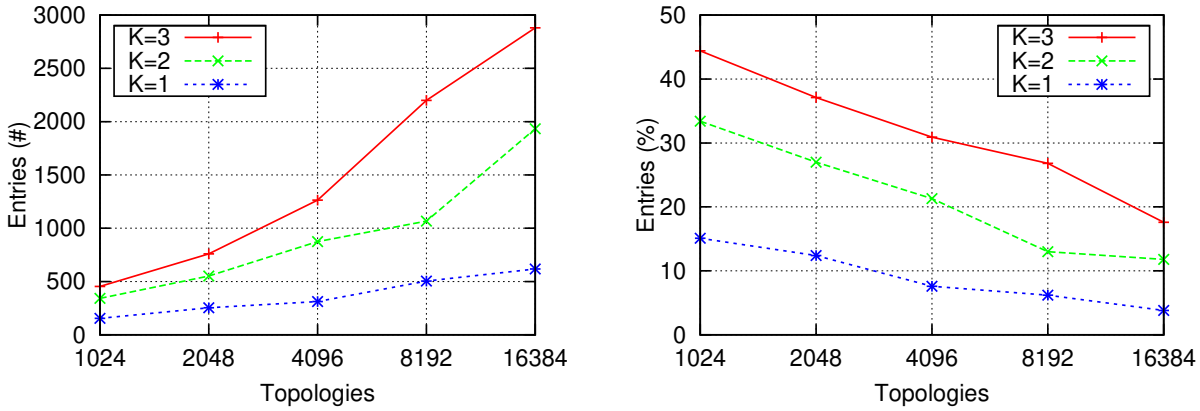
(b) Average percentage of interaction per node in the overall signaling.

Figure 6.2: Results related to the signaling mechanism of the XOR-based proposal.

However, the benefits of the XOR-based routing tables and the *local visibility* approach can be observed in Figure 6.2(b). Analyzing the percentage of interaction required per node during the process of building the routing tables, it is possible to check that the signaling overhead proportionally decreases as the networks grow. For example, in the evaluation case of the network topology composed of 1024 nodes using $K = 3$, nodes exchange signaling messages (`QUERY` and `RESPONSE`) with approximately 26% of the nodes available in the network. Conversely, analyzing the obtained results for the network topology with 16384 nodes using $K = 3$, such interaction is reduced to approximately 10% of the overall nodes.

In Figure 6.3(a) it is possible to observe the average number of entries required per node in each topology. In this case, since the proposed routing mechanism does not require global information about the network, the average number of entries present in the routing tables is considerable below the number of nodes available in the network. Furthermore, observing Figure 6.3(b), it is possible

to see that the proportional amount of information per node also decreases as the networks grow, similarly to the results obtained for the signaling messages.



(a) Average number of entries in the routing tables. (b) Average percentage of network knowledge per node.

Figure 6.3: Results related to the routing tables of the XOR-based proposal.

For example, the results obtained for the 1024 nodes topology using $K = 3$ show that almost 500 entries are required on average per node, representing a knowledge of approximately 45% of the nodes available in the network. However, observing the results obtained in the 16384 nodes topology using $K = 3$, although the number of entries present in the routing tables almost reach 3000 entries, it represents an average knowledge of less than 20% of the nodes. The results are even more significant for the results obtained using $K = 1$, where the topology with 16384 nodes presents a knowledge of approximately 4% of the nodes.

The behaviors presented in both Figures 6.2 and 6.3 are based on the proposed combination of the XOR-based routing tables and the *local visibility* concept. One of the most important properties of such mechanism comes from the manner that the routing tables are structured. For example, in a network topology with 1024 nodes ($n = 10$), the routing tables are organized in ten buckets. In order to double the network size, i.e., to reach the amount of 2048 nodes, it is required only one new bucket in the routing tables ($n = 11$). Consequently, from the system complexity perspective, doubling the size of the network requires neither the double of signaling messages nor the double of neighbors in the routing tables, since the number of buckets required in the routing tables grows logarithmically. Furthermore, the *local visibility* approach prioritizes the insertion of nodes physically near, allowing the occurrence of gaps and contributing for the overall system scalability.

The challenging question in this proposal is related to the level of efficiency that can be achieved using the pure XOR-based routing mechanism, since the proposed *local visibility* approach allows the occurrence of gaps in the routing tables. In this way, Figure 6.4 details the navigability of the proposed routing mechanism. Basically, it brings information regarding the amount of operational

paths achieved using the pure XOR-based routing mechanism (denoted as XOR in the caption), and the proposed *reachability service* developed using landmarks and Bloom filters (denoted as Reachability in the caption).

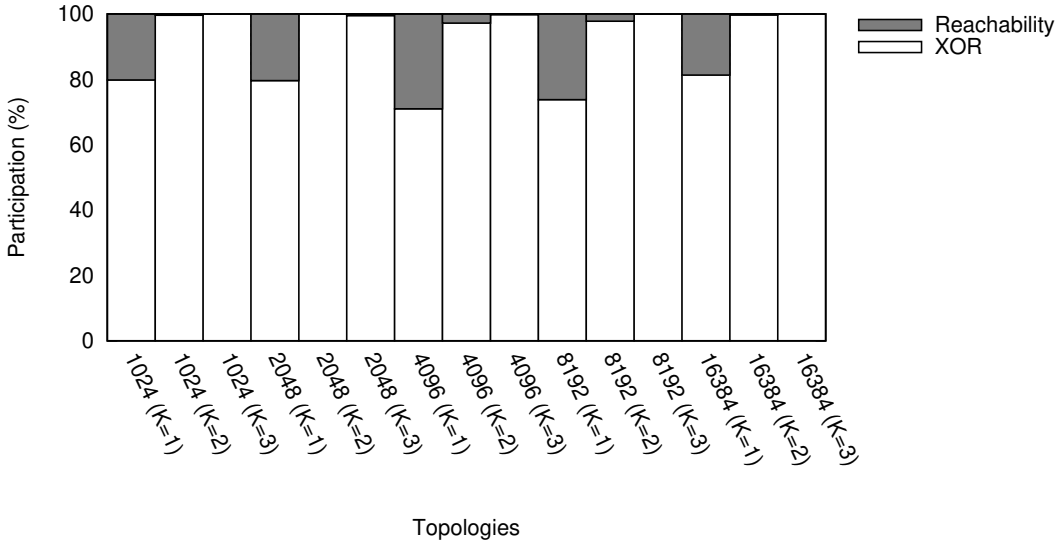


Figure 6.4: Percentage of participation in the overall traffic delivery.

As can be seen in Figure 6.4, independently of the size of the evaluated topologies, using $K = 1$ the level of navigability achieved with the pure XOR-based mechanism ranges from 70% to 80%, resulting in 20% to 30% of paths using the *reachability service*. Moreover, using $K = 2$ or $K = 3$, almost 100% of the paths are always operational using only the XOR-based routing mechanism. Such results prove the correct integration between the XOR-based mechanism and the *reachability service*, assuring 100% of traffic delivery.

As mentioned before, the use of Bloom filters incurs *false positives* due to its probabilistic nature. This section does not present *false positive* results obtained in the evaluations, since the *false positive* rate is adjustable by varying the size of the Bloom filter array. However, it is important to emphasize that the *false positives* occur only over the amount of paths relying on the proposed *reachability service*. In this way, assuming a Bloom filter designed to present 2% of *false positives*, and considering a network scenario where the *reachability service* is used in 20% of the communication cases, the *false positives* will occur in approximately 0.4% of the overall cases.

To conclude, Figure 6.5 presents the obtained path stretch values. In a link-state protocol, nodes have 100% information about the network topology (global knowledge). Consequently, the routing mechanism is able to find the shortest paths (stretch 1 paths) in 100% of the cases. Basically, the route stretch obtained is directly related to the amount of information present in the routing tables. So, it is expected that routing mechanisms with reduced routing tables present route stretch values higher

than 1 (non-optimal paths). However, the obtained results in the evaluations indicate the occurrence of path stretch values near to optimal, despite the reduced size of the routing tables.

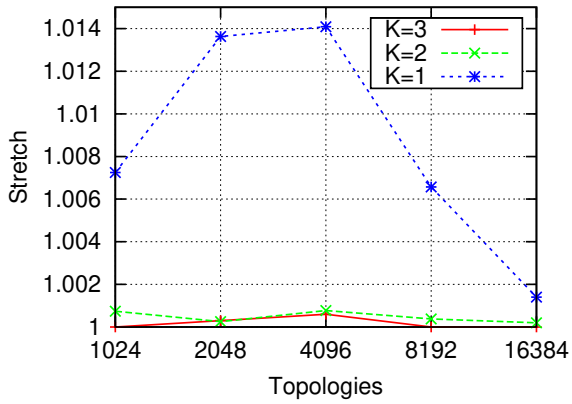


Figure 6.5: Average route stretch values.

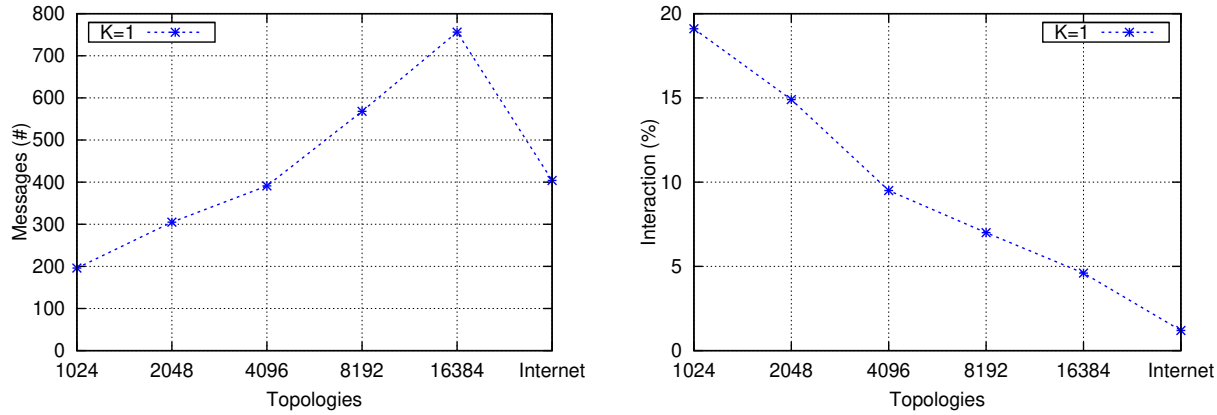
Essentially, the low stretch results are associated with the routing on top of flat identifiers approach, which eliminates the need for an underlay (tunneling) network. In this way, nodes present in the path from source to destination are able to inspect the header in which the destination identifier is present and, in this way, they are always able to take the best forwarding decision available on their routing tables. As mentioned before, such project decision frequently results in forwarding optimizations, including the cases where packets are forwarded using the *reachability service*, since the `LID` field is present in the same packet header.

6.3.2 Evaluations of the Real Internet AS-level Topology

This section presents the obtained results evaluating the real AS-level topology available at CAIDA with approximately 33,000 ASes. The experiments were performed using the real IDs currently assigned to the ASes present in the topology. The results were obtained using the developed emulation tool set to $K = 1$, and the objective is to present detailed information about the use of the proposed routing mechanism in the Internet. In this way, this section not only presents the amount of signaling messages exchanged, the size of the routing tables, route stretch and the navigability of the network, but also provides deeper information about such metrics.

Figure 6.6(a) presents the number of signaling messages exchanged in order to generate the routing tables. As can be seen in the figure, the amount of signaling messages used in the Internet case presents a considerable reduction when compared to the results obtained in the generated topology with 16384 nodes. Actually, the results for the real Internet topology are in the same level of the signaling required in the topology with 4096 nodes. Moreover, in Figure 6.6(b) it is possible to

check that in the real Internet topology nodes interact with approximately 2% of the nodes in order to generate the routing tables.



(a) Average number of signaling messages generated per node.

(b) Average percentage of interaction per node in the overall signaling.

Figure 6.6: Results related to the signaling mechanism in the real Internet topology.

Essentially, the better results obtained in the real Internet scenario are related to the existence of a core region, where tier 1 carriers with networks offering global coverage are present. As can be seen in the Cumulative Distribution Function (CDF) graph of Figure 6.7, the vast majority of the nodes present in the real topology interacts with a reduced number of nodes in order to generate their routing tables, and a few nodes present higher interaction rate. Analyzing the log files generated during the evaluations, there are 391 nodes (1.2% of the nodes) presenting signaling interaction with over than 10% of the overall nodes, including networks of companies such as: 1) Sprint, 2) Level 3 and 3) AT&T.

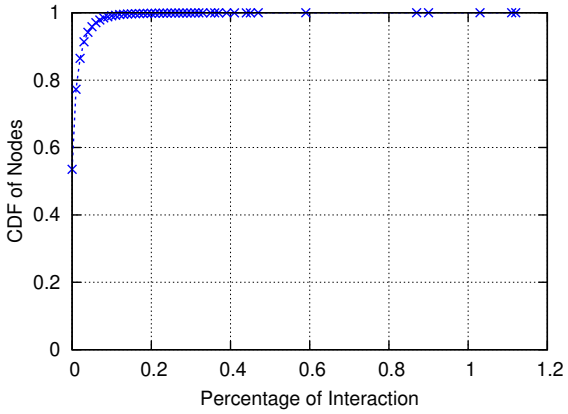


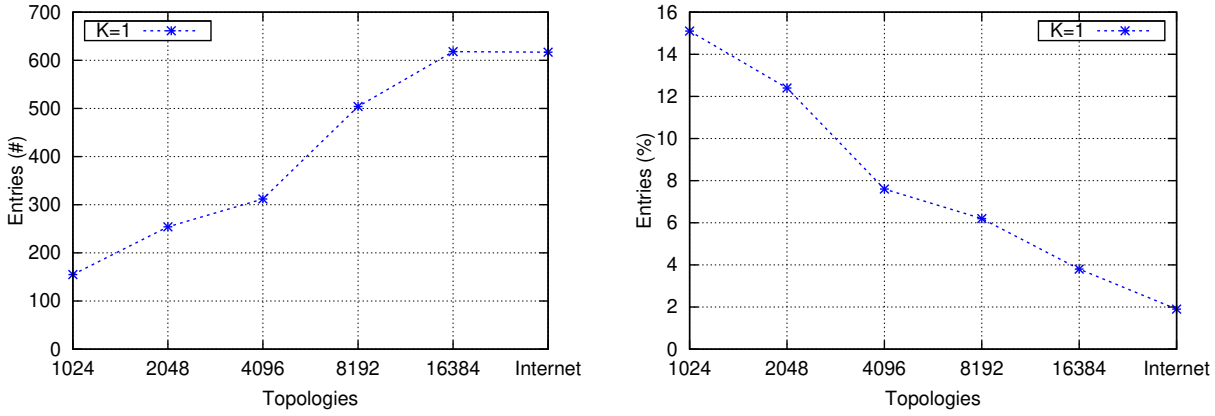
Figure 6.7: CDF of signaling messages interaction.

Based on the hierarchical evolution of the Internet, it is not common to find peering links between

ASes located in the edges of the network. In this way, edge networks are forced to use the network structure offered by the tier 1 carriers, leading to the concentration of signaling exchange in core networks during the process of building the routing tables. The adoption of a flat routing solution contributes for the establishment of peering links at the edges of the current topology, since it does not require the current hierarchical structure to operate. In this way, it supports the creation of a scenario of elevated path diversity, reducing the concentration of signaling in a few tier 1 networks.

Comparing the obtained signaling results with the current Internet routing solution developed using BGP, the main advantage of the proposed mechanism is the pull signaling mechanism for building the routing tables, where nodes use unicast messages to discover information of physically near nodes to introduce on their routing tables. The current BGP mechanism relies in the broadcast of reachability information, flooding the entire inter-domain routing system with BGP updates in order to create the routing tables. In this way, the reduced number of signaling messages of the XOR-based mechanism provides the fundamental basis for controlling the current problems regarding the routing mechanism convergence.

Figure 6.8(a) presents the average number of entries present in the routing tables. As can be seen, the number of entries obtained for the real Internet topology is similar to the number of entries present in the generated topology with 16384 nodes. Such knowledge represents approximately 2% of the nodes, as detailed in Figure 6.8(b).



(a) Average number of entries in the routing tables. (b) Average percentage of network knowledge per node.

Figure 6.8: Results related to the routing tables in the real Internet topology.

As mentioned before, nodes are forced to use the structure offered by tier 1 networks. Such characteristic of the Internet is defined as customer cone [49]. Basically, the lack of alternative paths lead to the creation of a network structure similar to a cone, where a few ASes (tier 1 carriers) cover the vast majority of the existent ASes worldwide. Consequently, similarly to the results obtained for the signaling, it is possible to verify in Figure 6.9 that the vast majority of the nodes have

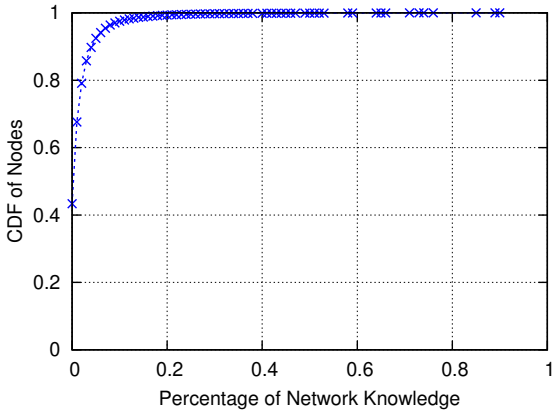


Figure 6.9: CDF of routing table knowledge.

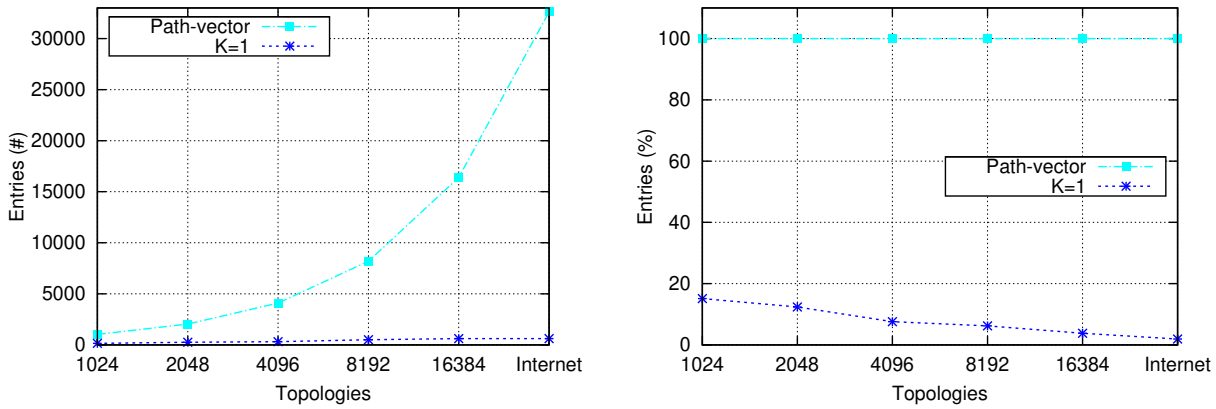
reduced routing tables, but the tier 1 networks have bigger routing tables. The results also reveal that none of the ASes has 100% knowledge. Table 6.1 shows the top ten ASes, presenting information regarding the size of the routing tables generated using the proposed XOR-based routing mechanism, and including information about the number of ASes covered by their customer cone. Information regarding the current customer cone of ASes is available at CAIDA. Note the overlap in the customer cones, there are approximately 33,000 ASes in the topology evaluated, but the top ten ASes present customer cone of similar size.

Table 6.1: Routing table size of the top ten ASes and their respective customer cone size.

Flat AS ID	AS Name	Routing Table Size	Customer Cone Size
3356	Level 3 Communications	29483 (90,38%)	31113
174	Cogent/PSI	29004 (88,92%)	29329
3549	Global Crossing Ltd.	27825 (85,3%)	29036
2914	NTT America Inc.	24678 (75,65%)	26833
1239	Sprint	24089 (73,85%)	29013
1299	TeliaNet Global Networking	23725 (72,73%)	27118
6939	Hurricane Electrics	23279 (71,36%)	27228
701	MCI Communications	23143 (70,95%)	29821
209	Qwest Communications	21435 (65,71%)	28984
7018	AT&T WorldNet Service	20970 (64,28%)	29979

Analyzing the logs, there are 976 ASes (3% of the nodes) with routing tables having more than 10% of the overall nodes. This is one important advantage of the proposed routing mechanism when compared to the current DFZ which needs to keep 100% of the IP prefixes in the routing tables of all ASes participating in the BGP routing mechanism. Nowadays, there are approximately 350,000

IP prefixes [1] per routing table generated using the BGP protocol, and such numbers tends to keep growing, for example, due to the effectively adoption of the IPv6 protocol [109]. It is not possible to compare the current IP-prefix-based routing tables of BGP with the proposed XOR-based scenario, since the proposed mechanism operates at the granularity of domains. However, Figure 6.10 compares the obtained results with an hypothetical scenario where a path-vector mechanism operates at the inter-domain Internet routing system using AS IDs, in order to demonstrate the achieved scalability level of the proposed XOR-based flat routing mechanism.



(a) Average number of entries in the routing tables. (b) Average percentage of network knowledge per node.

Figure 6.10: Comparison of the XOR-based routing tables with a path-vector mechanism in the investigated inter-domain Internet scenario.

As mentioned before, the proposed mechanism has the intrinsic behavior of generating routing tables that do not linearly follow the size of the networks. However, such behavior is more evident in the Internet scenario, where the obtained results are considerable smaller than the hypothetical path-vector scenario. Basically, such results are related to the proposed XOR-based mechanism with *local visibility* that provides the required mechanisms to leverage the Power-law characteristic of the Internet topology.

In the sequence, Figure 6.11 presents the navigability level achieved using both the XOR-based routing mechanism and the complementary *reachability service*. As can be seen, the pure XOR-based mechanism provided the communication in 99.69% of the overall paths evaluated, requiring the use of the proposed *reachability service* in only 0.31% of the cases. Based on such navigability results, it was discarded the need for evaluating the real Internet topology using bigger K values.

It is necessary to remark the reduced signaling messages used to create the routing tables, and the reduced number of entries present in the routing tables which were sufficient to assure the navigability level presented in Figure 6.11. Besides proving that the integration between the XOR-based mechanism and the *reachability service* assures 100% of traffic delivery, the results

indicate that it is possible to efficiently perform routing eliminating the need for global knowledge about the network, as currently performed in the DFZ of the Internet. Essentially, these numbers correspond to the findings of the navigability of complex networks research [46, 47], being closely related to the complex structure of the Internet topology. Such characteristic of the proposed routing mechanism could contribute for solving the current problems regarding the explosive growth of the routing tables.

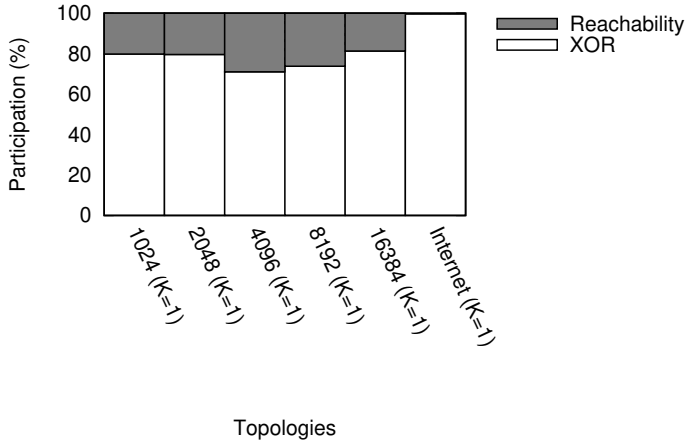
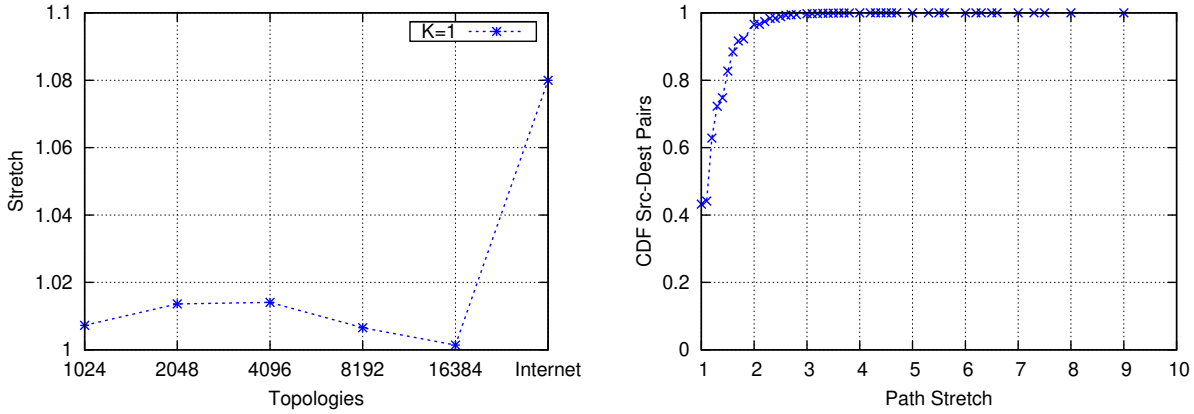


Figure 6.11: Percentage of participation in the overall traffic delivery in the real Internet topology.

Figure 6.12 details the route stretch obtained in the real Internet topology. In Figure 6.12(a) it is possible to verify the average stretch near to optimal, presenting a slight increase when compared to the results obtained with the previous five generated topologies. Afterwards, the results contained in Figure 6.12(b) details the route stretch, where 84 (0.26%) of the paths presented stretch higher than 3. Such behavior is expected given the reduced routing tables generated during the process of building the routing tables, since route stretch is directly associated to the size of the routing tables, i.e., routing tables of link-state routing mechanisms are composed of 100% of the routing information, assuring the shortest path in 100% of the communication cases.

Figure 6.13 details the length (in number of hops) of the paths obtained during the evaluations. As can be seen in the figure, the majority of the paths use 4 or 5 hops (the average value is 4.93 hops) from source to destination, indicating that although the Internet is a large-scale network, the maximum distance between ASes is reduced. In the vast majority of the cases, the packets depart from a small AS located in the edges of the Internet and is forwarded upwards in the customer cone, achieving the core in a few hops. In the sequence, the destination node might be found in the routing tables present in the core, and the packets are immediately forwarded downwards to the destination. Such results are similar to the results obtained in the small world research [45], specially when compared to the Milgram's sociologist experiment in late 60's [110].



(a) Average path stretch. (b) CDF of path stretch.

Figure 6.12: Results related to path stretch in the real Internet topology.

Recent analyzes of the Internet traffic evolution [111, 112, 113] reveal a scenario where the establishment of peering relations at the edge region of the Internet topology is more common. This change is mainly caused by new applications, such as video streaming, offered by content providers like Google and Microsoft. Basically, such content providers are leasing network infrastructure available from carriers, placing their data centers closer to the end-users. The objective is to avoid long distance communications, propitiating a better browsing experience due to data replication. In this way, the Internet topology tends to evolve to a scenario where the hierarchical characteristics will be amended, compromising even more the current IP-based Internet routing mechanism.

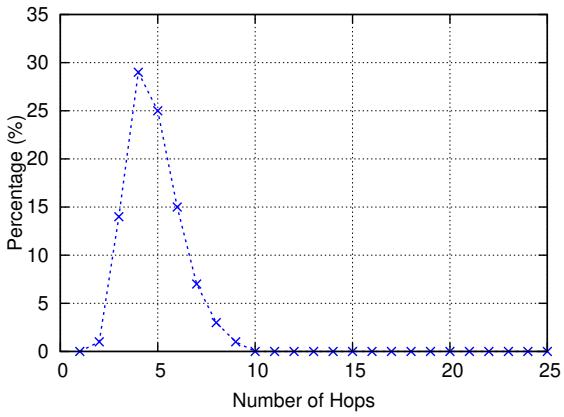


Figure 6.13: Number of hops used to deliver the traffic in the real Internet topology.

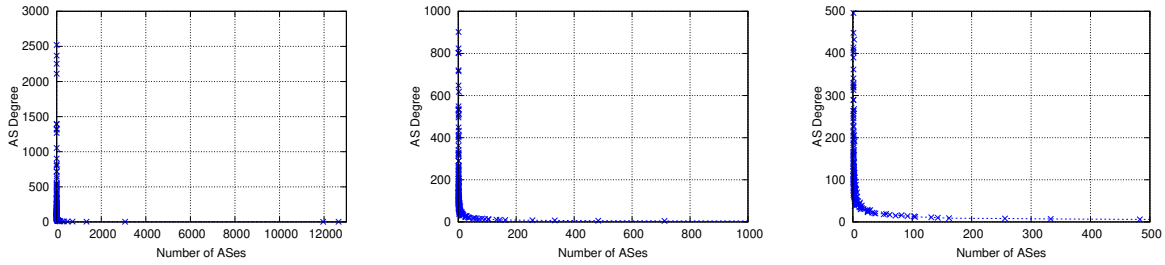
In this context, the proposed XOR-based routing mechanism in conjunction with the *local visibility* approach constitute an alternative to provide the global communication in such flattening scenario being faced by the Internet, since it prioritizes the insertion of neighbors physically near in

the routing tables. Afterwards, the proposed *reachability service* can be offered by the current tier 1 carriers, since they already pose networks with global coverage area. Basically, the *reachability service* can be mainly used to provide the long distance communications, representing an alternative business for the current tier 1 networks, since the current concentration of traffic transportation on their networks caused by the hierarchical Internet topology tends to be reduced.

6.4 Envisioned Internet Scenario

This section briefly presents an envisioned Internet routing scenario based on the proposed flat routing mechanism, providing a speculative discussion about an alternative to evolve from the current Internet towards the envisioned scenario. Basically, in order to provide end-to-end communication, the proposed scenario adopts two flat identity spaces. The first identity space is used at the inter-domain level, according to the scenario previously described in this chapter. Flat AS IDs of 32-bits are assigned to domains composing the Internet and are used to perform inter-domain traffic forwarding. On the other hand, the second identity space is used to uniquely refer to the equipments connected to the Internet, like end-hosts and gadgets. Such identity space can be implemented according to the mechanism proposed in HIP [22], where identifiers of 128-bits are generated using cryptographic keys. For terminology definition, this section refers to the flat identifiers assigned to domains as DID (Domain Identifier), and to the flat identifiers assigned to nodes/equipments as NID (Node Identifier).

Regarding the Internet topology, the envisioned scenario considers a mesh network structure presenting a higher number of path alternatives when compared to the current topology. Figure 6.14 details the current hierarchical Internet topology, presenting two extra zoom images of the current topology focusing the mid-size ASes of the current topology. Basically, Figure 6.14(a) shows the strong hierarchical structure of the current Internet, where a minor number of ASes pose huge network structures with elevated degree (connectivity), and the vast majority present one or two links connecting to the Internet. Such characteristic of the current Internet topology has historical reasons, being directly associated to the IP-based routing mechanism. Conversely, the envisioned scenario motivates the flattening evolution of the Internet, eliminating neither the presence of tier 1 carriers, nor networks with a single connection, but 1) increasing the number of existent medium sized ASes (like ASes composing Figures 6.14(b) and 6.14(c)) and 2) increasing the connectivity level between them. Consequently, the envisioned scenario is still Power-law, but less heavily tailed to tier 1 carriers as nowadays. Although it was not evaluated, we believe that the existence of elevated alternative paths tends to increase the route stretch in the Internet scenario, as observed in the evaluations with the data center scenario of Chapter 4.



(a) Entire topology. (b) Zoom view 1. (c) Zoom view 2.

Figure 6.14: Current Power-law Structure of the Internet topology.

As mentioned before, the proposed inter-domain flat routing scenario naturally supports practices such as (network) mobility and multi-homing, since there is no need for topological adherence while assigning the flat identifiers. In this context, the proposed scenario totally separates physical network connectivity from the routing mechanism, introducing the concept of connectivity service. The concept of offering network resources as services is aligned with the proposal found in [114]. In the proposed connectivity service, carriers are intended to purely offer physical links to ASes composing the Internet topology, i.e., there is no assignment of addresses (IP blocks) involved in the connectivity service. In this way, ASes can select which carrier to use based on their needs, such as costs and/or network resources offered, allowing ASes to naturally use more than one carrier (multi-homing) and/or change their carriers (mobility). Furthermore, current routing policies used in the BGP inter-domain routing system can be enforced by the proposed XOR-based mechanism, since the proposed mechanism for building the routing tables can be extended to consider policies while inserting information in the `RESPONSE` messages.

Based on the adoption of two flat identity spaces, the routing principle of the proposed scenario is to provide end-to-end communication using NIDs, and to perform inter-domain traffic forwarding with the proposed XOR-based flat routing mechanism using DIDs. In this way, it is assumed that nodes (NIDs) are present in at least one domain (DID) in order to provide the end-to-end communication. Relative to the internal network of ASes, the proposed scenario allows the use of any network technology, creating a scenario of heterogeneous domains [104]. For example, it is possible to create ASes where the internal networks use IPv4, IPv6, Ethernet or any other technology, as shown in Figure 6.15.

In the envisioned scenario, there is a Name Service responsible for maintaining information regarding the FQDNs and their respective tuples composed of <NID,DID>. Although NIDs can change their DIDs, for example for mobility reasons, the association between two identifiers is stabler than the association between an identifier and a locator, as presented in the ID/Loc separation proposals available in the literature [22, 24, 23]. Essentially, the flat routing mechanism presented

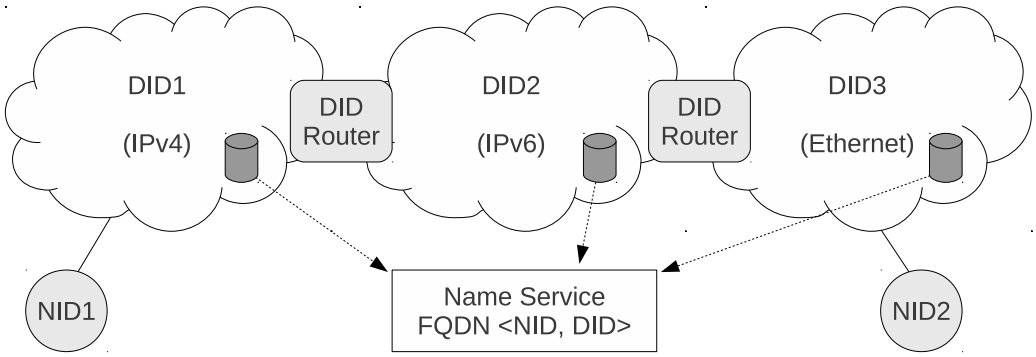


Figure 6.15: Envisioned Internet Scenario.

in this chapter is responsible for the correct maintenance under dynamic conditions (for example, mobility and failures) at the level of domains, and the envisioned Name Service is responsible for the maintenance at the level of nodes.

Based on the proposed name service, packets depart from source nodes containing the flat identifiers required to provide the end-to-end communication. In this way, assuming the routing solution proposed in this chapter, where the *reachability service* operates in conjunction with the XOR-based routing mechanism, the packet header comprises five fields: 1) the source node identifier (`SRC_NID`) of 128 bits, 2) the destination node identifier (`DST_NID` of 128 bits, 3) the source domain identifier (`SRC_DID`) of 32 bits, 4) the destination domain identifier (`DST_DID`) of 32 bits and 5) the landmark identifier (`LID`) of 32 bits.

For example, considering the communication between nodes NID1 and NID2 in Figure 6.15, NID1 generates a packet where `SRC_NID = NID1`, `DST_NID = NID2`, `SRC_DID = DID1`, `DST_DID = DID3` and `LID = NULL`. In the sequence, since both nodes are located in different domains (DID1 and DID3), NID1 forwards the packet towards its default `DID ROUTER` using the internal IPv4 technology, in order to start the inter-domain routing towards the destination node NID2. The `DID ROUTERs` are responsible for the inter-domain traffic forwarding, according to the flat routing mechanism proposed in this chapter, and are also responsible for inter-domain network technology translation, implementing both the required network stacks (IPv4, IPv6, Ethernet and others).

Essentially, the envisioned scenario comprises a set of changes in the current Internet, including flattening the network topology, changing the routing system and upgrading the end-hosts' stack, i.e., it constitutes a clean slate proposal, which prevents its immediate adoption. However, it is possible to gradually evolve from the current Internet towards the proposed scenario, allowing the coexistence of both scenarios. In this way, the remainder of this section briefly presents some speculative steps towards the envisioned scenario.

In a first moment, the evolution requires the establishment of a set of domains where the proposed flat routing mechanism is implemented. Such set of domains can start including the tier 1 carriers and be gradually expanded to include other domains. Basically, by using the tier 1 domains, it is possible to immediately offer global coverage using the new flat routing scenario, even tough the number of networks using the proposed routing mechanism is reduced. Afterwards, we consider as a natural evolution that tier 1 carriers offer the *reachability service*, constituting an alternative business to such ASes and justifying the option for starting the migration using such networks. The implementation of this phase requires the deployment of network infrastructure, including DID routers to provide traffic forwarding and DID proxies to assure the coexistence with the current Internet, as detailed in Figure 6.16.

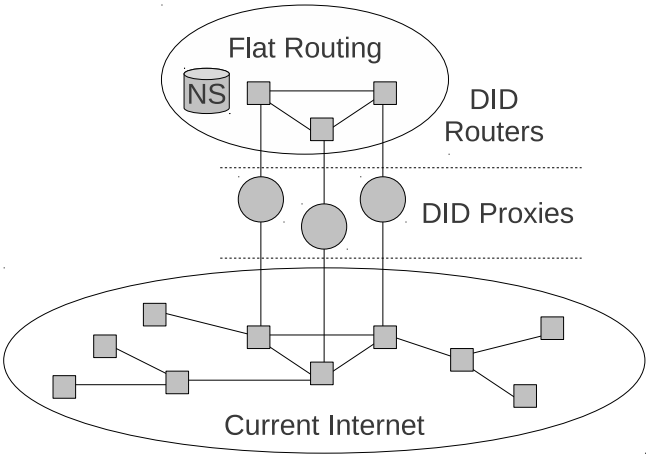


Figure 6.16: Network infrastructure required to start evolving towards the envisioned scenario.

The objective of using DID proxies is to provide inter-communication between legacy end-hosts using the current IPv4 or IPv6 network stacks and end-hosts using the proposed flat routing stack, and vice-versa. In order to allow such inter-communication, there are two possibilities including a network-based solution and an end-host-based solution. In the network-based scenario, end-hosts have their legacy network stack preserved, and all the required patches are applied in the network to transparently provide the inter-communication. We believe that the network-based migration can be performed using the LISP [24] proposal of IETF, extending it to consider the proposed flat routing mechanism. Basically, the ingress tunnel routers (ITRs) and egress tunnel routers (ETRs) of LISP can be implemented by the proposed DID proxies. Conversely, in the end-host-based scenario, the adoption of patches also affects the end-hosts but, in this case, the workload over the DID proxies is minimized.

In the end-hosts-based solution, the current Internet becomes a special IPv4/IPv6 AS (or a set of ASes), where DID Proxies represent a DID (or a set of DIDs), providing the communication

between both scenarios. In this way, the communication cases originated in the current Internet require tunneling the packets in IPv4 or IPv6 headers addressed towards a certain DID Proxy, which is responsible for injecting such traffic in the flat routing region. On the other hand, packets originated in the flat portion of the Internet are firstly routed towards the DID Proxy using the proposed flat routing mechanism, and delivered to the end-hosts tunneled across the legacy Internet segment. Basically, in the network-based solution it is required to offer a NAT-like mechanism at the DID Proxies, since the legacy nodes are not capable of interpreting the proposed packet header with all the five identifiers.

6.5 Summary

This chapter presented the instantiation of the proposed XOR-based flat routing mechanism at the inter-domain level of the Internet. Essentially, the proposed inter-domain routing is performed directly on top of flat ASes identifiers, effectively introducing domains in the routing system by considering them as "first class citizens". The proposed combination between the XOR-based routing mechanism and the *local visibility* concept provides efficient control to: 1) the size of routing tables and 2) the routing system convergence. Afterwards, the adoption of the *reachability service* assures 100% navigability of the network, and represents an alternative business to current tier 1 carriers.

The evaluation using the real Internet topology, with approximately 33,000 ASes, provide detailed information about the adoption of the proposed routing mechanism in the Internet. Basically, as alarming as the current explosive growth of the routing tables, the convergence is also pointed as one important scale-limiting factor of current routing mechanisms. Since the current Internet routing solution requires 100% of the routing information (global knowledge) in the DFZ, a change in a given region of the network requires messaging throughout the DFZ in order to converge to the updated condition. Consequently, a routing mechanism which requires only local messaging (not global convergence), and local routing information (not global knowledge) in the routing tables, becomes a promising research topic to tackle some problems so far identified in the Internet.

The next chapter concludes this work, detailing the main results obtained in the tree instantiation scenarios of the proposed XOR-based mechanism, and presents some open issues left to be investigated as future work.

Chapter 7

Conclusions

This work presented an XOR-based routing proposal which performs flat routing directly on top of flat identifiers. Besides the XOR proximity between flat identifiers, the proposed routing mechanism introduces the concept of *local visibility*, which prioritizes the insertion of physically near nodes in the routing tables in order to create a mesh network organization. The proposed scenario totally integrates the flat identity space with the physical network structure, contributing to several challenges faced in the investigated scenarios.

The approach of routing directly on top of flat identifiers was introduced by IBR but, as mentioned before, the virtual ring structure requires the global ring correctness in order to provide traffic forwarding. Basically, if a given node present in the virtual ring does not maintain the required relations with successor and/or predecessor nodes, it leads to the virtual ring partition, impacting in the overall traffic forwarding. Conversely, the proposed XOR-based routing mechanism with *local visibility* allows the occurrence of empty buckets, and even in this case it is able to perform traffic forwarding. As introduced in Chapter 3, in a n-bit identity space, each individual bucket $\beta_i, 0 \leq i \leq n-1$ of the XOR-based routing tables represents $\frac{1}{2^{i+1}}$ nodes from the overall 2^n nodes, amending the impact of gaps in the routing tables, specially in large scale scenarios, since the much longer the flat identity space used (the higher the n value), the lower the impact of the occurrence of empty buckets in the right positions (less significant bits) of the routing tables.

According to the evaluations presented in this work, it is possible to verify some important properties of the proposed routing mechanism regarding signaling complexity, route stretch, load distribution, traffic delivery ratio (navigability) and end-to-end delay, all of them observed by using the developed emulation tool. However, the main characteristic of the proposed flat routing mechanism is related to the routing tables. Essentially, the proposed XOR-based routing, in conjunction with the concept of *local visibility*, operates with a reduced number of entries per routing table, providing the fundamental basis for controlling the rate at which the routing tables grow,

specially in large-scale networks.

As discussed in Chapter 4, current data center (DC) networks rely on solutions like VLANs, tunneling and/or source routing, creating a scenario where the physical DC network is oblivious to the servers present inside it. Basically, it is impossible to insert information regarding the totality of servers in the forwarding tables of equipments like switches due to memory constraints of such equipments. In this way, the achieved results in the proposed DC scenario presents an important contribution for the DC networks, fully supporting the integration between servers and the DC network structure, once the XOR-based routing tables do not linearly follow the amount of routing information available in the network, as occur in the link-state mechanisms.

At the same time, the route stretch presented in the results of Chapter 4 reveals that, independent of the reduced routing tables, the stretch is near optimal. Such behavior is essential for operating the proposed flat routing mechanism in large-scale DC networks and, besides the reduced routing tables, the results also reveal similar behavior during the process of building the routing tables, requiring reduced exchanging of signaling messages. Furthermore, the random distribution of flat IDs not only simplifies the configuration of new DCs, but also provides adequate load distribution among servers while forwarding traffic inside the DC. The adopted 3-cube topology presents elevated path redundancy, contributing for the network resilience and significantly reducing the maximum distance between servers, which is essential for the proposed *local visibility* concept.

In the vehicular network scenario, the main objective was to analyze the achieved packet delivery ratio and the path end-to-end delay while forwarding traffic. In general, the routing mechanisms available in the literature for VANETs require the presence of information regarding the entire network topology and/or the current position of all nodes present in the network, in order to perform traffic forwarding. On the other hand, the proposed XOR-based mechanism with *local visibility* builds the routing tables considering the concept of query-range, controlling the exchange of signaling messages in the network and prioritizing the insertion of physically near neighbors in the routing tables.

As shown in Chapter 5, the packet delivery ratio of the XOR_1 and XOR_2 protocols is slightly better than the results obtained with OLSR, achieving approximately 50% success in the scenarios of higher concentration of cars. Regarding the path end-to-end delay, the BGL extension used in the XOR_2 version significantly reduces the delay, presenting results similar to the AODV protocol. In essence, the main result obtained in the investigations with the VANET's scenario indicates that it is possible to successfully deliver traffic, even though the routing tables does not present information regarding the entire network. Actually, the results reveal that the presence of 100% routing information in the routing tables of protocols available in the literature does not assure 100% packet delivery, since it is a very dynamic scenario. So, the option for using reduced routing tables and reduced

signaling exchange offered by the proposed XOR-based routing mechanism with *local visibility* can be interesting for the VANET's scenario.

The main motivation of this work was extracted from the IETF RRG report [5] published in 2006, where the current scalability problems affecting the inter-domain Internet routing mechanism were described. Basically, the DFZ of the Internet adopts the path-vector solution of BGP, where all the IP prefixes are present in the routing tables composing the DFZ. In this way, due to new demands like multi-homing, the number of IP prefixes are rapidly increasing, leading to an explosive growth in the number of entries present in the routing tables. In this context, this work proposes the effective insertion of domains in the inter-domain routing mechanism, considering domains as "first class citizens" in the envisioned scenario. Consequently, instead of performing routing on top of IP prefixes as done nowadays, we perform flat routing using AS IDs.

According to the evaluations presented in Chapter 6, the proposed XOR-based mechanism not only operates using reduced routing tables and reduced signaling exchange if compared to the global knowledge approach of BGP which disseminates routing updates in the entire DFZ, but also presents route stretch near to optimal and an elevated navigability rate, whose obtained values correspond to a stretch of 1.08 with a navigability level of 99.69%.

Finally, the evaluation using the real Internet topology evidences the hierarchical routing solution used nowadays, where aggregation is essential for the overall system scalability. For historical reasons, the Internet topology is heavily tailed to a few tier 1 carriers, responsible for the vast majority of traffic forwarding. Even though the current topology presents a Power-law structure, the BGP routing mechanism does not extract benefits from such condition, requiring routing tables with the entire routing information existent. Consequently, the proposed XOR-based mechanism could represent an interesting alternative to the Internet scenario, providing mechanisms to control the growing rate of the routing tables and the required signaling to converge the routing system.

7.1 Future Work

This work focused on the development of a practical and distributed flat routing solution. Usually, proposals found in the literature present theoretical and centralized algorithms, which are not implementable in practical networks. The proposed XOR-based mechanism was instantiated in three different scenarios and, although it was presented the entire protocol specification for each investigated scenario, there are open issues left for future work.

Common open issues in the investigated scenarios are related to security and the formalization of the proposed flat routing mechanism. Regarding the security aspect, even tough the use of flat identifiers contributes for the system security, since they can be instantiated using self-certifying keys,

there are security aspects related, for example, to the exchange of signaling messages to be addressed [115]. Afterwards, the protocol formalization is strongly associated to the network structure where the protocol is instantiated. Consequently, for each investigated scenario, the formalization of upper-bound limits for the number of entries in the routing tables, the amount of signaling messages exchanged and route stretch are different, requiring individual and complex research.

In the DC scenario, future work might consider the scenario of external communications, such as the communication with the Internet and inter-3-cubes. Nowadays, large scale DCs are commonly developed using the concept of modularization, simplifying the overall DC maintenance and contributing for the reduction of operational costs, such as energy and cooling. Another future work can investigate the use of virtual machines, focusing in questions regarding the migration of virtual machines inside the DC and their impact in the XOR-based routing mechanism. Afterwards, investigations regarding the creation of multicast tress and the support for quality of service by the XOR-based mechanism inside the DCs are necessary.

In the VANET's scenario, it is possible to investigate how the adoption of Bloom filters in the communication between BGLs can impact in the overall system behavior. Furthermore, deeper investigations considering the communication between cars moving in opposite directions is necessary. Although the time interval in which cars moving in opposite directions can communicate is reduced, this scenario is useful for certain application, such as the advertisement of traffic conditions and/or accidents. Another future work can analyze how the assignment of flat identifiers to multi-media contents, such as music, videos and images can impact in the routing mechanism. As the proposed XOR-based mechanism presents better performance for higher concentration of nodes in the network, perhaps building the routing tables also considering the existent flat IDs of multi-media contents might improve its performance in the VANET's scenario.

In the inter-domain Internet scenario, a future work can focus in the development of the envisioned scenario, proposing alternatives for migrating from the current scenario towards the envisioned one. Among the open issues, it is possible to mention the specification/development of DID Proxies, the integration between the proposed Name Service and the current DNS, investigations about packet forwarding at line rate and analyzes of the impact of the existence of heterogeneous domains in the Internet. Finally, it is necessary to perform deeper analyzes on the impact of the proposed *reachability service*, and how the flattening trend of the Internet topology benefits the proposed routing mechanism.

An alternative research topic that can be developed using the proposed XOR-based routing mechanism is the content network scenario [116]. Current proposals are divided in two groups: 1) based on hierarchical naming structures [117] and 2) based on flat names [118]. We consider that the XOR-based routing mechanism can contribute in the scenario based on flat names, since this scenario

is developed on top of flat identifiers assigned to the contents available in the network. Generally, such flat identifiers are generated by hashing the title or the data of such contents, not representing the similarities that they can have regarding their content and also their concepts. So, it is also interesting to investigate the use of LSH (Locality Sensitive Hash) functions to generate such flat identifiers [119], analyzing the impact of such LSH-based IDs in the XOR-based routing mechanism. Afterwards, the content network scenario relies on the establishment of caching mechanisms, which we consider can be leveraged by the *local visibility* concept proposed in this work.

Conclusões

Este trabalho apresentou uma proposta de roteamento baseada em XOR que efetua roteamento diretamente sobre identificadores planos. Além da proximidade XOR entre esses identificadores, o mecanismo de roteamento introduz o conceito de *visibilidade local*, que prioriza a inserção de nós fisicamente próximos nas tabelas de roteamento para criar um organização de rede em malha. O cenário proposto integra totalmente o espaço de identificação plano com a estrutura física da rede, ajudando a solucionar vários desafios presentes nos cenários investigados.

A abordagem de roteamento diretamente sobre identificadores planos foi introduzida pelo IBR mas, conforme apresentado no Capítulo 2, a estrutura de anel virtual requer uma organização global para prover o encaminhamento de tráfego. Basicamente, se um determinado nó presente no anel virtual não mantiver as relações necessárias com os nós sucessores e/ou predecessores, o anel virtual é particionado, impactando no encaminhamento de tráfego como um todo. Por outro lado, o mecanismo baseado em XOR e *visibilidade local* permite que *buckets* vazios ocorram e, mesmo assim, ele é capaz de efetuar o encaminhamento de tráfego. Conforme introduzido no Capítulo 3, em um espaço de identificação de n-bits, cada *bucket* $\beta_i, 0 \leq i \leq n-1$ presente nas tabelas de roteamento baseadas em XOR representa $\frac{1}{2^{i+1}}$ nós do conjunto total de 2^n nós, amenizando o impacto dos *buckets* vazios nas tabelas de roteamento, especialmente em um cenário de larga escala, uma vez que, quão maior for o espaço de identificação usado (maior o valor de n), menor será o impacto dos *buckets* vazios localizados à direita das tabelas de roteamento (bits menos significativos).

De acordo com as avaliações apresentadas neste trabalho, é possível verificar algumas das propriedades mais importantes do mecanismo de roteamento referentes à complexidade de sinalização, *stretch*, distribuição de carga, taxa de entrega de tráfego (navegabilidade) e atraso fim-a-fim, todas observadas através do uso da ferramenta de emulação que foi desenvolvida. Entretanto, a principal característica do mecanismo apresentado está relacionada com as tabelas de roteamento. Essencialmente, as tabelas baseadas em XOR, em conjunto com o conceito de *visibilidade local*, operam usando um número reduzido de entradas, provendo a base para controlar a taxa na qual as tabelas de roteamento crescem, especialmente em um cenário de larga escala.

Conforme discutido no Capítulo 4, as redes de *data center* atuais utilizam soluções como VLANs,

tunelamento e/ou rota na origem, criando um cenário onde a rede física do *data center* é totalmente desacoplada dos servidores presentes no interior do *data center*. De maneira simplista, é impossível inserir informação referente à totalidade dos servidores nas tabelas de encaminhamento dos *switches*, uma vez que a memória existente nesses equipamentos não suporta o conjunto de informações disponíveis. Sendo assim, os resultados obtidos no cenário de *data center* apresentam uma importante contribuição, suportando a integração entre servidores e a estrutura física do *data center*, uma vez que as tabelas de roteamento não acompanham linearmente a informação de roteamento disponível na rede, como ocorre nos mecanismos de estado do enlace.

Ao mesmo tempo, o *stretch* apresentado nos resultados do Capítulo 4 revela que, independente das tabelas de roteamento reduzidas, o *stretch* é próximo do valor ótimo. Tal comportamento é essencial para utilizar o mecanismo de roteamento plano em redes de *data center* de larga escala e, além das tabelas reduzidas, os resultados também revelam um comportamento similar durante o processo de construção das tabelas de roteamento, utilizando uma reduzida troca de mensagens de sinalização. A distribuição aleatória de identificadores planos simplifica a configuração do *data center* e provê uma adequada distribuição de carga entre os servidores durante o encaminhamento de tráfego no *data center*. A topologia em cubo 3-dimensional apresenta uma elevada redundância de caminhos, contribuindo para a resiliência da rede e para a redução da distância máxima entre os servidores, características essenciais para a utilização do conceito de *visibilidade local* proposto.

No cenário de redes veiculares, o objetivo principal era analisar a taxa de entrega de pacotes obtida e o atraso fim-a-fim durante o encaminhamento de tráfego. No geral, os mecanismos de roteamento disponíveis na literatura para VANETs exigem a presença de informação referente à topologia da rede como um todo e/ou a posição atual de todos os nós presentes na rede, para efetuar o encaminhamento de tráfego. Por outro lado, o mecanismo baseado em XOR com *visibilidade local* constrói suas tabelas de roteamento considerando o conceito de *query-range*, controlando a troca de mensagens de sinalização na rede e priorizando a inserção de vizinhos fisicamente próximos nas tabelas de roteamento.

Conforme apresentado no Capítulo 5, a taxa de entrega de pacotes dos protocolos XOR_1 e XOR_2 são um pouco melhores que os resultados obtidos com o protocolo OLSR, atingindo aproximadamente 50% de sucesso nos cenários com uma maior concentração de carros. Referente ao atraso fim-a-fim dos caminhos, a extensão que utiliza os nós BGLs na versão XOR_2 significativamente reduz esta métrica, apresentando resultados similares ao protocolo AODV. Em essência, o principal resultado obtido nas investigações feitas no cenário das VANETs indica que é possível entregar tráfego de forma eficaz, embora as tabelas de roteamento não apresentem informação referente à rede como um todo. Na verdade, os resultados revelam que a presença de 100% da informação de roteamento nas tabelas dos protocolos disponíveis na literatura não garante 100% de entrega de

pacotes, uma vez que as redes veiculares são muito dinâmicas. Desta forma, a opção de usar tabelas de roteamento reduzidas e um número reduzido de mensagens de sinalização, oferecida pelo mecanismo baseado em XOR com *visibilidade local*, pode ser interessante para o cenário das VANETs.

A motivação principal deste trabalho foi extraída do relatório publicado em 2006 pelo RRG do IETF [5], no qual os problemas atuais de escalabilidade que afetam o mecanismo de roteamento entre domínios da Internet foram descritos. Basicamente, a DFZ da Internet adota a solução de vetor de caminhos do BGP, na qual todos os prefixos IP estão presentes nas tabelas de roteamento que compõem a DFZ. Desta forma, devido a novas demandas como o *multi-homing*, o número de prefixos IP tem aumentado rapidamente, levando a um crescimento explosivo no número de entradas presentes nas tabelas de roteamento. Neste contexto, este trabalho propõe a inserção efetiva dos domínios no mecanismo de roteamento entre domínios, considerando-os "cidadãos de primeira classe" no cenário previsto. Consequentemente, ao invés de efetuar roteamento usando os prefixos IP conforme o mecanismo atual, optou-se pelo roteamento plano baseado nos identificadores de ASs.

De acordo com as avaliações apresentadas no Capítulo 6, o mecanismo proposto opera utilizando tabelas de roteamento reduzidas e efetua uma baixa troca de sinalização, especialmente se comparado com a abordagem de conhecimento global do BGP que efetua a disseminação de atualizações através de toda a DFZ. Além disso, o mecanismo proposto apresenta valores de *stretch* próximos do ótimo e uma alta taxa de navegabilidade, cujos valores correspondem a um *stretch* de 1,08 e uma taxa de navegabilidade de 99,69%, ambos valores obtidos utilizando puramente a solução baseada em XOR.

Finalmente, as avaliações usando a topologia real da Internet evidenciam a solução de roteamento hierárquica utilizada atualmente, na qual a agregação é essencial para a escalabilidade do sistema como um todo. Por motivos históricos, a topologia da Internet é fortemente dependente de um pequeno grupo de redes portadoras (*tier 1*), responsáveis pela grande maioria do encaminhamento de tráfego. Porém, embora a topologia atual da Internet apresente a estrutura *Power-law*, o mecanismo de roteamento do BGP não extrai benefícios desta condição, criando tabelas de roteamento com toda a informação de roteamento existente. Consequentemente, o mecanismo baseado em XOR proposto pode representar uma alternativa interessante para o cenário Internet, provendo meios para controlar a taxa de crescimento das tabelas de roteamento e a sinalização necessária para convergir o sistema de roteamento.

Trabalhos Futuros

Este trabalho focou no desenvolvimento de uma solução de roteamento prática e distribuída. Geralmente, as propostas disponíveis na literatura apresentam soluções teóricas e centralizadas, as quais não podem ser implementadas em redes reais. O mecanismo de roteamento proposto foi

instanciado em três cenários diferentes e, embora este trabalho tenha apresentado a especificação completa do protocolo para cada um dos cenários investigados, ainda existem questões em aberto que poderão ser abordadas em trabalhos futuros.

Dentre as questões em aberto comuns aos cenários investigados, temos questões relacionadas com a segurança e a formalização do mecanismo de roteamento plano proposto. Referente ao aspecto de segurança, embora o uso de identificadores planos contribua para a segurança do sistema, uma vez que eles podem ser gerados a partir de chaves auto-certificadoras, ainda existem questões, por exemplo, relacionadas à troca de mensagens de sinalização a serem analisadas [115]. Além disso, a formalização do protocolo é fortemente associada à estrutura de rede na qual o protocolo será instanciado. Consequentemente, em cada cenário a formalização de limites máximos referentes ao número de entradas nas tabelas de roteamento, a quantidade de mensagens de sinalização trocadas e o *stretch* são diferentes, exigindo distintas e complexas pesquisas.

No cenário de redes de *data center*, trabalhos futuros poderiam considerar a comunicação externa, tal como a comunicação com a Internet e entre cubos. Atualmente, *data centers* de larga escala são comumente desenvolvidos usando o conceito de modularização, simplificando o manutenção geral do *data center* e contribuindo para a redução de custos operacionais, tais como energia e resfriamento. Um outro trabalho futuro poderia investigar o uso de máquinas virtuais, focando em questões relacionadas à migração de máquinas virtuais dentro do *data center* e o impacto no mecanismo de roteamento baseado em XOR. Além disso, investigações relacionadas ao estabelecimento de árvores de *multicast* e ao suporte à qualidade de serviço poderiam ser efetuadas.

No cenário de redes veiculares, seria interessante investigar o uso de filtros de Bloom para prover a comunicação entre nós BGL, estudando qual o impacto no comportamento geral do sistema. Ao mesmo tempo, investigações mais detalhadas considerando a comunicação entre carros movendo-se em direções opostas deveriam ser abordadas. Embora o intervalo de tempo no qual carros movendo-se em direções opostas podem se comunicar seja reduzido, este cenário é importante para algumas aplicações, tal como a divulgação sobre a condição de tráfego e/ou acidentes. Um outro trabalho futuro poderia pesquisar como a atribuição de identificadores planos para conteúdos multimídia, tais como músicas, vídeos e imagens pode impactar no mecanismo de roteamento. Uma vez que o mecanismo proposto apresenta um melhor desempenho em cenários onde há uma maior concentração de nós, talvez construir as tabelas de roteamento considerando os identificadores planos de conteúdos multimídia possa melhorar o desempenho do protocolo nas redes veiculares.

No cenário entre domínios da Internet, um trabalho futuro poderia focar no desenvolvimento do cenário previsto, propondo alternativas de migração a partir do cenário atual. Dentre as questões em aberto, é possível mencionar a especificação/desenvolvimento dos *DID Proxies*, a integração entre o Serviço de Nomes proposto e o DNS atual, investigações sobre o encaminhamento de pacotes

em velocidade de linha e análises sobre o impacto da existência de domínios heterogêneos na Internet. Por último, seria interessante efetuar análises mais detalhadas sobre o impacto do serviço de alcançabilidade proposto e como a tendência da topologia da Internet tornar-se mais plana poderia beneficiar o mecanismo de roteamento proposto.

Um tópico de pesquisa alternativo poderia ser desenvolvido usando o mecanismo de roteamento baseado em XOR no cenário de redes de conteúdo [116]. Propostas atuais estão divididas em dois grupos: 1) baseadas em estruturas de nomes hierárquicos [117] e 2) baseadas em nomes planos [118]. Consideramos que o mecanismo de roteamento baseado em XOR poderia contribuir para o cenário baseado em nomes planos, uma vez que esse cenário é desenvolvido sobre identificadores planos atribuídos aos conteúdos disponíveis na rede. Tais identificadores são geralmente criados através do *hash* do nome ou dos dados desses conteúdos, não representando as similaridades que eles podem ter em relação ao seu conteúdo e, também, a sua classificação dentro de um domínio de conhecimento. Sendo assim, seria interessante investigar o uso de funções de LSH (*Locality Sensitive Hash*) para gerar tais identificadores planos [119], analisando o impacto desses identificadores no mecanismo de roteamento baseado em XOR. Além disso, o cenário de redes de conteúdo baseia-se no estabelecimento de mecanismos de *cache*, os quais nós consideramos que podem ser alavancados pelo conceito de *visibilidade local* proposto neste trabalho.

Referências Bibliográficas

[1] BGP Routing Table Analysis Reports. Available at `http://bgp.potaroo.net/`, 2011.

[2] J. Moy. OSPF Version 2. *IETF RFC-1131*, July 1991.

[3] G. Malkin. RIP Version 2 - Carrying Additional Information. *IETF RFC-1723*, November 1994.

[4] Y. Rekhter and T. Li. A Border Gateway Protocol 4 (BGP-4). *IETF RFC-1771*, March 1995.

[5] D. Meyer, L. Zhang, and K. Fall. Report from the IAB workshop on routing and addressing. *IETF RFC-4984*, September 2007.

[6] C. Semeria. Understanding IP Addressing: Everything You Ever Wanted To Know. *3Com Corporation - Available at `http://www.tcpip-lab.net/links/ip_subnet.html`*, April 1996.

[7] V. Cerf and R. Kahan. A Protocol for Packet Network Intercommunication. *IEEE Transactions on Communications*, 22(5):637–648, May 1974.

[8] L. Kleinrock. Information Flow in Large Communication Nets. *RLE Quarterly Progress Report, Massachusetts Institute of Technology*, July 1961.

[9] P. Mähönen, D. Trossen, D. Papadimitriou, George Polyzos, and David Kennedy. The Future Networked Society. *A white paper from the EIFFEL Think-Tank*, December 2006.

[10] V. Fuller, T. Li, J. Yu, and K. Varadhan. Classless Inter-Domain Routing (CIDR): an Address Assignment and Aggregation Strategy. *IETF RFC-1519*, September 1993.

[11] S. Deering and R. Hinden. Internet Protocol, Version 6 (IPv6) Specification. *IETF RFC-2460*, December 1998.

[12] P. Srisuresh and M. Holdrege. IP Network Address Translator (NAT) Terminology and Considerations. *IETF RFC-2663*, August 1999.

[13] Ion Stoica, Robert Morris, David Karger, M. Frans Kaashoek, and Hari Balakrishnan. Chord: A Scalable Peer-to-peer Lookup Service for Internet Applications. *In Proceedings of the ACM SIGCOMM 2001 - San Diego, CA, EUA, August 27-31, 2001*.

[14] Antony Rowstron and Peter Druschel. Pastry: Scalable, decentralized object location and routing for large-scale peer-to-peer systems. *In Proceedings of the 28th IFIP/ACM Internnational Conference on Distributed Systems Platforms (Middleware 2001), Heidelberg, Germany*, November 2001.

[15] Ben Y. Zhao, Ling Huang, Jeremy Stribling, Sean C. Rhea, Anthony D. Joseph, and John D. Kubiatowicz. Tapestry: A Resilient Global-Scale Overlay for Service Deployment. *IEEE Journal on Selected Areas in Communications, volume 22, number 1*, January 2004.

[16] P. Maymounkov and D. Mazières. Kademlia: A Peer-to-peer Information System Based on the XOR Metric. *In 1st International Workshop on Peer-to-Peer Systems (IPTPS) - MIT Faculty Club, Cambridge, MA, USA*, March 7-8 2002.

[17] Bryan Ford. UIA: A Global Connectivity Architecture for Mobile Personal Devices. *Ph.D. Thesis. Massachusetts Institute of Technology (MIT), Department of Electrical Engineering and Computer Science*, September 2008.

[18] B. Ford. Scalable Internet Routing on Topology-Independent Node Identities. *MIT technical report conducted as part of the IRIS project (http://project-iris.net)*, October 2003.

[19] J. Abley, K. Lindqvist, E. Davies, B. Black, and V. Gill. IPv4 Multihoming Practices and Limitations. *IETF RFC-4116*, July 2005.

[20] Xiaomei Liu and Li Xiao. A Survey of Multihoming Technology in Stub Networks: Current Research and Open Issues. *Network, IEEE, pages: 32 - 40*, May/June 2007.

[21] A. Doria, E. Davies, and F. Kastenholz. A Set of Possible Requirements for a Future Routing Architecture. *IETF RFC-5772*, February 2010.

[22] R. Moskowitz and P. Nikander. Host Identity Protocol (HIP) Architecture. *IETF RFC-4423*, May 2006.

[23] Bengt Ahlgren, Jari Arkko, Lars Eggert, and Jarno Rajahalme. A Node Identity Internetworking Architecture. *In Proceedings of the 9th IEEE Global Internet Symposium realized in conjunction with IEEE INFOCOM, Barcelona, Spain, April 28-29, 2006*.

[24] D. Farinacci, V. Fuller, D. Oran, and D. Meyer. Locator/ID Separation Protocol (LISP). *IETF Draft - draft-farinacci-lisp-12.txt*, September 2009.

[25] S. Brim, N. Chiappa, D. Farinacci, V. Fuller, D. Lewis, and D. Meyer. LISP-CONS: A Content distribution Overlay Network Service for LISP. *IETF Draft - draft-meyer-lisp-cons-04.txt*, April, 9 2008.

[26] D. Jen, M. Meisel, D. Massey, L. Wang, B. Zhang, and L. Zhang. APT: A Pratical Transit Mapping Service. *IETF Draft - draft-jen-apt-01.txt*, November, 18 2007.

[27] E. Lear. NERD: A Not-so-novel EID to RLOC Database. *IETF Draft - draft-lear-lisp-nerd-08.txt*, March, 6 2010.

[28] Luigi Iannone and Olivier Bonaventure. On the Cost of Caching Locator/ID Mappings. *In Proceedings of the CoNEXT 2007 - New York, USA*, December 10 - 13 2007.

[29] Matthew Chapman Caesar. Identity-Based Routing. *Ph.D. Thesis - University of California, Berkeley, USA*, September 3 2007.

[30] M. Caesar, M. Castro, E. B. Nightingale, G. O'Shea, and A. Rowstron. Virtual Ring Routing: Network Routing Inspired by DHTs. *In Proceedings of the ACM SIGCOMM - Pisa, Italy*, pages 351 – 362, September 11-15 2006.

[31] M. Caesar, K. Lakshminarayanan, T. Condie, I. Stoica, J. Kannan, and S. Shenker. ROFL: Routing on Flat Labels. *In Proceedings of the ACM SIGCOMM - Pisa, Italy*, pages 363 – 374, September 11-15 2006.

[32] Marta Arias, Lenore J. Cowen, Kofi A. Laing, Rajmohan Rajaraman, and Orjeta Taka. Compact Routing With Name Independence. *In SPAA'03, San Diego, California, USA*, June 7 - 9 2003.

[33] M. Thorup and U. Zwick. Compact Routing Schemes. *In SPAA'01, Crete Island, Greece*, July 4 - 6 2001.

[34] Paolo Costa, Thomas Zahn, Ant Rowstron, Greg O'Shea, and Simon Schubert. Why Should we Integrate Services, Servers, and Networking in a Data Center? In *Proceedings of the 1st ACM Workshop on Research on Enterprise Networking (WREN'09)*, pages 111–118, New York, NY, USA, 2009.

[35] Chuanxiong Guo, Guohan Lu, Dan Li, Haitao Wu, Xuan Zhang, Yunfeng Shi, Chen Tian, Yongguang Zhang, and Songwu Lu. BCube: A High Performance, Server-centric Network Architecture for Modular Data Centers. *In Proceedings of the ACM SIGCOMM 2009 - Conference on Data Communication, Barcelona, Spain*, August 17 - 21 2009.

[36] Haitao Wu, Guohan Lu, Dan Li, Chuanxiong Guo, and Yongguang Zhang. MDCube: A High Performance Network Structure for Modular Data Center Interconnection. *In Proceedings of the ACM CONEXT 2009, Rome, Italy*, December 1-4 2009.

[37] Elizabeth M. Royer and Chai-Keong Toh. A Review of Current Routing Protocols for ad hoc Mobile Wireless Networks. *IEEE Personal Communications, Volume: 6 Issue: 2, pages: 46 - 55*, April 1999.

[38] J. Chennikara-Varghese, W. Chen, O. Altintas, and S. Cai. Survey of routing protocols for inter-vehicle communications. In *Mobile and Ubiquitous Systems: Networking and Services, 2006 Third Annual International Conference on*, pages 1–5, July 2006.

[39] M. Mauve, A. Widmer, and H. Hartenstein. A survey on position-based routing in mobile ad hoc networks. *Network, IEEE*, 15(6):30–39, Nov/Dec 2001.

[40] Rodolfo Oliveira, Luís Bernardo, and Paulo Pinto. Searching for resources in manets: A cluster based flooding approach. *ICETE*, pages 105–111, 2005.

[41] Rodolfo Oliveira. Controlo de Acesso ao Meio em Redes Ad Hoc Móveis IEEE 802.11. *Ph.D. Thesis - Universidade Nova de Lisboa*, 2009.

[42] Rodolfo Oliveira, Luis Bernardo, Miguel Luis, and Paulo Pinto. Improving Routing Performance in High Mobility and High Density ad hoc Vehicular Networks. *IEEE Sarnoff Symposium, Princeton, NJ*, April 12 - 14 2010.

[43] D. Krioukov, k c claffy, K. Fall, and A. Brady. On Compact Routing for the Internet. *In Proceedings of the ACM SIGCOMM Computer Communication Review*, 37(3):41–52, July 2007.

[44] D. Krioukov, K. Fall, and X. Yang. Compact Routing on Internet-Like Graphs. *In Proceedings of the IEEE INFOCOM 2004*, March 2004.

[45] Shudong Jin and Azer Bestavros. Small-world characteristics of Internet topologies and implications on multicast scaling. *Computer Networks, Volume 50, Issue 5, Responsible Editor: Jon Crowcroft*, pages 648 – 666, April 2006.

[46] M. Boguñá, D. Krioukov, and kc claffy. Navigability of complex networks. *Nature Physics, v.5, p.74-80, 2009*.

[47] M. Á. Serrano, D. Krioukov, and M. Boguñá. Self-similarity of complex networks and hidden metric spaces. *Physical Review Letters, vol. 100, no. 078701, in February 2008.*

[48] P. F. Tsuchiya. The Landmark Hierarchy: A New Hierarchy for Routing in Very Large Networks. *In Proceedings of the ACM SIGCOMM 88. August 1988.*

[49] The Cooperative Association for Internet Data Analysis, `http://www.caida.org`, 2011.

[50] L. Kleinrock and F. Kamoun. Hierarchical routing for large networks: Performance evaluation and optimization. *Computer Networks, 1:155-174*, 1977.

[51] L. Cowen. Compact Routing With Minimum Stretch. *Journal of Algorithms, 38(1):170-183*, 2001.

[52] B. Awerbuch, A. Bar-Noy, N. Linial, and D. Peleg. Compact distributed data structures for adaptive network routing. *In Proceedings of the 21st ACM Symposium on Theory of Computing, pages 479 - 489*, May 1989.

[53] Ittai Abraham, Cyril Gavoille, Dahlia Malkhi, Noam Nisan, and Mikkel Thorup. Compact Name-Independent Routing with Minimum Stretch. *In SPAA'04, Barcelona, Spain*, June 27 - 30 2004.

[54] Sourabh Jain, Yingying Chen, Zhi-Li Zhang, and Saurabh Jain. VIRO: A Scalable, Robust and Namespace Independent Virtual Id ROuting for Future Networks. *In 30th IEEE International Conference on Computer Communications (IEEE INFOCOM), Shanghai, China*, April 10-15 2011.

[55] Ankit Singla, P. Brighten Godfrey, Kevin Fall, Gianluca Iannaccone, and Sylvia Ratnasamy. Scalable Routing on Flat Names. *In Proceedings of the ACM CoNEXT, Philadelphia, USA*, November 30 - December 3 2010.

[56] Nancy A. Lynch. Distributed algorithms. *Morgan Kaufmann Publishers Inc. San Francisco, CA, USA - ISBN:1558603484*, 1996.

[57] T. Clausen and P. Jacquet. Optimized Link State Routing Protocol (OLSR). *IETF RFC-3626*, October 2003.

[58] Alec Woo, Terence Tong, and David Culler. Taming the Underlying Challenges of Reliable Multihop Routing in Sensor Networks. *In SenSys*, November 2003.

[59] C. Perkins and P. Bhagwat. Highly Dynamic Destination-Sequenced Distance-Vector Routing (DSDV) for Mobile Computers. In *Proceedings of the ACM SIGCOMM'94 Conference on Communications Architectures, Protocols and Applications*, pages 234–244, 1994.

[60] Prasanna Ganesan, Krishna Gummadi, and Hector Garcia-Molina. Canon in G Major: Designing DHTs with Hierarchical Structure. *In 24th International Conference on Distributed Computing Systems (ICDCS 2004), Tokyo, Japan*, March 23-26 2004.

[61] K. Gummadi, R. Gummadi, S. Gribble, S. Ratnasamy, S. Shenker, and I. Stoica. The Impact of DHT Routing Geometry on Resilience and Proximity. *In Proceedings of the ACM SIGCOMM'03, Karlsruhe, Germany*, August 25-29 2003.

[62] A. Broder and M. Mitzenmacher. Network Applications of Bloom Filters: A Survey. *Internet Mathematics. 2002.*

[63] F. Chang, J. Dean, S. Ghemawat, W. C. Hsieh, D. A. Wallach, M. Burrows, T. Chandra, A. Fikes, and R. E. Gruber. BigTable: A Distributed Storage System for Structured Data. *In Proceedings of the OSDI 2006, Seattle, WA, USA*, November 06 - 08 2006.

[64] S. Ghemawat, H. Gobioff, and S. T. Leung. The Google File System. *In Proceedings of the SOSP 2003, Bolton Landing (Lake George), New York, USA*, October 19 - 22 2003.

[65] C. Guo, H. Wu, K. Tan, L. Shiy, Y. Zhang, and S. Luz. MapReduce: Simplified Data Processing on Large Clusters. *In Proceedings of the OSDI 2004, San Francisco, CA, USA*, December 06 - 08 2004.

[66] Cisco. Cisco Unified Computing System. Available at `http://www.cisco.com/go/unifiedcomputing`, 2011.

[67] Google. Google's Custem Web Server, Revealed. Available at `http://tinyurl.com/google-custom-server`, 2011.

[68] F. L. Verdi, C. Esteve, R. Pasquini, and M. F. Magalhães. Novas Arquiteturas de Data Center para Cloud Computing. *Book Chapter published as a Mini Course of the 28th Brazilian Symposium on Computer Networks and Distributed Systems (SBRC 2010), Organized by: C.*

A. Kamienski; L. P. Gaspary; M. P. Barcellos. ISBN: 9772177497006. Idiom: Portuguese. Gramado, RS, Brazil., May 24 - 28 2010.

[69] Albert Greenberg, Parantap Lahiri, David A. Maltz, Parveen Patel, and Sudipta Sengupta. Towards a Next Generation Data Center Architecture: Scalability and Commoditization. *In Proceedings of the ACM Workshop on Programmable Routers For Extensible Services of Tomorrow, Seattle, WA, USA*, August 22 2008.

[70] Albert Greenberg, James R. Hamilton, Navendu Jain, Srikanth Kandula, Changhoon Kim, Parantap Lahiri, David A. Maltz, Parveen Patel, and Sudipta Sengupta. VL2: A Scalable and Flexible Data Center Network. *In Proceedings of the ACM SIGCOMM 2009 - Conference on Data Communication, Barcelona, Spain*, August 17 - 21 2009.

[71] Killer NIC. Killer NIC. Available at `http://www.killernic.com`, 2011.

[72] R. Pasquini, F. L. Verdi, and M. F. Magalhães. Integrating Servers and Networking using an XOR-based Flat Routing Mechanism in 3-cube Server-centric Data Centers. *29th Brazilian Symposium on Computer Networks and Distributed Systems (SBRC 2011), Campo Grande, MS, Brazil*, May 30 - June 03 2011.

[73] J. Dean and S. Ghemawat. DCell: A Scalable and Fault-Tolerant Network Structure for Data Centers. *In Proceedings of the ACM SIGCOMM 2008, Seattle, WA, USA*, August 17 - 22 2008.

[74] Radhika Niranjan Mysore, Andreas Pamboris, Nathan Farrington, Nelson Huang, Pardis Miri, Sivasankar Radhakrishnan, Vikram Subramanya, and Amin Vahdat. PortLand: A Scalable Fault-Tolerant Layer 2 Data Center Network Fabric. *In Proceedings of the ACM SIGCOMM 2009 - Conference on Data Communication, Barcelona, Spain*, August 17 - 21 2009.

[75] Amazon Elastic Computing Cloud, `http://aws.amazon.com/ec2/`, 2011.

[76] eSafety. Available at `http://ec.europa.eu/information_society/activities/esafety/index_en.htm`, 2011.

[77] Intellidrive. Available at `http://www.its.dot.gov/`, 2011.

[78] simTD. Available at `http://www.sit.fraunhofer.de/en/forschungsbereiche/projekte/simTD.jsp`, 2011.

[79] SCORE@F. Available at `http://imara.inria.fr/projects/scoref`, 2011.

[80] J.J. Blum, A. Eskandarian, and L.J. Hoffman. Challenges of intervehicle ad hoc networks. *IEEE Transactions on Intelligent Transportation Systems*, 5(4):347–351, Dec. 2004.

[81] C.E. Perkins and E.M. Royer. Ad-hoc on-demand distance vector routing. *In Proceedings of the 2nd IEEE Workshop on Mobile Computing Systems and Applications WMCSA '99*, pages 90–100, February 25-26 1999.

[82] D. B. Johnson and D. A. Maltz. Dynamic Source Routing in Ad Hoc Wireless Networks. Mobile Computing, volume 353, editors: Tomasz Imielinski and Hank Korth, Kluwer Academic Publishers, 1996.

[83] V.D. Park and M.S. Corson. A highly adaptive distributed routing algorithm for mobile wireless networks. In *Proceedings of the INFOCOM '97. Sixteenth Annual Joint Conference of the IEEE Computer and Communications Societies*, volume 3, pages 1405–1413 vol.3, Apr 1997.

[84] G. P. Mario, M. Gerla, and T. Chen. Fisheye state routing: A routing scheme for ad hoc wireless networks. In *Proceedings of the ICC 2000*, pages 70–74, 2000.

[85] Rodolfo Oliveira, Luis Bernardo, and Paulo Pinto. The Influence of Broadcast Traffic on IEEE 802.11 DCF Networks. *Elsevier Computer Communications*, 32(2):439–452, 2009.

[86] Sze-Yao Ni, Yu-Chee Tseng, Yuh-Shyan Chen, and Jang-Ping Sheu. The broadcast storm problem in a mobile ad hoc network. In *MobiCom '99: Proceedings of the 5th annual ACM/IEEE international conference on Mobile computing and networking*, pages 151–162, New York, NY, USA, 1999.

[87] B. K. and H. T. Kung. GPSR: Greedy Perimeter Stateless Routing for Wireless Networks. In *MobiCom '00: Proceedings of the 6th annual international conference on Mobile computing and networking*, pages 243–254, 2000.

[88] C. Lochert, H. Hartenstein, J. Tian, H. Fussler, D. Hermann, and M. Mauve. A routing strategy for vehicular ad hoc networks in city environments. *IEEE Intelligent Vehicles Symposium*, 2000:156–161, 2003.

[89] B. C. Seet, Genping Liu, B. S. Lee, Chuang heng Foh, K. J. Wong, and K. K. Lee. A-STAR: A mobile ad hoc routing strategy for metropolis vehicular communications. *Networking Technologies, Services, and Protocols; Performance of Computer and Communication Networks; Mobile and Wireless Communications (NETWORKING 2004)*, pages 989–999, 2004.

[90] V. Naumov and TR. Gross. Connectivity-Aware Routing (CAR) in vehicular ad-hoc networks. *26th IEEE International Conference on Computer Communications (INFOCOM 2007)*, pages 1919–1927, 2007.

[91] R. Oliveira, A. Garrido, M. Luis, R. Pasquini, L. Bernardo, R. Dinis, and P. Pinto. Performance Analysis of XOR-Based Routing Protocols in Vehicular ad hoc Networks. *Book Chapter published in Internet Policies and Issues, v. 8, Nova Publishers, Organized by: B. G. Kutais. ISBN: 978-1-61122-840-3. Idiom: English. New York, USA.*, 2011.

[92] R. Oliveira, A. Garrido, R. Pasquini, M. Luis, L. Bernardo, R. Dinis, and P. Pinto. Towards the use of XOR-based Routing Protocols in Vehicular ad hoc Networks. *73rd IEEE Vehicular Technology Conference - VTC 2011, Budapest, Hungary.*, May 15 - 18 2011.

[93] M. Luis, R. Oliveira, L. Bernardo, A. Garrido, and P. Pinto. Joint topology control and routing in ad hoc vehicular networks. *European Wireless Conference (EW)*, pages 528–535, 2010.

[94] TraNS. open source tool for realistic simulations of VANET applications. Available at `http://trans.epfl.ch/`, 2011.

[95] Centre for Applied Informatics (ZAIK) and Institute of Transport Research at the German Aerospace Centre. SUMO - Simulation of Urban Mobility. Available at `http://sumo.sourceforge.net`, 2011.

[96] Information Sciences Institute. NS-2 network simulator (version 2.31). Available at `http://www.isi.edu/nsnam/ns/`, 2007.

[97] ANSI/IEEE 802.11 Standard. Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications, 1999.

[98] A. Qayyum, L. Viennot, and A. Laouiti. Multipoint relaying for flooding broadcast messages in mobile wireless networks. *Hawaii International Conference on System Sciences*, 9:298, 2002.

[99] R. Pasquini, F. L. Verdi, M. F. Magalhães, and Annikki Welin. Bloom filters in a Landmark-based Flat Routing. *In Proceedings of the IEEE ICC 2010, Cape Town, South Africa*, May 23 - 27 2010.

[100] R. Pasquini, F. L. Verdi, Rodolfo Oliveira, M. F. Magalhães, and Annikki Welin. A Proposal for an XOR-based Flat Routing Mechanism in Internet-like Topologies. *In Proceedings of the IEEE GLOBECOM 2010, Miami, FL, USA*, December 6 - 10 2010.

[101] André Gabriel Garrido. Encaminhamento plano em redes ad-hoc veiculares. *Master Dissertation - Universidade Nova de Lisboa*, 2010.

[102] Walter Wong, Rafael Pasquini, Rodolfo Villaça, Luciano B. Paula, Fábio Verdi, and Maurício Magalhães. A Framework for Mobility and Flat Addressing in Heterogeneous Domains. *25th Brazilian Symposium on Computer Networks and Distributed Systems - SBRC 2007. Belém - PA, Brazil. May 2007*.

[103] W. Wong, R. Villaça, L. B. Paula, R. Pasquini, F. L. Verdi, and M. Magalhães. An Architecture for Mobility Support in a Next Generation Internet. *The 22nd IEEE International Conference on Advanced Information, Networking and Applications (AINA 2008)*, March 2008.

[104] R. Pasquini, L. B. de Paula, F. L. Verdi, and M. F. Magalhães. Domain Identifiers in a Next Generation Internet Architecture. *IEEE Wireless Communications & Networking Conference (WCNC) - Budapest, Hungary*, April 5-8 2009.

[105] R. Pasquini, R. Oliveira, F. L. Verdi, M. F. Magalhães, and A. Welin. Towards Local Routing State in the Internet. *Submitted to IEEE Communications Letters*, 2011.

[106] Q. Vohra and E. Chen. BGP Support for Four-octet AS Number Space. *IETF RFC-4893*, May 2007.

[107] R. Pasquini, F. L. Verdi, and M. F. Magalhães. Towards a Landmark-based Flat Routing. *27th Brazilian Symposium on Computer Networks and Distributed Systems (SBRC 2009), Recife, PE, Brazil*, May 25 - 29 2009.

[108] Boston university Representative Internet Topology gEnerator (BRITE). Available at `http://www.cs.bu.edu/brite`, 2011.

[109] Xiaoliang Zhao, Dante J. Pacella, and Jason Schiller. Routing Scalability: An Operator's View. *In IEEE Journal on Selected Areas in Communications, Volume 28, Number 8*, pages 1262–1270, October 2010.

[110] Jeffrey Travers and Stanley Milgram. An Experimental Study of the Small World Problem. *Sociometry, Volume 32, Number 4*, pages 425–443, 1969.

[111] Amogh Dhamdhere and Constantine Dovrolis. The Internet is Flat: Modeling the Transition from a Transit Hierarchy to a Peering Mesh. *In Proceedings of ACM CoNEXT 2010, Philadelphia, USA*, November 30 - December 3 2010.

[112] Craig Labovitz, Scott Iekel-Johnson, Danny McPherson, Jon Oberheide, and Farnam Jahanian. Internet Inter-Domain Traffic. *In Proceedings of ACM SIGCOMM 2010, New Delhi, India*, August 30 - September 3 2010.

[113] Phillipa Gill, Martin Arlitt, Zongpeng Li, and Anirban Mahanti. The Flattening Internet Topology: Natural Evolution, Unsightly Barnacles or Contrived Collapse? *In Proceedings of of PAM'2008, Cleveland, Ohio, USA*, pages 1–10, April 29 - 30 2008.

[114] N. Feamster, L. Gao, and J. Rexford. How to Lease the Internet in Your Spare Time. *In the Editorial Zone of ACM SIGCOMM Computer Communications Review, p. 61-64*, January 2007.

[115] Walter Wong. Um plano de segurança para autenticação de dados em redes orientadas à informação. *Ph.D. Thesis. University of Campinas (DCA/FEEC/UNICAMP)*, 2011.

[116] Christian Esteve Rothenberg. Compact forwarding: A probabilistic approach to packet forwarding in content-oriented networks. *Ph.D. Thesis. University of Campinas (DCA/FEEC/UNICAMP)*, December 15 2010.

[117] V. Jacobson, D. Smetters, J. Thornton, M. Plass, N. Briggs, and R. Braynard. Networking Named Content. *In Conext 2009, Rome, Italy*, December 2009.

[118] D. Trossen, M. Särelä, and K. Sollins. Arguments for an Information-Centric Internetworking Architecture. *In ACM Computer Communication Review (CCR)*, 2010.

[119] Luciano Bernardes de Paula. Utilização de funções LSH para busca conceitual baseada em ontologia. *Ph.D. Thesis. University of Campinas (DCA/FEEC/UNICAMP)*, July 2011.

[120] Otter Network Visualization Tool (OTTER). Available at `http://www.caida.org/tools/visualization/otter`, 2011.

Appendix A

Developed Emulation Tool

This appendix details the architecture of the developed emulation tool used in this work. Besides the function of gathering information about the operation of the proposed flat routing mechanism, the tool massively contributed to the development of the protocol. Essentially, the tool offers an environment in which the protocol can be emulated considering any type of network topology, helping to identify problems in the protocol specification and leading to the evolution of the entire routing mechanism.

It is a challenging task to analyze the operation of new protocols. The main difficult is to prepare a testbed network, since the envisioned scenario of new protocols may consider enormous networks, such as the Internet. In this way, common evaluations found in the literature rely on simulation tools, where scripts are used to conduct the analyzes of new protocols. Such scenario ossifies the interactions between all nodes composing the network, removing the spontaneous characteristic present in real networks.

In this context, the developed tool offers an emulation environment aimed at analyzing the proposed XOR-based flat routing mechanism. In the developed tool, a real implementation of the protocol is embedded in threads emulating individual nodes present in the network topology. The main property of such tool is related to the fact that none sequential script is required to conduct the analyzes of the protocol, all the emulated nodes are instantiated as if they were real nodes in the network. Consequently, emulated nodes exchange the required signaling messages to build their own routing tables according to the protocol specification, recovering the spontaneity of real networks, and also allowing traffic exchange between the emulated network.

Basically, the tool is divided in two main functionalities and is entirely implemented using the JAVA language. The first functionality is responsible for the network emulation, instantiating nodes (threads) and links (TCP connections) composing the network graph in a distributed environment. In this way, big topologies can be distributed in a set of computers, offering a scenario where resources

such as memory and processing can be grouped to scale the evaluations. The second functionality provides a control graphical interface developed to manage the evaluations of the proposed flat routing mechanism. Using the interface it is possible to observe the execution of the emulated nodes and to dispatch commands to them.

The remainder of this appendix is organized as follows. Section A.1 presents the tool specification, describing the architectural details of the emulated nodes and the management interface. Section A.2 briefly presents complementary modules of the tool. Section A.3 explains how to operate the tool in the distributed environment.

A.1 Tool Specification

As mentioned before, the tool is organized in two main functionalities, where the first functionality is responsible for the network emulation and the second for the management interface. Section A.1.1 details the network emulation functionality and Section A.1.2 describes the management interface.

A.1.1 Network Emulation

The main objective of the proposed emulation tool is to create an environment where real implementations of the proposed XOR-based flat routing mechanism can be evaluated, supporting the scenarios in which the protocol is instantiated in this work. Such environment is supposed to provide the maximum as possible of the realistic characteristics found in real networks.

In this way, the solution proposed for the development of the emulation tool is to instantiate the vertex (nodes) composing the network graph as individual threads, and the links as TCP connections established between the threads (emulated nodes). The threads pose a full implementation of the proposed routing mechanism, including the signaling exchange required to build the routing tables using the TCP connections (emulated links), and also supporting traffic forwarding between nodes.

Figure A.1 details the architectural modules of a single node in the developed tool. The architectural design is aimed at creating nodes whose internal organization is similar to real routers. There are five main modules in the proposed architecture: 1) Configurations Module, 2) Statistics Module, 3) Fast Memory Module, 4) Engine Module and 5) Physical Module. Each individual module is detailed in the sequence.

Configuration Module

This module is responsible for keeping the configuration parameters related to: 1) particular details of the node being emulated, such as its flat ID, 2) details of the overall network environment,

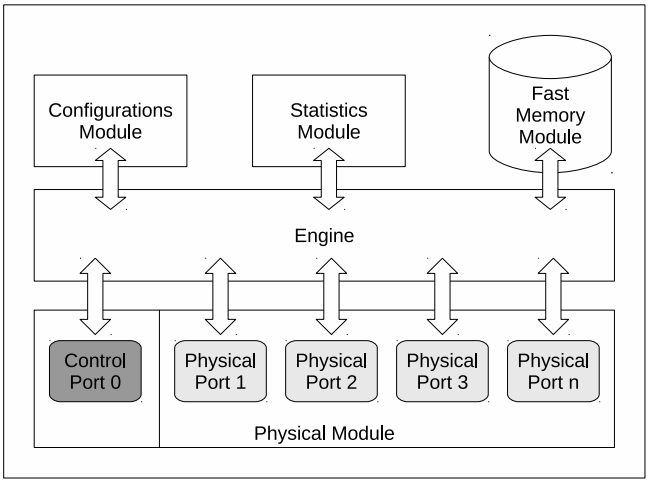


Figure A.1: Architecture of a single emulated node.

like the number n of bits adopted in the flat identity space and 3) specific details of the emulated scenario, such as the manner in which nodes are spread among the available computers. In this last case, the tool offers a configuration file where the distribution rules of the emulated scenario are defined. Basically, ranges of flat node IDs are associated to the IP addresses of the computers where these nodes will be executed during the evaluations. An exemplification is presented in the following lines, where a network composed of 2048 nodes is emulated using four computers.

```
0 511 10.1.1.1
512 1023 10.1.1.2
1024 1535 10.1.1.3
1536 2047 10.1.1.4
```

The configurations are loaded at the moment that the emulated nodes are instantiated. The tool offers a repository to store the topologies used in the evaluations in individual folders, where the files related to each topology are maintained, including the configurations files. Examples of other configuration parameters include: 1) the `DISCOVERY_EXPANSION` value used in the scenario of the data center, 2) the number of bits used in the Bloom filters exchanged between Landmarks in the Internet scenario, 3) the number of hash functions used to create the Bloom filters, also in the Internet scenario and 4) the IP address and the TCP port where the management interface is executed. The rationale of the configuration module is to provide a repository with the essential information regarding the emulation, allowing the engine module of each emulated node to correctly operate in the distributed environment.

Statistics Module

The statistics module is responsible for storing data regarding the entire emulation, contributing to the analyzes of the proposed flat routing mechanism. It offers an extensible API, which is accessible from all modules composing the nodes. Essentially, the functions of the routing mechanism can be implemented including calls to this API. For example, emulated nodes can use the statistics module to register the exchange of signaling messages during the process of building the routing tables, such as `QUERY` and `RESPONSE`. It is also possible to use the API to store information about packets forwarded and/or to check the amount of entries present in the routing tables.

The collected information are individually stored in each node composing the network, and the management interface is responsible to retrieve this information from the nodes. In order to use the statistics module, the only requirement is to extend the API to include the necessary statistical information, and introduce the hookers in the modules responsible for gathering the specific information.

Fast Memory Module

The fast memory module is responsible for keeping in memory the essential data structures of the protocol, such as the routing tables. In a comparison with real routers, this module corresponds to the memories of low access time, such as TCAM, SRAM and DRAM. The main idea is to provide a module which can be easily extended to maintain the critical data structures required by the proposed routing mechanism. This module is reachable from the engine module, since the tasks performed by the engine module are related to it. For example, packets being forwarded by the engine trigger a query in the routing table in order to define the next hop.

Afterwards, during the process of building the routing tables, this module is frequently accessed to retrieve information present in the routing tables in order to generate the `RESPONSE` messages, and/or to store information regarding the neighbors discovered and/or learned. Specifically in the case of the Internet scenario investigated in this work, the fast memory module is also responsible for maintain the LIBs (Landmark Information Bases), where the Bloom filters of the Landmarks composing the *reachability service* are stored.

Engine Module

The engine module is the core of the tool, being responsible for the interactions between all modules composing the emulated node. It is divided in three main functions: 1) control engine, 2) signaling engine and 3) forwarding engine.

- Control engine: it is specifically developed to communicate with the management interface. It contains the required lines of JAVA code to threat the commands received in a node from the management interface. After receiving the commands, the control engine starts the required function and, if necessary, it returns some information to the management interface. There are two types of commands, unidirectional and bi-directional. For example, an unidirectional command is used to ask nodes to start building their routing tables. On the other hand, a bi-directional command is used to recover statistics information;

- Signaling engine: it implements the signaling aspects of the protocol, such as the mechanism for building the routing tables. It is composed of JAVA code to handle the signaling messages exchange, interpreting the fields contained in the messages;

- Forwarding engine: it implements the routing process. This function has access to the routing tables stored in the fast memory module and emulates the process of forwarding packets through the network.

Physical Module

The physical module emulates the physical interfaces present in a node and the links connecting neighbor nodes using the TCP protocol of the sockets' library. Basically, the number of ports present in a given node corresponds to the number of edges that such node has in the network graph. This information is found in files representing each individual node of the graph, which are found in the repository folder of the topology being evaluated. In such files, each line represents an edge of the graph. The following lines exemplify the three edges with neighbors of node 0.

```
0 135
0 513
0 817
```

The proposed physical module also includes a control port, specially developed to provide the communication with the management interface. This port is always instantiated as the port 0 of nodes. To conclude, the physical module offers the flexibility required to emulate any kind of node. For example, in the data center scenario, the emulated servers have six ports, besides the control interface. On the other hand, in the Internet scenario there are emulated ASes with a single neighbor, such as ASes located in the edges of the Internet, and there are ASes with thousands of neighbors, like tier 1 ASes located in the core region of the Internet.

A.1.2 Management Interface

The objective of the management interface is to easily orchestrate the emulated network, offering a graphic interface where commands can be easily dispatched towards the nodes composing the network. At the instant that the interface is executed, it immediately establish TCP connections with all nodes, using the control port 0 present on them. Basically, such connection directly binds the management interface with the control engine of the nodes.

The rationale of the tool is to offer a panel where buttons can be inserted to perform tasks related to the evaluations. For example, it can send commands to the nodes present in the network, asking them to start the process of building their own routing tables or, it can send commands to query nodes about their statistics values. Specifically for the statistics communication, the tool sequentially interacts with the nodes, creating sorted log files to simplify their analyzes. All the log files are stored inside the respective folder of the network being emulated.

The management interface also offers mechanism to inform about the evaluation progress, indicating the amount of paths already computed, and also estimating the remaining evaluation time. In order to compute the paths, the tool sends a command asking a given node to generate a packet towards another node available in the network. As the packet is pushed across the network, information regarding the path is appended in its payload. Such information is returned to the management interface at the end of the path, allowing detailed analyzes of the paths generated by the protocol.

A.2 Complementary Tools

The emulation tool is mainly responsible for evaluating the proposed XOR-based flat routing mechanism in different network topologies. In this way, it is composed of the two main modules described in Section A.1. However, a set of complementary tools are used in conjunction with the emulation tool. Some of these complementary tools were also develop during this work, and other tools are available in the community [108, 120]. Among the complementary tools which were developed in this work are:

- Topology parser: it is responsible for reading the files where the entire topologies are described, creating all the individual files representing the edges of each node, and saving them in the respective folders of the repository;

- Distribution module: it is responsible for spreading files associated to the evaluations in the computers which will be used in the tests;

- Shortest path module: it is an implementation of the Dijkstra algorithm used to generate a log file in which the shortest paths are computed. Such information is essential to compute the route stretch of the XOR-based flat routing mechanism;
- 3-cube topology generator: it is used specifically in the data center scenario for the creation of 3-cube topologies. The 3-cube topology generator is capable of assign flat IDs to servers using a random distribution, creating topologies according to the envisioned scenario proposed for data centers.
- Log analyzes module: this module contributes for processing the log files resultant of the evaluations. It includes tools to compute information regarding the signaling messages exchanged, the number of entries present in the routing tables, the amount of successful path, the gap cases, route stretch and others. Basically, it seeks for the log information in the folders where the topologies are stored, serializing the output of the analyzes in the respective folders.

There are two complementary tools available in the community which were used in this work. Both of them are specific for the Internet scenario. The first one is the BRITE topology generator [108], which offers mechanisms for the generation of Power-law (Internet-like) topologies, such as the topology of 16384 nodes depicted in Figure A.2. BRITE generates a file describing the entire topology, and the topology parser was developed to extract the required information from this file. The second tool used is the OTTER network visualization tool [120], which offers a graphical interface capable of plotting the topologies generated by BRITE, contributing to check the aspect of the evaluated topologies. Figure A.2 was plotted using the OTTER visualizer tool.

A.3 Using the Tool

The use of the tool requires the execution of JAVA processes in the computers involved in the evaluations. Basically, the developed distribution module spreads the required software infrastructure in the computers, and generates a configuration file for each computer, describing the range of nodes which needs to be instantiated on each machine. In the sequence, the JAVA process starts a thread (an emulated node) for each entry contained in the file describing the ranges of nodes. Such threads are able to locate the folder where the topology is stored, reading the required configuration parameters.

Afterwards, it is required to instantiate the management interface to control the execution of the experiments. According to the distributed environment offered by the tool, the management interface can be instantiated in any other machine available. As mentioned before, one of the main characteristics of the tool is its support to operate in a total distributed environment, offering the fundamental basis for aggregating the computational power of several machines in order to scale

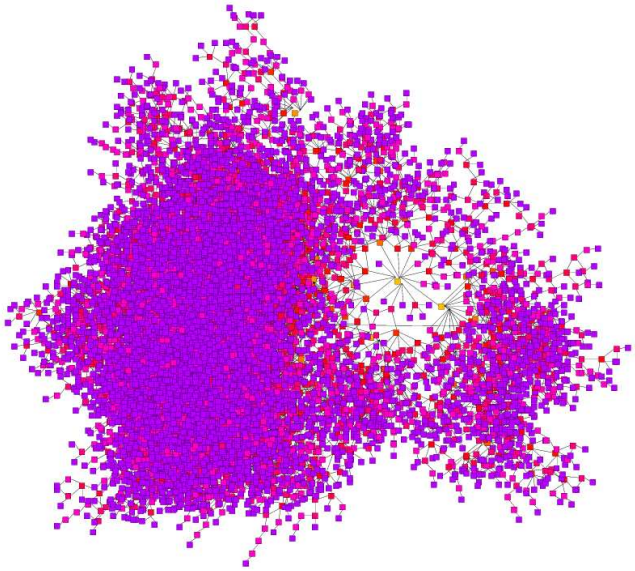


Figure A.2: Internet-like topology with 16384 nodes.

the evaluations to bigger topologies, as the real AS topology evaluated in the Internet scenario with approximately 33,000 emulated ASes.

The distributed environment also allows the use of computational resources through the Internet to perform the evaluations. Specifically in the evaluations performed for the real Internet scenario, they were entirely executed in servers located at Ericsson Research in Stockholm. There are some tests for the data center scenario which were also executed using the Amazon EC2 (Elastic Compute Cloud) service.

Finally, all the log information gathered during the evaluations are stored in the machine where the management interface was executed, simplifying the analyzes of the entire information stored.