Universidade de Aveiro

Departamento de Electrónica, Telecomunicações e Informática

2009

Tiago André Marques
Cristina Carapeto

Recuperação Autonómica de Ligações em Redes Virtuais

Dissertação apresentada à Universidade de Aveiro para cumprimento dos requisitos necessários à obtenção do grau de Mestre em Engenharia Electrónica e de Telecomunicações (Mestrado Integrado), realizada sob a orientação científica da Professora Dra. Susana Sargento, Professora auxiliar do Departamento de Electrónica, Telecomunicações e Informática da Universidade de Aveiro, e do Engenheiro Vítor Mirones, Engenheiro da PTInovação.

o júri

presidente

Prof. Doutor Atílio Gameiro

Professor associado do Departamento de Electrónica, Telecomunicações e Informática da Universidade de Aveiro

orientador

Prof. Doutora Susana Isabel Barreto de Miranda Sargento

Professora auxiliar do Departamento de Electrónica, Telecomunicações e Informática da Universidade de Aveiro

arguente

Prof. Doutor Jorge Sá Silva

Professor auxiliar do Departamento de Engenharia Informática da Faculdade de Ciências e Tecnologia da Universidade de Coimbra

agradecimentos

Este trabalho assinala o fim de mais uma etapa da minha vida académica, mas não o teria conseguido concluir sem a contribuição de algumas pessoas. Por isso, não queria deixar de expressar aqui os meus mais sinceros agradecimentos a todos os que, directa ou indirectamente, contribuíram para a realização desta dissertação de mestrado.

À minha orientadora Prof. Dra. Susana Sargento, pela disponibilidade, apoio e motivação dada ao longo destes meses na realização desta tese.

Ao meu co-orientador Vítor Mirones, por ter partilhado comigo todos os seus conhecimentos, revelando-se fundamental para a correcta abordagem dos diferentes desafios que me foram sucessivamente propostos.

Ao Instituto de Telecomunicações de Aveiro e aos seus colaboradores (principalmente Márcio Melo, Nuno Coutinho, Tiago Condeixa e João Mateiro) por me terem oferecido todo o apoio e condições necessárias para o correcto desenvolvimento do meu trabalho.

Aos meus Pais, Irmãos e Namorada pelo incansável apoio, paciência e motivação que sempre me deram durante o desenvolvimento desta dissertação.

palavras-chave

Redes Virtuais, Nós Virtuais, Gestão Autonómica, Recuperação de falhas, Centralizado, Distribuído

resumo

Com a explosão no número de entidades de rede interligadas e diversidade de serviços fornecidos, verificou-se um grande crescimento das redes de comunicação, levantando problemas a nível de controlo de forma eficaz, por parte dos organismos de gestão de redes. Para combater este problema, possibilitando um crescimento ainda maior e satisfazendo o nível de qualidade de serviço exigido, é necessário uma evolução dos métodos de gestão e controlo da rede. Uma das soluções apresentadas visa a redução do processo de gestão centralizado e da própria interacção humana trazendo a inteligência para os elementos de rede possibilitando uma gestão distribuída e autonómica.

A dissertação apresentada encontra-se inserida no âmbito da gestão autonómica de redes aplicada à recuperação de falhas em redes virtuais como consequência de falhas em ligações físicas. Para este efeito foram desenvolvidos dois algoritmos diferentes tendo em vista a comparação de duas perspectivas diferentes, a perspectiva centralizada e a distribuída.

Para a implementação e teste de toda a arquitectura, recorreu-se ao simulador de redes NS-2 onde, para além de todo o código necessário ao desenvolvimento dos dois algoritmos, foram criados vários cenários possibilitando a avaliação da influência de certos parâmetros em ambos os algoritmos assim como possibilitando a comparação dos algoritmos entre si. Os resultados mostram que o algoritmo distribuído tal como foi implementado, é capaz de recuperar praticamente todas as falhas em ligações virtuais, passíveis de serem recuperadas. Apesar de atingir uma eficiência semelhante à do algoritmo centralizado, demora mais tempo e troca mais mensagens. No entanto estes resultados não têm em consideração o tempo e recursos gastos em processamento, o que ajuda a proporcionar melhores tempos ao algoritmo centralizado.

keywords

Virtual Network, Virtual Node, Autonomic Management, Failure Recovery, Centralized, Distributed

abstract

With the explosion in the number of interconnected network entities and diversity of services provided, there was a large growth of the communication networks, posing problems in the effective monitoring by the network management entities. To overcome this problem, allowing further growth and satisfying the quality level of service required, it's necessary a change the management and control methods of the network. One of the proposed solutions aims to the reduction of centralized management processes, and the human interaction, pushing the intelligence to the network entities, enabling distributed and autonomic management.

This dissertation is in the scope of autonomic network management for failures recovery of virtual networks, as a result of failures in physical links of the network. For this purpose, two different algorithms have been developed in order to compare two different perspectives, the centralized and distributed perspectives.

For the implementation and testing of the architecture, network simulator NS-2 was used. In addition to all the necessary code for the development of the two algorithms, several possible scenarios were also created to evaluate the influence of certain parameters in both algorithms, as well as allowing the comparison between the algorithms. Results shows that the implemented distributed algorithm is capable of recover almost all the possible virtual links failures. Despite reaching efficiency similar to the centralized algorithm, it takes longer and exchanges more messages. However, these results do not take into consideration the computation time and resources, resulting in better results for the centralized algorithm.

Table of Contents

Index of Figures

Index of Tables

Acronyms

#

3GPP	3rd Generation Partnership Project

A

AI	Artificial Intelligence
A/C-RACF	Access / Core-Resource and Admission Control Function

B

BML	Business Management Level ()

C

COPS	Common Open Policy Service
CMIP	Common Management Information Protocol
CS	Computer Science

E

EML	Element Management Level
EMP	External Management Point
ETSI	European Telecommunications Standards Institute

F

FCAPS	Fault Management, Configuration Management, Accounting Management, Performance Management and Security Management

G

GMP	Global Management Point
GIS	Geographic Information Systems

I

ISO	International Organization for Standardization
InP	Infrastructure Provider
ITU	International Telecommunications Union
IETF	Internet Engineering Task Force
INM	In-Network Management

M

MC	Management Console
NETCONF	Network configuration
MIB	Management Information Base
MSF	Multiservice Switching Forum
MS	Management Science

N

NASS	Network Attachment Subsystem
NGNs	Next Generation Networks
NML	Network Management Level
NS-2	Network Simulator 2

O

OMG	Object Management Group
OR	Operations Research
OSI	Open Systems Interconnection
OTCL	Object oriented Tool Command Language
OSS	Operation Support System

P

PBN	Policy-Based Networking
PCC	Policy and Charging Control
PCEF	Policy Charging Enforcement Function
PCRF	Policy and Charging Rules Function
PDN	Public Data Networks
PDP	Policy Decision Points

PEP	Policy Enforcement Points
PR	Policy Repository
PSTN	Public Switched Telephone Network

Q

QoS	Quality of Service

R

RACS	Resource Admission Control Subsystem
RMON	Remote Network MONitoring

S

SNMP	Simple Network Management Protocol
SML	Service Management Level
SMP	Service Management Point
SPDF	Service-based Policy Decision Function
SLA	Service Level Agreement

T

Tcl	Tool Command Language
TISPAN	Telecoms & Internet converged Services & Protocols for Advanced Networks
TMN	Telecommunications Management Network

V

VNet	Virtual Network
VNode	Virtual Node
VPN	Virtual Private Networks
VNP	Virtual Network Provider
VNO	Virtual Network Operator
VINT	Virtual Inter Network Project

1 Introduction

1.1 Motivation

A telecommunications network is a cluster of links and many different devices, arranged and interconnected so that, as primary goal, it enables the exchange of information between end-users or end-systems by exchanging messages through multiple links and network elements. Network management is also a fundamental concept in a telecommunications network, since it ensures an effective and efficient way of controlling, planning, allocating, deploying, coordinating, and monitoring the network elements and resources.

The traditional approach to network management architecture is based on a centralized paradigm where dedicated stations and servers support management functions communicating with the network elements using standard management protocols. However, with all the technological advances in electronics and transmission systems (with the breaking of the wired barrier), the increase in bandwidth, mobility, the massification of broadband, the growth in number of users and service providers, telecommunications network are becoming much more complex (increase on the number and variety of devices) and dynamic (constant changes on the network) requiring longer periods and higher costs of deploying, configuring and operating networks, making the centralized network management paradigm obsolete.

Future networks are expected to become even more complex, dynamically providing the interaction of multiple different networks and technologies into a global network environment. Besides that, the notion of user will also be re-defined; e.g., houses, vehicles, pets, etc., everything may become a network user and will be connected to the future Internet.

Since centralized network management may no longer be applicable to the large-scale networks and services foreseen in the future, new approaches that confer intelligence to the network are required, enabling network entities with the ability of

self-managing their behaviours, distributing the work load and automatically executing the management tasks reducing the human interaction (simplifying processes) and the centralized control, aiming at reducing the reaction times to changes. This thesis presents distributed autonomic management as a solution for simplifying the processes and scalability issues. It also presents some preliminary results on the comparison between centralized and distributed network management.

1.2 Objectives

This Master Thesis is in the scope of 4WARD [1] project, more precisely in restoring virtual links by autonomic mechanisms. 4WARD is a European Union project of the 7th Framework Program and addresses the Architecture and Design for the Future Internet. 4WARD aims to increase the competitiveness of the European networking industry and to improve the quality of life for European citizens by creating a family of dependable and interoperable networks providing direct and ubiquitous access to information [2].

The main objective of this thesis is to present an autonomic distributed network management algorithm over virtualized networks built by an operator, allowing the operator to recover automatically from failures on the physical infrastructure and find alternative virtual links for the network traffic.

To fulfil this objective, for baseline comparison, this thesis implements an architecture based on a centralized approach, where a centralized intelligent entity is responsible for gathering all the network information, and therefore, when facing physical link failures, be able to take all the decisions in order to re-establish the affected virtual links (above those physical links), taking into account all the resources they require. This thesis also implements an architecture based on a distributed approach, where every network entity is responsible for gathering essential information from all its direct neighbours, for once more, when facing physical link failures, be able to take decisions in cooperation with its neighbours in order to re-establish the affected virtual links taking into account all the resources they require.

In order to implement, evaluate and compare the proposed solutions, Network Simulator (NS-2) was used, allowing the evaluation of different scenarios with various different characteristics. This implementation aims to answer to the questions of if a distributed algorithm is capable of achieve a solution, if it can outperform a centralized algorithm, and if it can offer better scalability and adaptability in larger networks.

1.3 Contribution of this work

As a result of the accomplishment of the above proposed objectives, this work provides the following set of contributions:

- The development of a network architecture that enables autonomic and distributed mechanisms and decisions, providing to all virtual networks the capacity of distributed self-repairing from a physical link-failure, maintaining all the previously agreed conditions (during the virtual network setup).

- An evaluation, via simulation, of both architectures performance in different fields and its response in different situations.

1.4 Document Outline

The work developed in this Thesis is organized in six main chapters. Each one explains briefly the work performed in different phases of the research, introduction, state-of-the-art, conceptual ideas, implementation, evaluation of the architecture and conclusion/future work.

The present chapter contextualizes the thesis in the current and future generation network management paradigms and presents its objectives and contributions.

The second chapter presents the state-of-the-art related to network management. It presents an overview of network management concepts and protocols, as well as an overview on In-Network Management concepts and also an overview on the current research in Network Virtualization.

The third chapter presents the main ideas, concepts and mechanisms envisioned for Centralized and Distributed paradigms.

The fourth chapter presents the architecture implementation performed in the Network Simulator (NS 2.31). This chapter is divided in sub-chapters, according to the different paradigms (Centralized and Distributed).

The fifth chapter presents an evaluation of the implementation made by testing the efficiency and performance of different paradigms in several scenarios and conditions.

Finally, the sixth chapter summarizes the main results and a description of the performed research. It also describes future work to be carried out after this Thesis, issued from an evaluation of the results and the identification of possible improvements in the developed algorithms and methods.

2 Background Information

This section presents the background information researched for the duration of this thesis, that supported the implementation of the algorithms here described, and the achieved conclusions. Section 2.1 will present an overview on Telecommunications and Network Management as well as some management protocols. Section 2.2 will present an overview on the concept and architecture of the In-Network Management. Section 2.3 will present Network Virtualization concepts, motivations, challenges, and main elements, as well as its management, and finally Section 2.4 will present the resume of the entire chapter.

2.1 Telecommunications and Network Management Overview

Network management is treated as an inconvenience by equipment manufacturers. The traditional approach to network management is based on a centralized architecture. To be managed, the equipments have a database of management information (MIB) [3] held by a management agent to which the management applications access via a network management protocol (Simple Network Management [4][5][6] (SNMP) within the Internet community, and Common Management Information Protocol [7] (CMIP) in the OSI community). Nevertheless, the ability to manage network devices is one of the most important aspects of today's telecommunications. Network management includes all the tools, methods, procedures and activities to configure, operate, administrate, maintain and provision networking systems. Usually, networks are spread for huge distances and may contain different elements, such as routers and switches, which were made by different manufacturers. Standard bodies such as the ISO (International Standards Organization), the ITU (International Telecommunications Union), the TM Forum (TeleManagement Forum), the IETF (Internet Engineering Task Force), the OMG (Object Management Group), and the MSF (Multiservice Switching Forum), concern substantial time and effort to specifying architectures, data schemas, and management communication protocols, aiming to standardize the way in which networks and network equipment are managed.

ISO defines network management tasks into five distinct categories known by fault management, configuration management, accounting management, performance management, and security management (FCAPS):

- operations that keep the network up and running, and services operating properly;
- monitoring systems that provide information on the status of the links and elements thus enabling the other procedures;
- resource management activities that keep track of the expectations of users versus the quality of service being provided plus control of the cost/efficiency relation;
- maintenance of equipments, repairs and upgrades;
- configuration and adjustment of diverse equipment parameters and configuration for a better operation;
- provisioning of resources and services.

Since the current management paradigm is centralized and is oriented to small networks of fixed devices where topology changes occasionally, resources are statically reserved and, when some abnormal situation occurs, the reaction time is variable since it depends on the time spent to report the situation to the central system, and the equipment or human capacity of analyzing and taking a decision. This makes the current management paradigm inadequate for the future scenarios where networks are expected to become bigger, more complex and more dynamic.

Thus, research has been carried out in this area in order to solve the previous problems.

2.1.1 SNMP

The SNMP stack [4] was introduced by the IETF as part of the internet protocol suite. It is a typical management protocol of TCP / IP formed by a set of standards for network management. Lately, this protocol experimented some developments including now a good amount of improvements such as additional protocol operations, new protocol data units (PDUs), improved performance, security, confidentiality and communications manager-to-manager SNMPv2 [4]. Finally, a new version, the current standard SNMPv3 [6][7] contains improvements in security and remote configuration enhancements. SNMP is based on the manager/agent model and consists in the following elements:

- An **SNMP manager** which provides the interface between the human network manager and the management system;
- An **SNMP agent** that is the interface between the manager and the physical device(s) being managed.
- A **MIB**, the management information database which includes all the information of each managed devices. It is organized in a tree structure with individual variables, such as point status or description, being represented as leaves on the branches. A long numeric tag or object identifier (OID) is used to distinguish each variable uniquely in the MIB and in SNMP messages.
- **Managed devices**, e.g. routers, switches, modems, printers…
- A **Network Protocol**, usually UDP.

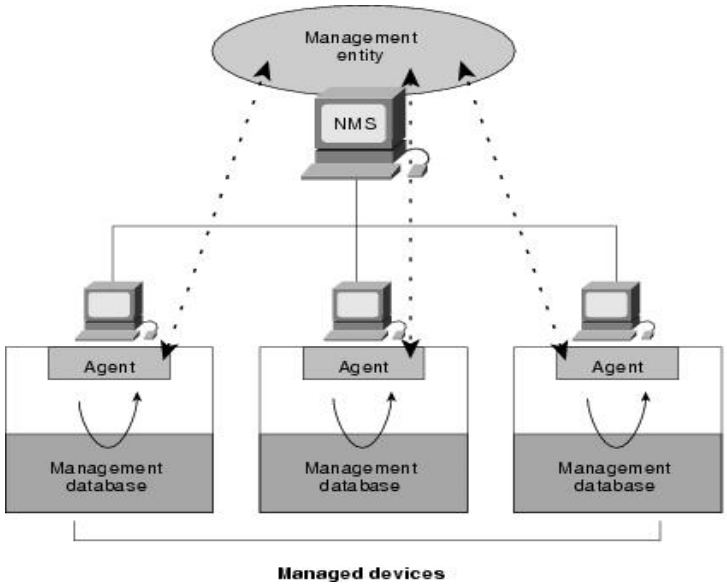


Figure 1: An SNMP-Managed Network - Managed Devices, Agents, and NMSs

The SNMP manager and agent use an SNMP Management Information Base (MIB) and a relatively small set of commands to exchange information.

SNMP uses five basic messages (GET, GET-NEXT, GET-RESPONSE, SET, and TRAP) to communicate between the SNMP manager and the SNMP agent. The GET and GET-NEXT messages allow the manager to request information for a specific variable.

The Remote Network Monitoring (RMON [8]) MIB was developed by the IETF to support monitoring and protocol analysis of Local Area Networks (LANs). Initially, it focused on OSI Layer 1 and Layer 2 information in Ethernet and Token Ring networks. Lately, it was extended to RMON2 with the support of Network- and Application-layer monitoring.

2.1.2 NETCONF

NETCONF [9] or Network configuration is a network management protocol developed by the IETF within the NETCONF working group and published as RFC 4741, in December 2006. The NETCONF protocol provides mechanisms to install, manipulate, and delete the configuration of network devices and perform monitoring

functions. It was developed to use XML (Extensible Markup Language) data encodings for the configuration data and the protocol messages over a simple Remote Procedure Call (RPC) layer. This is done on top of a transport protocol: SSH (RFC 4742), SOAP (RFC 4743), BEEP (RFC 4744), TLS (RFC 5539), etc. NETCONF was introduced since most operators were primarily using proprietary Command Line Interfaces (CLI) to configure their boxes over SNMP. CLI was being preferred due to its simplicity and easiness of use.

2.1.3 Policy-Based Networking (PBN)

The IETF's Policy Framework group developed the Policy-Based Networking (PBN) [10] concept and architecture that has the objective of enabling centralized network administration. This was done through the definition of abstract rules, and specifying what the network should do using a high-level language instead of specifying how to do it. This is done in a network-element-independent way, through a policy specification language.

As can be seen on Figure 2, four basic components were specified: Management Console, Policy Repository, Policy Decision Point and Policy Enforcement Point; and two protocols: repository access protocols and policy transfer protocols to be used in between.

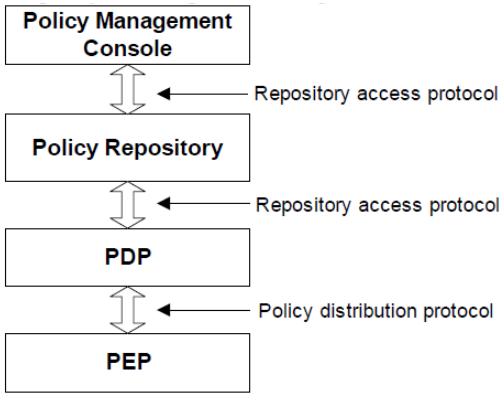


Figure 2: Components of PBN Architecture

The *Management Console* (MC) is the top most element in the framework and allows provides an interface, typically web-based, to the administrator and the policy

management system. This allows the specification, edition, translation and validation of the policies stored in the policy repository.

The *Policy Repository* (PR) stores the applicable policies to a given administration. These policies database are structured as a Policy Information Base (PIB), which resembles the structure of the SNMP Management Information Bases, independently of each implementation approaches.

The *Policy Decision Points* (PDP) or Policy Servers translate high-level policies stored in the policy repository into lower level policy rules. By combining this information with other network information such as status, PDPs can produce the precise rules to be implemented in the Policy Enforcement Points.

The *Policy Enforcement Points* (PEP) receive rules from PDPs and deal with network elements accordingly. PEPs that are integrated with network elements can perform actions such as packet filtering, packet marking in addition to other resource management operations. In addition to enforcing the rules established by PDPs, PEPs can send information to PDPs as, for instance, is the case when physical configurations change.

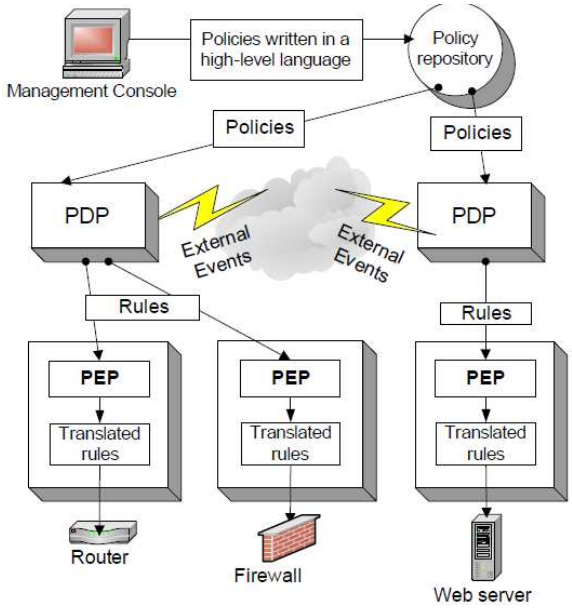


Figure 3: Typical PBN Configuration

Figure 3 presents a typical PBN configuration. Communication between PDPs and PEPs is achieved using a policy transfer protocol such as the Common Open Policy Service (COPS) protocol

2.1.4 Common Open Policy Service (COPS) Protocol

The Common Open Policy Service (COPS) Protocol is a standard that specifies a simple client/server model for intra-domain signalling that supports policy control over QoS signaling protocols. It is used between a Policy Enforcement Point (PEP), e.g. an edge router that stores the policies and a Policy Decision Point (PDP), e.g. a bandwidth broker. Cops was defined from IETF's RFC 2748 [11] and RFC 3084 [12] where two models where specified: The Outsourcing Model and the Provisioning Model depending on the view of the client or PEP.

The Outsourcing Model is the simplest one, since all policies are stored at the PDP and whenever the PEP needs to make a decision, it sends all relevant information to the PDP. Then PDP analyzes the information, makes the decision, and relays it to the PEP who simply enforces the decision. The Provisioning Model [13] has the PEP reporting its decision-making capabilities to the PDP who downloads relevant policies on to the PEP and use them to make its own decisions. This model uses the Policy Information (PIB) Base as a repository of the policies.

2.1.5 Telecommunications Management Network

The Telecommunications Management Network (TMN) [14] model concept model was introduced in 1985 by ITU-T Recommendation in series M.3000, and is an architectural framework for the interconnection of different types of OSS components and network elements. Initially it was introduced by ITU-T as a reference model for the Operation Support System (OSS) with the purpose of defining how to manage heterogeneous networks, services and equipments from a wide variety of different manufacturers and technologies. TMN is based on the OSI management specifications in ITU-T Recommendation series X.700 and also describes the standardized interfaces and protocols used for the exchange of information between OSS components and network elements, plus the functionalities needed for network management.

The framework identifies four logical layers of network management which are applicable within an OSS:

- **Business Management Level (BML)**
  - Performs functions related to business aspects, analyzes trends and quality issues, for example, or to provide a basis for billing and other financial reports.
- **Service Management Level (SML)**
  - Performs functions for the handling of services in the network: definition, administration and charging of services
- **Network Management Level (NML)**
  - Performs functions for distribution of network resources: configuration, control and supervision of the network.
- **Element Management Level (EML)**
  - Contains functions for the handling of individual network elements. This includes alarm management, handling of information, backup, logging, and maintenance of hardware and software.

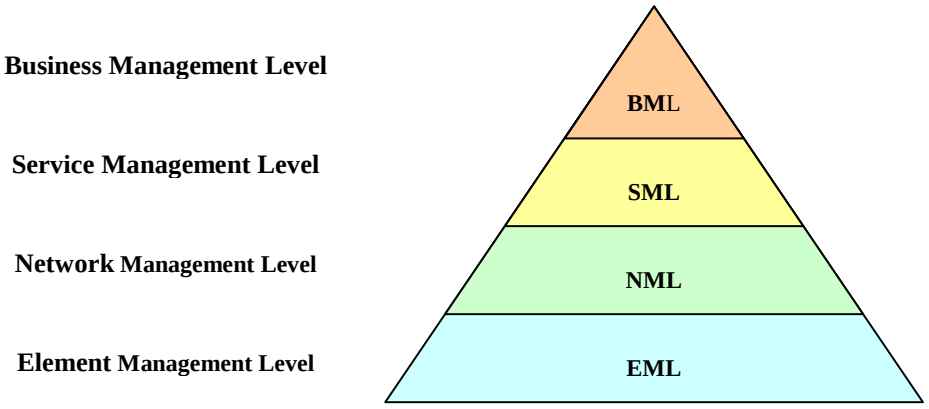


Figure 4: The OSS Layers

At each layer in the TMN model, five functional areas: Fault, Configuration, Accounting, Performance, Security (FCAPS) are the basis of all network management systems for both data and telecommunications.

- **Fault Management**: Fault recognition, isolation, reporting and recording.
- **Configuration Management**: Installation of network equipment, setting of states and parameters, configuration of network capacity.
- **Accounting Management**: Collection, buffering and delivery of payment and accounting information.
- **Performance Management**: Collection, buffering and delivery of operating statistics for network optimization and capacity planning.
- **Security Management**: Administration of authorization functions; handling of simultaneous use of an OSS, protection against intrusion from unauthorized users.

The mapping between the TMN and FCAPS is shown in Table 1 below:

	Fault	Accounting	Configuration	Performance	Security
Business Management	**No**	**Yes**	**No**	**Yes**	**Yes**
Service Management	**Yes**	**Yes**	**Yes**	**Yes**	**Yes**
Network Management	**Yes**	**No**	**Yes**	**Yes**	**Yes**
Element Management	**Yes**	**No**	**Yes**	**Yes**	**Yes**

Table 1: mapping between the TMN and FCAPS

TMN is a framework that enables interconnectivity and communication across heterogeneous systems and telecommunication networks, by defining a set of interface points to the elements which perform the actual communications processing (such as a call processing switch) be accessed by other elements, such as management workstations, which monitor and control them. The standard interface allows elements from different manufacturers to be incorporated into a network under a single management control.

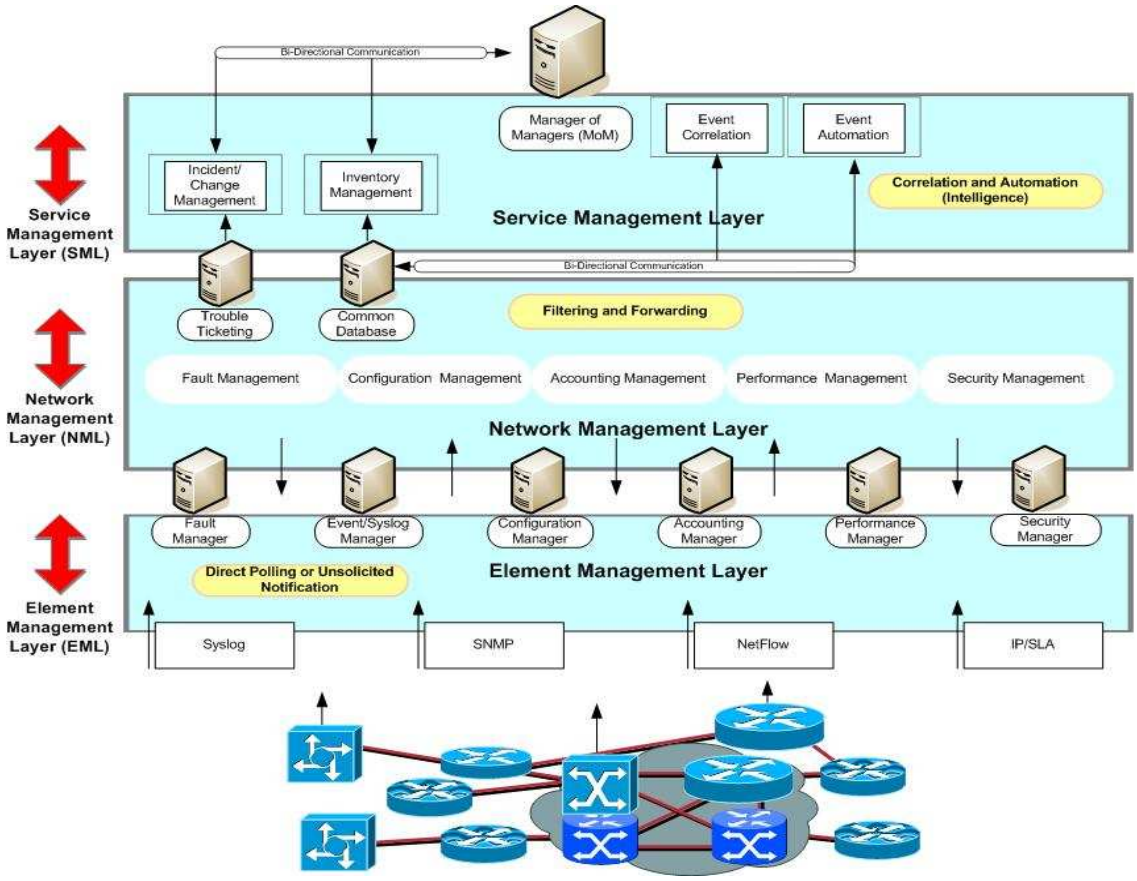


Figure 5: Network Management System (NMS) Architecture

2.1.6 "Next Generation Network" Management Architecture

The NGNs (Next Generation Networks) standards are already implemented by several operators in their networks and allow them to provide more efficiency in the management of their customers, services and the used resources. Today's existing standards are an evolution of more traditional infrastructures network control and resource management schemes although divided into narrower functionalities.

With a broader number of objectives and functionalities, the standards and architectures that support NGNs present in its core the ability to provision resources and provide Quality of Service. For this matter, the major standardization bodies defined the current edge mechanisms of telecom operator technologies. ETSI/TISPAN presented the TISPAN NGNs used for most wired and wireless (IEEE compatible) communications, and the IMS for 3GPP which support the cellular networks. With 3GPP release 7, both have been combined into a single one, the 3GPP/TISPAN IMS Architecture as presented in Figure 6. The combined architecture can be seen as part of

the FMC (Fixed-mobile convergence) trend and allows resource reservation between different cellular and internet services as well as call and service continuity over the different technologies simultaneously.

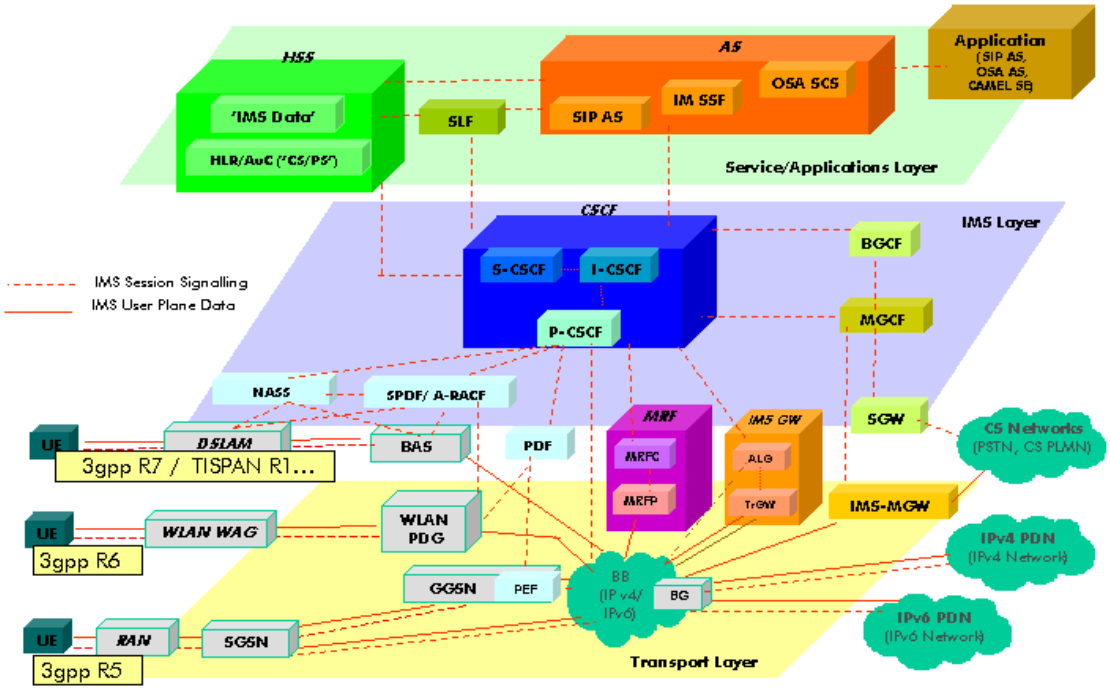


Figure 6: 3GPP / TISPAN IMS Architectural Overview

The European Telecommunications Standards Institute (ETSI), more concretely the Telecoms & Internet converged Services & Protocols for Advanced Networks (TISPAN), has developed its own standard for NGNs [15] based on three major layers (or subsystems) that can be observed in Figure 7. These are Services and applications, Network Attachment Subsystem (NASS) and Resource Admission Control Subsystem (RACS).

The services and applications subsystems are responsible for the most human interaction and support [16]. The Network Attachment Subsystem includes the information about users, and the Resource Admission Control Subsystem is mostly responsible for managing resources on the operator network; this includes the definition of QoS paths, the evaluation of resources on each individual link (and end to end) plus the support of mobility. Finally, for the effects of this thesis, the interest is centred on

the RACS which can be divided in two distinct functional entities, the SPDF (Service-based Policy Decision Function) and the Resource Admission Control Subsystem

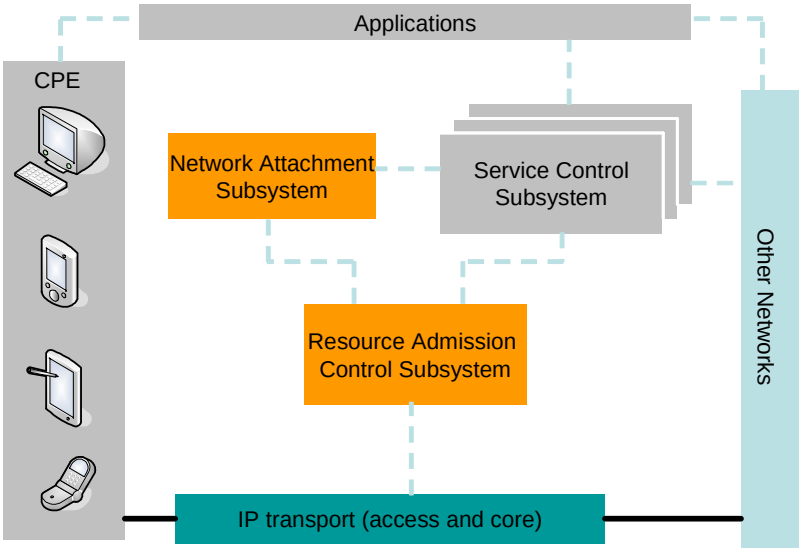


Figure 7: TISPAN NGN Architecture

The ***SPDF (Service-based Policy Decision Function)*** receives information through the Gq' interface from the Application Function, which are normally requests for network resources. This element hides all the details concerning the physical network. The SPDF is a policy-based operator that, each time an application receives a request, can accept it and forward the request to the A-RACF through the Rq interface. The SPDF is also responsible for the configuration of BGFs (Border Gateway Function) Packet marking, QoS and opening firewall open doors or controlling NAT.

The ***A/C-RACF (Access/Core-Resource and Admission Control Function)*** is responsible for performing admission control whenever this is requested by the SPDF (through the interface Rq). The admission control process is performed based on the profile of the requesting user, the network status at the time of application and any other applicable policies.

In parallel, 3GPP (3rd Generation Partnership Project) specified the 3GPP TS 23.203 definition which defines the architecture for Policy and Charging Control (PCC) as can be seen in [17].

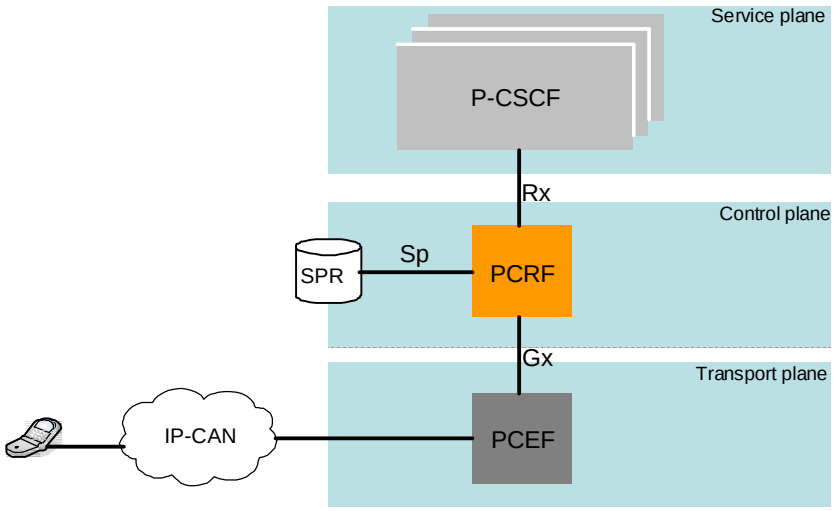


Figure 8: PCC Architecture from 3GPP

The PCRF (Policy and Charging Rules Function) has the objective of deciding which QoS Policies and charging should be executed by the PCEF (Policy Charging Enforcement Function) during the session setup. The configuration of PCEF concerning packet marking and port opening is also part of PCRF.

2.2 In-Network Management

Management functions [18] based on current network management technologies are usually placed in management stations interacting with network elements and devices via management protocols, in order to execute management tasks (FCAPS).

Although this approach have proven to be fairly successful in moderated size networks with nearly flat configurations, whose states evolve slowly and that do not require rapid intervention by external system, this type of approach has some limitations such as lack of scalability and slowly adaptation to changes in many of today‘s emerging networks, whose management domains can potentially include millions of network elements with variable configurations and whose state can be highly dynamic.

Thus, in order to achieve scalable and robust management systems with low complexity and capable of adapting to changes in todays emerging networks, a new approach for network management was proposed: In-Network Management (INM) [18]. This approach basis itself in decentralization, self-organization, embedding of

functionality and autonomic principles, aiming to bring network management tasks or subtasks, performed on outside stations, for the inside of the network (Figure 9).

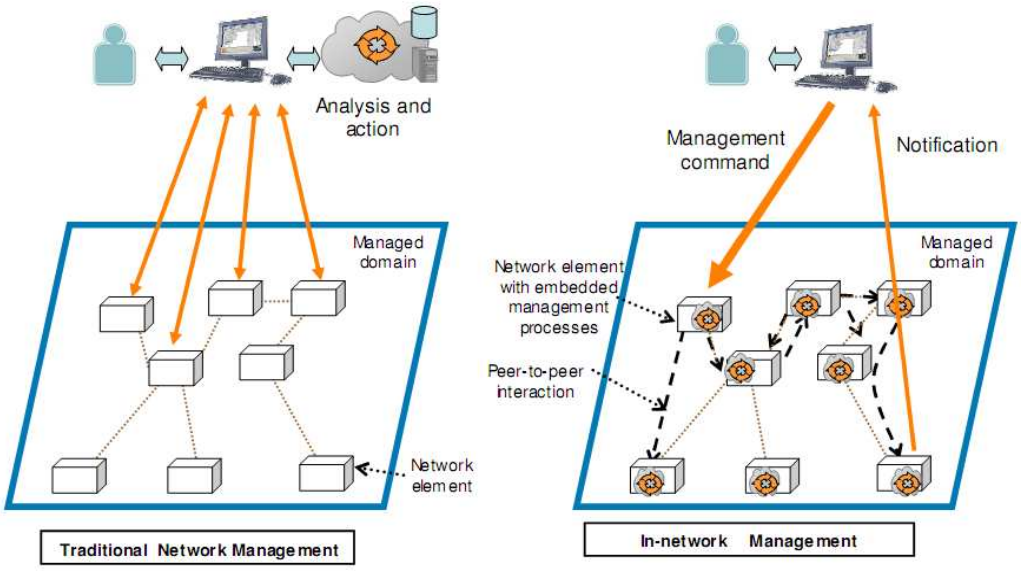


Figure 9: Paradigms for network management: traditional vs. and the In-network management [19]

This approach can be understood as an attempt to embed the intelligence into the network, once it includes the development of a network management plane that self-configures and dynamically adapts to changes in network topology, failures and external events. On the other hand, to realize this plane, a management entity with processing and communication functions has to be associated with each network element or device, which, in addition to monitoring and configuring local parameters, communicates with peer entities in its proximity. The collection of these entities creates a thin layer of management functionality inside the network that can perform monitoring and control tasks end-to-end (Figure 10). This plane does not try to replace the outside management entity but rather, to reduce the rate of interaction between this entity and the network; this plane tries to complement it, once all the results of management actions are reported to this entity in order to provide outside intervention if anything goes wrong or even if business objectives and policies configurations are needed.

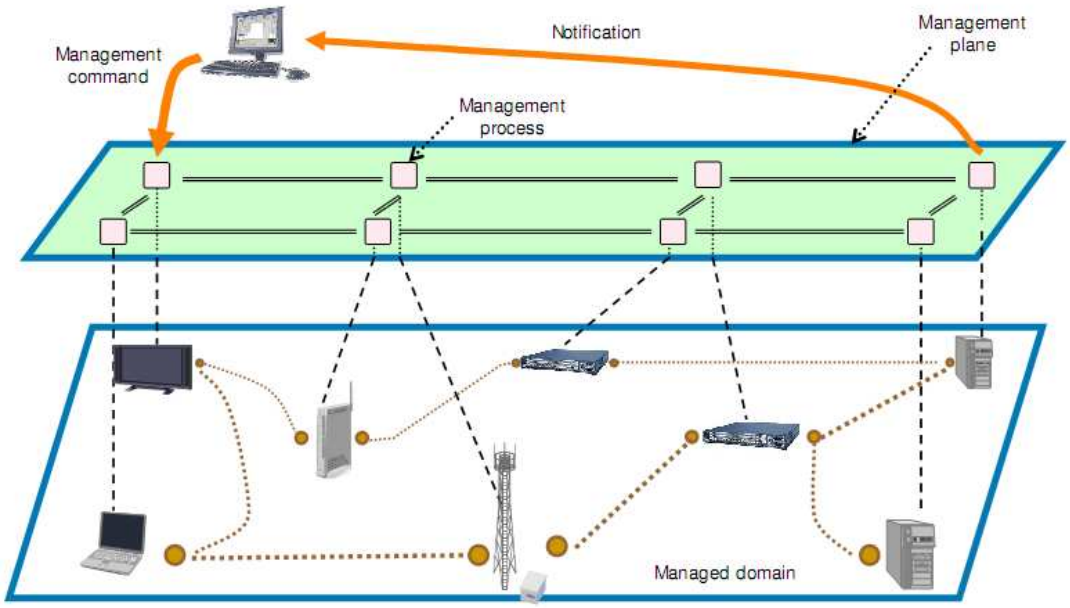


Figure 10: Presents the management plane for In-Network Management [1]

In this section it will be introduced the fundamental principles and elements of the INM framework that are the basis for designing and implementing embedded management processes.

The purpose of INM is to introduce cost-effective instruments to operate the future Internet. Two main actors are main actors stand into the overall picture of 4WARD:

Provider: entity who is responsible for and operates network resources (either physical or virtual resources). To the provider, INM offers a set of interfaces to operate those resources and to respect Service Level Agreements (SLAs) with users;

User: entity that requests and uses provider's resources as services (can be assumed as end-user as well as provider).

After understanding who the main actors are, and given the legal relationships between them, an SLA is used to define the type of service delivered to the user and the guaranteed quality in delivering it. An SLA's technical description is therefore used as input by INM to configure services.

Since the traditional FCAPS model cannot easily be adopted to trigger embedded and distributed management capabilities featured by INM, a new set of high-level interfaces with specific requisites of abstraction, self-configuration, feasibility and feedback mechanisms, must be introduced to constitute the basis for three types of management points (Figure 11) that define the set of operations accessible to INM and the actors by which they are accessed. Those management points are:

Global management point (GMP): Different from traditional GMP's, it provides operations to define network-wide objectives more abstract, less complex, and closer to business goals to construct global indicators from the network.

Service Management Point (SMP): Different from traditional management, it enables a high degree of automation and reliability in service provisioning and executes operations to activate embedded management capabilities for the correct delivery of network services.

External Management Point (EMP): It receives operations that were not supported through embedded management capabilities, which are needed for complete management architecture, such as operations that are out of the scope of self-management, and operations that do not perform efficiently through distributed algorithms.

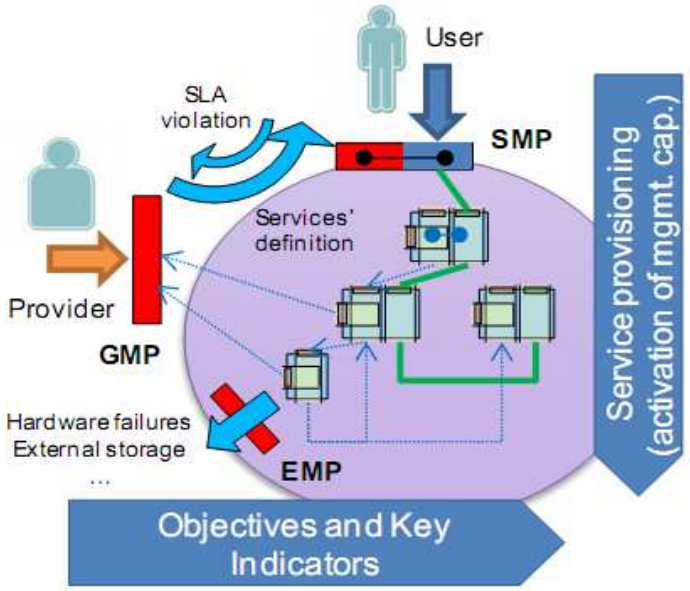


Figure 11: Managements abstractions in INM [18]

Self-management is initiated and maintained within the cloud delimited by these management points.

The INM framework stipulates five fundamental principles that will guide the design of management capabilities in the future Internet:

Intrinsic principle: This fundamental principle captures the fact that the network is the management entity at the same time (management is intrinsic to the network). The following three principles are consequences of this principle and support the clean slate design of embedded management functions in the future Internet.

Inherent principle: Management is an inherent part of network elements, protocols, and services. As such, management functions come an inseparable, i.e. co-designed part of the network.

Autonomous principle: In an extreme case, management is autonomous and does not involve any external manual intervention. This principle would lead to the adoption of a fully self-organizing management plane, which would also automate the enforcement of high level business goals and physical intervention. This principle is clearly not feasible in its pure form, but it defines long term objectives for the research in automation of the future Internet.

Abstraction principle: External management operations occur on the highest possible level of abstraction. In the theoretical extreme case, the network may be triggered by an external stimulus only once at the beginning of its lifetime. All subsequent management actions and processes are concealed and autonomous in the sense of the autonomous principle. This principle guides us towards the definition of management interfaces for operators that hide internal self-management processes more than today‘s approaches.

Transition principle: The INM framework defines that the architectural design principles 2-4 should be implemented and developed in operative networks in a way that they can be gradually adopted. This principle is essential in that it allows gradual deployment of self-management technologies and, in particular, assures marketability of INM results.

The INM framework also defines three key INM artefacts that provide elements by which the INM framework is realized:

Architectural elements include some components which are the construction building blocks from which INM functionality is constructed including: ***Management capabilities***, ***Functional Components***, ***INM entities***; ***INM deployment environment*** that facilities the instantiation, deployment and operation of the architectural elements within the self-organizing management plane; ***INM algorithms*** for providing situation-awareness, anomaly detection, and self-adaptation mechanisms. It uses the architectural elements and their interaction to realize their solutions and then deploy within the INM deployment environment.

In-Network Management capabilities aim to allow for a level of real-time awareness of the network behaviour since it is envisioned that the management plane will typically reconfigure within a sub-second, when facing node, link addition or failure, and resume correct operation thereafter (that is not feasible with present management technology). Such a capability will significantly reduce the reaction time of today's centralized management systems, making INM especially beneficial in large networks with dynamic network environments.

2.3 Network Virtualization

The internet has grown a life of its own and evolves at a tremendous rate. Still, Internet itself is aging fast and suffering hard from some of the initially imposed limitations. The rapid growth in size is accompanied by a severe growth in complexity and maintenance costs for its actors. In many ways virtualization emerges as a possible salvation board and also, somewhat unexpectedly, as a hopeful solution to one of the most serious problems of today's networks: the scalability of self-management of networks and resources [20][21][22][23].

Virtualization itself is not a recent concept [24] and has been used mostly to reduce deployment time and complexity and also to isolate failure domains, improve

resource reservation efficiency and test environments for new strategies over already available network structures. Virtualization also appears as a way to revolutionize the various aspects of a service provider, through the numerous advantages it offers.

2.3.1 Motivations towards Network Virtualization

There are several well known motivations that are pushing operators and researchers towards the concept of networks virtualization: economical reasons, security and logistic issues.

Network Virtualization allows an increase on revenues, since it allows sharing infrastructures and equipments plus an optimization of physical resources [2][25]. Operators can place several virtual networks over their physical infrastructure to provide different and concurrent services. At the same time, virtualization permits the reduction of costs in the acquisition and upgrade of physical infrastructures by sharing and aggregating older and new equipments. Expensive equipment that supports high bandwidth connections can be replaced with multiple lower bandwidth ones, the traffic split without a major impact on QoS. In the same way, resources like high-speed interfaces and WAN links can be shared among the hardware logical routers minimizing the consumption of power and space. Multiple and different services are able to run using a single virtualized infrastructure instead of running above separate networks, allowing once again a reduction of costs [2].

Network Virtualization also makes possible real partition and isolation of physical resources. Traffic belonging to a specific Virtual Network (VNet) is only visible to elements from this VNet. This increases the security of the overall network by creating different security domains, each for different users and for separated access to network resources.

Another key aspect of Virtual Networks is its flexibility and the easiness in which it can be adapted to different scenarios and services. It allows an easy replacement of damaged links and nodes, plus a fast recovery from fail situations. Replacing equipment and re-routing traffic is made easier, since many fallback configurations are made possible. Furthermore, this simplifies the implementation of

automatic mechanisms that can reconfigure and update the network without a proactive human assistance.

2.3.2 Challenges of Network Virtualization

Virtualization techniques are still far from achieving maturity, but due to recent focus on research on this area, they are evolving at a tremendous rate. Still, the main challenges for architectures that support the management of VNets are the same [25]:

- Scalability: Numerous virtual networks must be supported in parallel and this number or the complexity of the architecture that supports them, must not add significant impact on the QoS they provide.
- Quality of Service: Services running over VNets must be able to be aware of the QoS being provided to them and these resources must be able to be provisioned so that the QoS is guaranteed. Services may be running on a VNET in Best-Effort on in a defined class of service as if they were running on the physical network.
- Inter-technology: Virtual networks should support different physical technologies and its implementation must be possible over any access or core network.
- Inter-VNETs: Different architectures should allow migration between themselves avoiding “network islands” isolated of different technology versions.
- Security and privacy: Virtual networks access and management functions, especially if provided by a third party, must be properly secured and the information kept private and only accessible with proper authorization.
- Adaptable on runtime: The VNets must be adaptable on runtime and be able to be expanded, modified and terminated while online. Moreover, problems should be possible to be corrected without requiring the framework to go offline. This also includes resource provisioning.

2.3.3 Network Virtualization Concept

To better describe Network Virtualization it is necessary to take a look back into some historical perspective and realize why this concept is in fact an evolving paradigm. Way before the actual concept of "Virtualization", the Public Data Networks (PDN) (i.e. the global Internet) were considered as the virtual network, since they allowed the transport of data without the traditional physical means over a wire. PDNs allowed any connected entity to exchange data with others connected entities similarly to Public Switched Telephone Network (PSTN). Later, Virtual Private Networks (VPNs) were developed to provide elusive multiple different networks over a common large scale network infrastructure, their concept is normally bound to a specific protocol and they are unable to provide a real partitioning and isolation of physical network resources. In fact, VPNs only provide traffic isolation. On the other hand, overlay networks can provide incremental deployment, instead of a mass deployment on every network element. They are ineffective because they have a limited visibility to the underlying network.

In the beginning of the remotes 1960s, the term Virtualization was used to refer to a virtual machine, and only recently it changed, and the word became broader, including the possibility of virtualizing other elements as network components, end systems, storage functions, operative systems and even full meshed networks. Since most of the costs are now related to operational expenses, it became essential to rethink how networks are operated, aiming to reduce those expenses, in parallel with the creation of new services.

2.3.4 Actors in a Virtual Network

In principle, virtualization still has the same actors as traditional networks. The current complexity can introduce new roles on the existing actors or split existing in multiple ones. This kind of Virtualization enables the expected decouple between infrastructures and services, providing new business models and roles [2][25]. Some of the roles that were defined by the research community, as in 4WARD are:

- Infrastructure Provider (InP)

- Is responsible for deploying and running the network physical resources, be them the network cables, the network and wireless elements;
- InP provides access to the physical resources and allows the partitioning of those resources into isolated virtual resources. InP must have information about the resources allocated to each virtual network but does not know anything about the protocols that may be running inside.

- Virtual Network Provider (VNP)
  - Is responsible for building all the virtual infrastructure over the physical network, provided by other InPs;
  - Controls any communication between one or multiple InPs in order to negotiate and collect the needed set of virtual resources required by Virtual Network Operator (only provides means for Virtual Network Operator being able to make the VNet to come alive).

- Virtual Network Operator (VNO):
  - Is responsible for deploying, operating and controlling the VNets constructed by VNPs;
  - It has access to its virtual resources and they are equivalent slice of physical ones, allocated by a VNP.

2.3.5 Physical Elements in a Virtual Network

Substrate Network Level is the physical nodes. The Substrate nodes are network elements, normally routers, which are able to support Virtualization, allowing the creation of one or multiple Virtual nodes into each one. There are not any restrictions about the means of Virtualization that could be used for this purpose. Besides the type of Virtualization used, Substrate nodes can be divided in three groups:

- Edge nodes (could support Virtualization):
  - Like the name suggests, these nodes are located at the edge of the substrate network so, normally they are connected to end users (directly

or through an access network provider) and their geographical location is a key attribute.

- Core nodes (could support Virtualization):
  - Once again, like the names suggest, these nodes are located at network's core and they could be Virtualized (have one or more Virtual Nodes) or not (passing node);
  - These nodes are also able to be passing nodes for a Virtual Network and at the same time have virtual nodes from other Virtual Networks. These nodes may be connected to edge nodes and/or connected to other core nodes.

- Border nodes:
  - These nodes are located at the borders of networks domains, are connected with core nodes and also with others border nodes from different domains.

Substrate links are network elements between two substrate nodes, responsible for their interconnections. Depending on their bandwidth, these links may or not be able to support one or multiple Virtual links above them.

2.3.6 Virtual Elements in a Virtual Network

Virtual Network Level is a level that may have multiple layers, each one for each Virtual Network into the Substrate Network. In order to better understand Network's Virtualization, it is necessary to know their basic building blocks which are Virtual Nodes and Virtual Links:

- Virtual nodes:

- Node Virtualization can be seen as a division (e.g. CPU, memory, storage capacity, link bandwidth, I/O resources) from a substrate node with the aim of isolating and allocating part of these resources for a Virtual Network;
- Once this allocation is done, a Virtual Node owns part of substrate node physical resources, and for all purposes each Virtual Node can now be seen as (e.g. configuration, management, monitoring, troubleshooting) a dedicated physical node. Also, each substrate node can support one or multiple substrate links, which each one can also carry one or multiple Virtual Links.

- Virtual links:
  - With Link Virtualization it is possible to provide the transport of multiple virtual links from different Virtual Networks over one or more physical links;
  - A virtual link is often identified explicitly by a tag, but can also be identified implicitly, by a time slot or a wavelength, for example. A wide variety of the standard link virtualization techniques available in today's Internet (e.g. ATM, Ethernet 802.1q, MPLS) may in principle be used for this purpose.

2.3.7 Management of Virtual Networks

To allow the establishment of virtual networks [26], it is necessary that the InP has the ability to manage its physical resources, so the InP must always be aware of the resources it has reserved and available at every moment (Discovery) for, if necessary, verifying whether there are available resources to fulfil a virtual network request made by the VNP. At this phase, InP only has to guarantee, if possible, one solution that does not have to be the best one (VNet Admission Control).

Some approaches, although oriented to small-scale networks or research testbeds, were proposed to solve this problem [26][27]. For the mapping of virtual nodes it is

necessary, in the first place, to take into account the existence of nodes, called edge nodes or Points of Presence (PoPs), whose locations are important for economic reasons, being confined to specific areas close to the end users (these restrictions diminish the number of physical nodes available to engage virtual nodes, but normally at least one virtual node should be located in each geographic area where the service is to be deployed). However, for all other virtual nodes, the geographical location becomes an element almost irrelevant, since they are not directly connected to end-users. In the second place, it is necessary to take into account a set of restrictions and also some optimization criteria in order to acquire the best possible choice (for InP) avoiding the waste of resources (Resource Mapping).

Eventually, the state of the physical network may change either due to the establishment or removal of virtual networks, or even due to links or physical nodes failures which may lead to an inefficient use and an uneven distribution of resources. All the information related with these changes must be collected in order to notify if any deviations on the expected network behaviour occur (Monitoring). Thus, it is essential to re-optimize all the allocated resources by all virtual networks on a periodic basis or triggered by a specific event (Re-Optimization).

All these management phases can be observed into Figure 12.

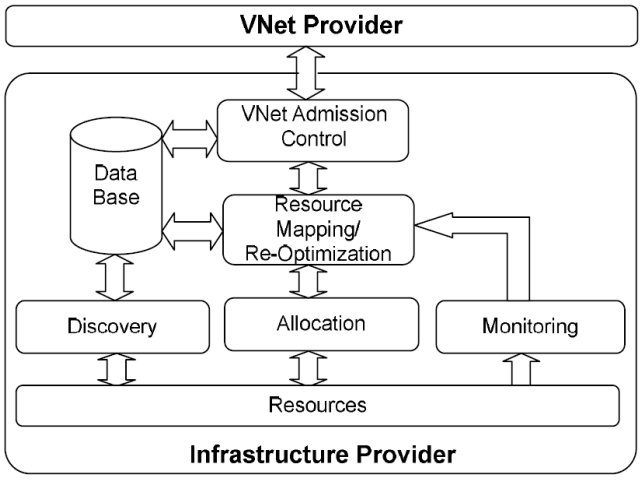


Figure 12: InP Block Diagram [26]

It is imperative to know two things: there may be situations where there is only one candidate InP, while in others there may be multiple and different candidates InP's to provide the required network resources; and the VNP acts as representative of the VNO and thus, through the exchange of messages between themselves, is aware of all requirements and conditions that the VNO desires for its virtual network. Since it is the VNP who applies for resources, and since as explained above, it may exist multiple InP's satisfying the request, it will be required a negotiation between both elements (may be two elements if one InP, or multiple elements if multiple InP's) in an attempt to satisfy both sides without anyone exit hampered. These negotiations are held between the VNP and each of InP's, providing to the first one the choice between using the resources of one of those InP's. However, these negotiations can be divided into two phases.

The Query phase, where the VNP exchanges messages with the InP(s) which include details about the requisites of the virtual network that seeks to establish, such as its Vnet Id, the desired network topology, the bandwidth and delay of virtual links, etc., aiming to know if the InP(s) is capable of meeting all of them. If and only if the InP(s) can meet all requirements (i.e. if it has a solution, whatever it is) (Vnet Admission Control), it will send back to the VNP a positive message and at the same time, through optimization algorithms, look for the best mapping solution among its physical resources to therefore make a reservation for the concerned Vnet (Resource Mapping). This reservation will be cancelled if the InP does not receive a Commit message.

In the Commit phase, where the VNP has received positive messages from various InP's, it will have to choose from one of them and then respond with a Commit in order to allocate definitively all the pre-reserved resources and therefore establish the Vnet. If at this stage no pre-reservation has been performed, the process fails and all the negotiations and processes must be performed again. After all these steps, the VNP informs the VNO if it was possible to establish the required VNet. All these negotiations and processes can be observed in Figure 13.

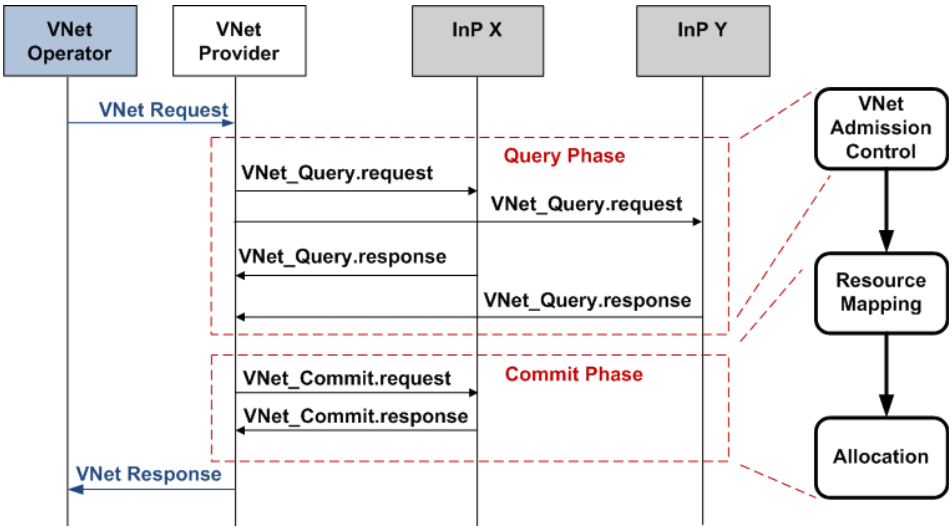


Figure 13: VNet creation sequence chart and flow diagram [26]

All these processes are part of the network management either on the physical level or even on the virtual level, and may be executed every time it is necessary to create a virtual network and / or re-establish a one that had failed completely, needing again to be negotiated.

2.3.8 Virtual Link Failures

When facing any network failure, obviously some Virtual Networks may be affected and may fail too. If a scenario like this occurs, it is required by Vnet users, that those Vnets are able to repair themselves. In order to maintain QoS, Vnets should be able to re-establish themselves as quickly as possible, maintaining all the conditions previously agreed in the first establishment.

Specifically, those network failures could be physical links failures which may lead to Virtual Link failures that, lead in turn to Virtual Networks failure.

This type of scenarios with one or multiple physical link failures were implemented, tested and evaluated with the help of two different perspectives, the Centralized vision and the Distributed vision. These link failures and Virtual Networks self-repairing were the main concerns into the experimental part of this thesis, as shown into chapter 5.

2.4 Summary

In this chapter a brief overview of Telecommunications and Network Management was presented, with emphasis on the characteristics of traditional management that make it inadequate for the future scenarios, where networks are expected to become bigger, more complex and more dynamic. On the other hand, the In-Network Management concepts and mechanisms where also focused in this chapter and it is expected that INM significantly reduces the reaction time of today's centralized management systems, making it especially beneficial in large networks with dynamic network environments. Other important mentioned subject was the network Virtualization. In this chapter, Network Virtualization motivations, challenges, concepts and main elements, as well as its management where also described.

3 Centralized and Distributed Approaches for Network Resources Control

This section describes in detail some of the basic concepts and main ideas considered during the realization of this thesis and the architectures that support them. It presented the basis of the discovery, removal and recovery mechanisms for each approach. For the recovery mechanism, both approaches will be explained in detail.

Section 3.1 and 3.2 present the algorithms and mechanisms researched that were implemented for this thesis for both centralized and distributed paradigms. Finally section 3.4 resumes the entire chapter.

3.1 Link Recovery and Resource Allocation

As mentioned previously, the main goal of this thesis focuses on the support the ability of virtual networks to be capable of self-repairing when affected by link faults on a physical network.

Before starting the actual implementation, it was primarily necessary to understand what algorithms and mechanisms would be required for this purpose, either considering a single view, where all information will be stored in a central body which in turn will take all necessary decisions, either considering a distributed vision where each node will have a partial amount of information and will take cooperative decisions with its neighbours.

Although the mechanisms and algorithms implementation are explained in detail into the next chapter, this chapter will explain on a more theoretical way, the algorithms and mechanisms which were specified and which will be used in the implementation phase.

3.2 Centralized Paradigm

The central organism responsible for network management must have a global knowledge of the network. In order to accomplish this purpose, it was decided that, after starting the learning process, the central element sends to every node an update request and therefore each one of those nodes reply with an update response which must contain all the required information about their characteristics and connections either physical or even virtual, so that it has the global knowledge of the network.

Later, when a physical link failure occurs and after the central element is aware of that, it will have to trigger the discovery mechanism.

3.2.1 Discovery Mechanism

This mechanism should occur on the central element and is responsible for, given a specific physical link, access the database of the central node, which contains the databases of each of the end-nodes of this link where all the required information is stored (information on all resources allocated), in order to discover how many virtual links are established above this physical link, discover the Identification (ID) of each node (virtual nodes in the ends of each virtual link) and finally, discover the ID of each virtual network where they belong to.

3.2.2 Recovery Mechanism

After the discovery mechanism, the central element starts this mechanism which is responsible for, given a particular virtual link that failed and from which all the necessary information is known (discovery mechanism), performing all operations necessary for its recovery.

This mechanism can also be divided in three sub-mechanisms:

The first sub-mechanism is responsible for finding a new alternative path capable of meeting all the resources required for this specific virtual link. In order to accomplish this purpose, it was required a short path algorithm, thus Dijkstra algorithm was the

chosen one. Before the implementation it is important to know deeply the algorithm concept and its mechanisms. Dijkstra Algorithm is one of the most popular algorithms in computer science since it is one of the most important and useful algorithms available for generating (exact) optimal solutions to a large class of shortest path problems. Dijkstra's algorithm is a greedy algorithm and similar to Prim's algorithm (is an algorithm in graph theory [28] that finds a minimum spanning tree for a connected weighted graph.).

It is safe to say that the shortest path problem is one of the most important generic problems in such fields as *Operations Research (*OR*) and Management Science* (MS), computer science (CS) and artificial intelligence (AI). The shortest path problem is a fundamental network optimization problem. Algorithms for this problem have been studied for a long time [29][30][31][32][33][34]. New classes of genuine shortest path problem are becoming very important these days in connection with practical applications of Geographic Information Systems (GIS) such as on line computing of driving directions.

Dijkstra algorithm finds, for a given source vertex (node) in the graph, the path with lowest cost (lower delay in this case) between that vertex and every other vertex. It can also be used for finding costs of shortest paths from a single vertex to a single destination vertex by stopping the algorithm once the shortest path to the destination vertex has been determined.

For an easy understanding of the algorithm used on the searching for an alternative path between two virtual nodes (e.g. virtual nodes A and J) in order to restore the virtual link between these two nodes, an example with 10 nodes will be presented below. In this example, four images will be presented showing the different phases of the algorithm.

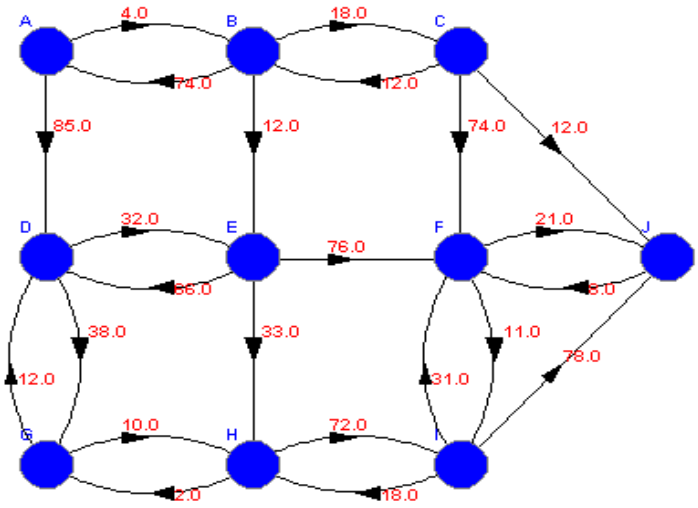


Figure 14: Initial state of the algorithm

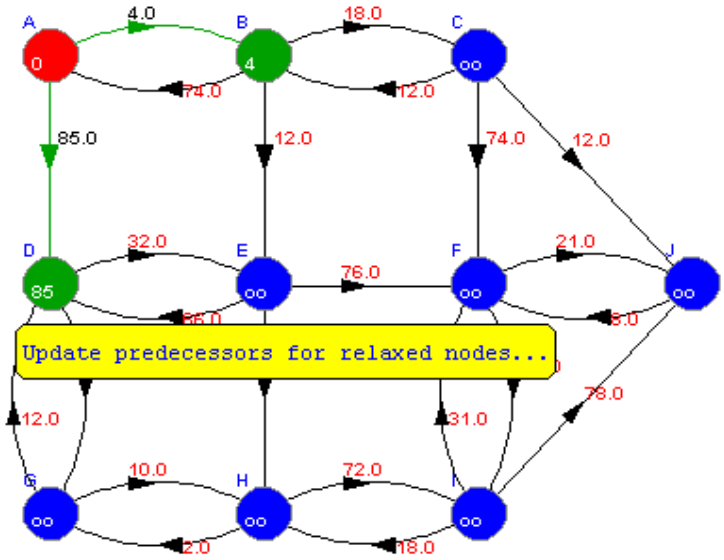


Figure 15: State of the algorithm few iterations after the state represented in Figure 14

This network consists of 10 nodes with physical links between each other with different weights. Initially, before the algorithm starts, all nodes have a blue colour (Figure 14) which represents all nodes that have not been yet covered by the algorithm. In the next iteration (Figure 15), node A (source node) initiates the process, and therefore, all the neighbour nodes will be identified (nodes represented in green) as well as the pass through costs for each one of these nodes (links represented in green). After

carrying out the identification of costs, the next step would be the choice of the link that provides the lowest cost for the source being discarded in all the other links. However, in order to meet the resources required for each virtual link, the algorithm will only be extended to nodes that are connected by physical links that satisfy the required bandwidth for the virtual link in question. That is, if you are searching for an alternative path between the virtual nodes A and B, only the nodes that have physical connections with bandwidth not less than 1Mbit be covered by the algorithm (If the choice is A-C-D-B, links A-C, C-D, D-B are all contain at least 1Mbit of bandwidth).

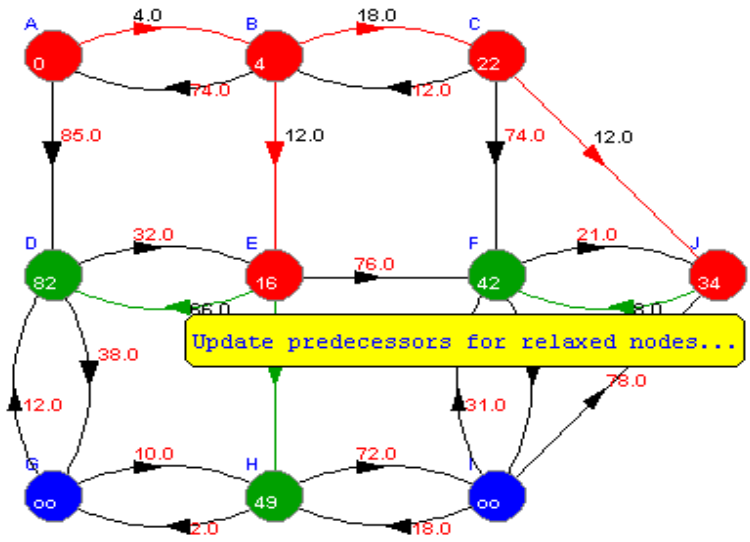


Figure 16: State of the algorithm few iterations after the state represented in Figure 15

Assuming that all these nodes have physical connections which satisfy the bandwidth required for this particular virtual connection, in this stage (Figure 16) of the process (after few iterations), it is possible to see that the previous choice falls on the node B, since it is through this node that the lower cost to the root is obtained. Then, after identified all node B neighbour nodes and corresponding cumulative costs (cost through node E -> 16 = 4.0 +12.0 and cost through node C -> 22 = 4.0 +22), it states that among these two nodes, the one that leads to the lower cost to source node is E.

Therefore, all node E neighbour nodes and corresponding cumulative costs (cost through node D -> 82 = 4.0+12.0+32.0, cost through node H -> 49 = 4.0+12.0+33.0 and cost through node F -> 92 = 4.0+12.0+76) are identified. In this case, it was verified that the best alternative to source node is through node C since it has a cost of 22 to the root

lower value from the values presented by nodes D, H and F (82, 49 and 42 correspondingly).

Therefore, for similarly reasons, the algorithm evolves through node J, thus node F, node E and so on until all nodes are orderly (by the lower cost for node source) covered by this algorithm. In the previous figure can be seen the final state of the algorithm where the only active the links are the ones that guaranteed the best values for each node from the node source.

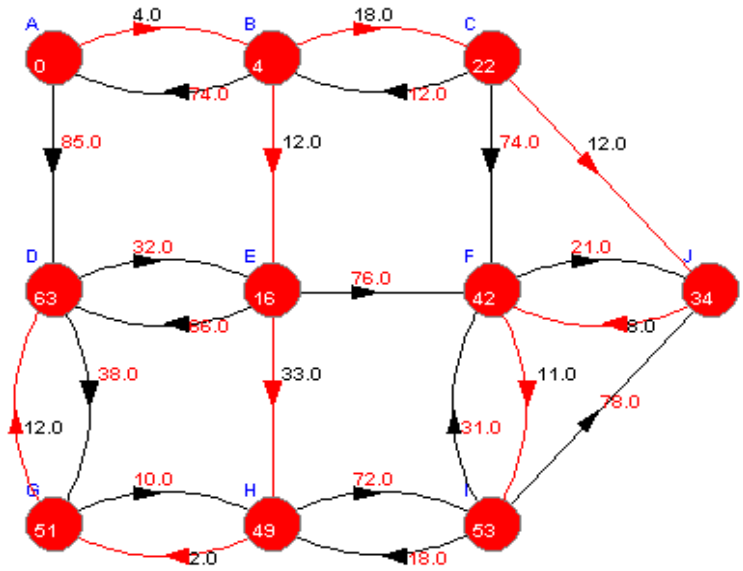


Figure 17: Final state of the algorithm

Through this algorithm, as shown in Figure 17 it is possible to get the best value for each one of the paths, between the source node and all the other nodes (whose are connected through physical links that meet the required bandwidth for the virtual link in question), covered by the algorithm. Finally, the path between nodes A and J is selected among the resulting paths, which in turn is also the option that offers the best possible cost between these two nodes. Therefore it is necessary to evaluate if there is any possibility of restoring the virtual connection between virtual nodes A and J, satisfying the desired maximum delay for this virtual link. Thus, if the cost obtained exceeds the maximum delay desired for this specific virtual link, it means that there is no possibility of recovery. On the other hand if the cost obtained is equal or lower, this means the virtual link is able of being restored by using this route.

The search algorithm must be repeated whenever is necessary to restore another virtual link. The subsequent iterations of the algorithm will already cover the changes (changes in allocated and free resources) caused by previous virtual links re-establishments.

The second sub-mechanism is responsible for the allocation and de-allocation of resources through all nodes and physical links of the network that form the alternative path. In order to accomplish this purpose, it is required the execution of two phases:

- First phase should include the removal of all the remains of the virtual link through the old path. This implies the de-allocation of all the resources previously reserved all along the old path for the virtual link in cause. On the other hand should also include the allocation of resources through all nodes and physical links which form the alternative path into the central element database (refresh central database information).
- Therefore, the second phase should include the communication between the central element and all the nodes affected by these changes in the route of the virtual link in cause, in order to update their databases (performing the new allocation of resources into the network).

This second phase requires an exchange of messages between the central element and each affected node, either for the allocation of new resources to re-establish a virtual link, or to release them on a previous path. To serve this purpose was necessary to create a new type of messages in order to update each node database whenever it was necessary. This type of messages, in the case of a resource allocation, should contain information about:

- Identification of the destination node;
- Information related to the new(s) virtual link(s) such as:
    - Virtual Network identification;
    - Bandwidth required in one or more physical links between this node and other neighbour nodes in order to re-establish the virtual link.

In the case of a resource de-allocation, the messages should contain information about:

- Identification of the destination node;

- Information related to the virtual link(s) that will be removed, such as:
    - Virtual Network identification;
    - Bandwidth allocated in one or more physical links between this node and other neighbour nodes where the virtual link was established;
    - If the receiver node is a virtual node of the virtual link in question, this message must also contain information about the resources allocated into the physical node, regarding the virtual machine settled on this physical node (e.g. allocated memory, and so on), in order to allow their removal.

3.3 Distributed Paradigm

Algorithms based on this paradigm are designed to run concurrently and independently, containing only a limited amount of information. Even if different network elements fail or work at different speeds, it is supposed that these algorithms work correctly. The design of this kind of algorithms can be an extremely difficult task due the complicated settings in which they are running.

Later, when a physical link failure occurs and after both nodes in the ends of this physical link are aware of that, the discovery mechanism has to be triggered.

3.3.1 Discovery Mechanism

This mechanism should occur in each end node and is responsible for, given a specific physical link, perform the necessary operations in order to discover how many virtual links were established above this physical link, discover the ID of each node (virtual nodes in the ends of each virtual link) and finally, discover the ID of each virtual network they belong. All this information is acquired through the research into each end node database.

3.3.2 Removal Mechanism

When a link failure takes place, before the recovery mechanism is performed, since this is a distributed paradigm, firstly it will be required the de-allocation of all resources that were allocated in every node and link all along the actual path of this particular virtual link.

In Figure 18 we can see a similar situation to better understand the removal mechanism. Thus, by assuming that there was a failure on physical link 4-7, it is required the removal of all the resources allocated in the passing nodes 4, 7, and 10 as well as the resources allocated in nodes with virtual machines, 2 and 12. Only after the end of this process, it will be possible the start of the recovery process.

Figure 18: Present the removal of all the virtual link resources when facing a physical link failure

To perform this mechanism it is necessary that the nodes of the network are able to exchange messages with some of its neighbours. Thus, using once again the Figure 18 as an example, it appears that nodes 4 and 7 after consulting its databases (in order to find out, for which neighbour or neighbours should send the messages), need to send messages to Nodes 2 and 10 respectively in order to continue the mechanism. These messages (called “vnode_search_msg” since they are responsible for find the virtual node of a specific virtual link and remove the information regarding the virtual link from all nodes and links through which they pass) should contain information about:

- Identification of the destination node;
- Information related to the virtual link that is being removed, such as:
    - Virtual Network identification;
    - Bandwidth allocated in every physical links where the virtual link was established;
    - Maximum delay allowed for the sum of delays of all the physical connections that make up the virtual connection. This information is

important, since the node receiver of this message can be the virtual node responsible for running the link recovery mechanism.

3.3.3 Recovery Mechanism

After the previously mechanisms, in one of the end nodes, the recovery mechanism starts, which is responsible for, given a particular virtual link that failed and from which all the necessary information is known (discovery mechanism), performing all the necessary operations for its recovery, satisfying all requirements (bandwidth and delay). Since each virtual node will have information about the spatial location of the opposite node in the end of the virtual link, it was decided that only the leftmost node in the virtual link would start the searching algorithm (responsible for the searching for an alternative path).

The distributed shortest path problem and its variations have been studied because of their many applications. A decentralized algorithm for finding shortest paths in a network was presented by Abraham and Rhodes [34]. A distributed algorithm for finding shortest distances in an undirected graph was presented by Ravichandran, et. al. [35] in which at the termination of the algorithm, each node contains the shortest path between itself and all other nodes. The algorithm described by Chandry and Mista [36] finds shortest path from a node i to node j in a directed graph.

So, based in these distributed concepts and mainly based on Dijkstra algorithm but at this time oriented for a distributed perspective, it was necessary to develop a new algorithm with the goal of discovering a path between two nodes, capable of satisfying bandwidth and delay requirements.

To easily explain the design of the distributed search algorithm, an example will be provided. Through a mechanism of flooding which has to be developed to support research, it will be possible for each node, the sending of messages for all its neighbours. These messages should contain information about:

- Identification of the destination node;
- The time when message was sent (Important for the calculation of the transport delay, for each table entry);

- Information related to the virtual link that is being recovered, such as:
  - Virtual Network identification;
  - Bandwidth allocated in every physical links where the virtual link was established (Important, for the node to know through which links the message should be sent);

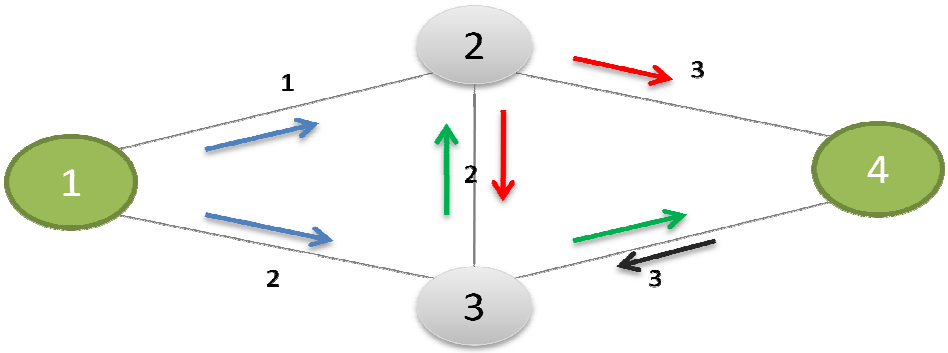


Figure 19: Example that shows the messages exchanged between nodes, during a searching for a new path.

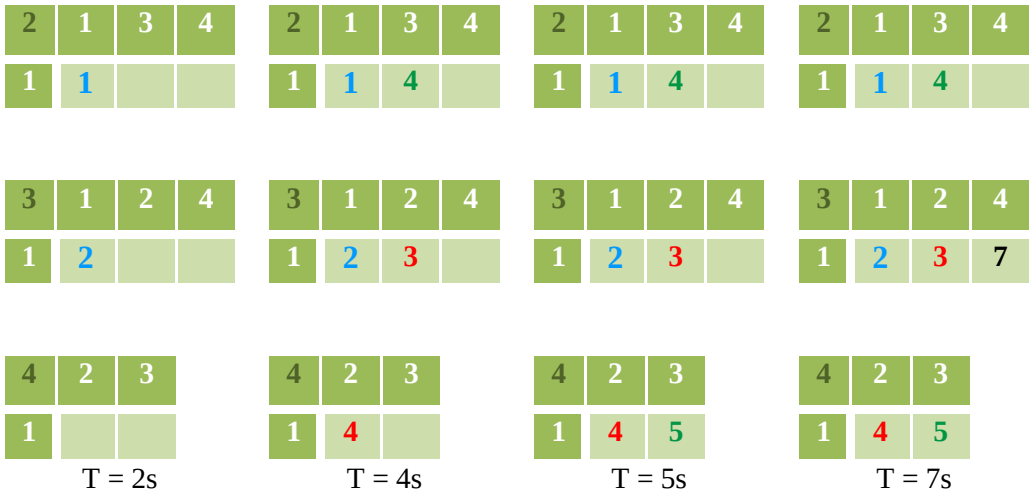


Figure 20: Auxiliary tables for the search algorithm

Initially, node 1 sends messages to its neighbours (Figure 19), whose connections satisfy the required bandwidth (in this case we assume that all connections meet the required bandwidth).

After a second, the message from node 1 reaches node 2, creating a new entry in its table with a cost of 1 (the costs added in each table entry are always related with the virtual node that started the process; in this case node 1). Immediately after the

reception, node 2 sends messages for all its neighbours (nodes 3 and 4 only), except for the neighbour from which it has already received a message (node 1).

After two seconds (Figure 20 for T=2s), the message from node 1 reaches node 3, creating a new entry in its table with a cost of 2. Once more, immediately after this reception, node 3 sends messages for all its neighbours (nodes 2 and 4 only), except for the neighbour from which it has already received a message (node 1).

Then, after three seconds, the message from node 2 reaches node 3, creating a new entry in its table with a cost of 3. In this case no message will be sent for two reasons: the new cost is lower than others already contained in the table; this node only floods once.

After four seconds (Figure 20 for T=4s), messages from nodes 2 and 3 reach nodes 4 and 2 respectively, create new entries in their tables with equal costs of 4. Although in both nodes tables are created new entries, for reasons similar to those listed above, only the node 4 will send messages to its neighbours (in this case only node 3, since it has not received any message from this node).

After five seconds (Figure 20 for T=5s), the message from node 3 reaches node 4, creating a new entry in its table with a cost of 5. Once more, no message will be sent.

Finally after seven seconds (Figure 20 for T=7s), the message from node 4 reaches node 3, creating a new entry in its table with a cost of 7. Once more, no message will be sent.

After the finishing of this process, in order to select the best possible path, it is only necessary to go through all tables (from node 4 to 1), following the links with lower cost. The chosen path would then be formed by nodes 4 – 2 – 1 with a total cost of 4 (delay). So, node 4 should send back a message for node 1, through this path, if it is possible to establish the virtual link on it. During this course, the required resources will be allocated in each link (refreshing nodes databases) through which the message passes (bandwidth required by the virtual link). This message should contain information about:

- Identification of the destination node;
- Nodes and Links that form the path;

- Delay obtained at the end of the research process (will become the new delay of the virtual link);
- Information related to the virtual link that is being established, such as:
    - Virtual Network identification;
    - Bandwidth allocated in every physical links where the virtual link was established;

All the details of the implementation of this algorithm, as well as results (tables, and other information) obtained by the simulations in this thesis will be presented in the following chapter.

3.4 Summary

This chapter described the main ideas and concepts that formed the basis of both implemented algorithms (centralized and distributed). It also included the basis for the discovery mechanism, responsible for detecting the link failure. The removal mechanism is responsible for de-allocating the previously allocated resources. Finally it was presented the recovery mechanism of both approaches, responsible of repairing the virtual links.

4 Implementation on Network Simulator 2

This chapter describes the implementation of the algorithms and mechanisms in NS2, which results will be shown in chapter 5. This chapter begins with an overview of network simulation concepts, highlighting the importance of simulation as a technique for the evaluation of the referred mechanisms. It also provides a high level description of the NS2 simulator.

Even though virtualization is not a new concept, it is still a new research branch on NS2 and, in the beginning of this work, it lacked significant mechanisms that could be used as base for evaluating this concept. For this reason, almost all the mechanisms used in this thesis, to simulate network virtualization, had to be designed and implemented from the beginning.

Section 4.1 begins with an overview of the simulator and its main principles as well as its limitations. Section 4.2 presents a detailed explanation of all mechanisms and modules developed and implemented both for the centralized and for the distributed perspective, allowing a better comprehension of the entire simulator implementation. Finally, Section 4.3 resumes the chapter.

4.1 Network Simulation in Research

To face the costs and complexity of large scale deployment of networks in real-world experiments, network simulation is often used to validate the analysis. Simulations can provide an effective method to validate newly developed protocols and their performance. Other advantages of simulations are the flexibility of implementing different scenarios, multiple error cases and reproducing simulation results in contrast to real-world experiments. In simulations however, there are no interactions with the real environment.

4.1.1 Network Simulator 2 Description

NS [37] [38] is an object oriented discrete event simulator developed with the aim of supporting and unifying research in data networks and telecommunications. It

includes an extended collection of applications, protocols, network types and elements and traffic models. The current version now provides substantial support for simulation of TCP, routing, and multicast protocols over wired and wireless (local and satellite) networks.

NS-2, implemented until 1989 for Virtual Inter Network (VINT) project [39], begun as a variant of the REAL network simulator [40]. Although there is available a new version of NS (NS-3) [41], NS-2 was the chosen platform for the implementation due to its maturity and broader supported mechanisms. NS-2 is well-suited for packet switched networks and is used mostly for small scale simulations of queuing algorithms and transport protocol congestion control [42].

NS-2 incorporates a modular programming model using two-languages rather than a single programming language that usually defines a monolithic simulation. Actually two class hierarchies exist, C++ for the implementation of the core elements and Tcl (Tool Command Language), an interpreted language used for scripting configuration and to control the simulation.

Low-level events and tasks such as processing packets and forwarding them through a router require high performance and efficiency, and therefore they are implemented in a mature compiled language as C++. The compiled hierarchy, C++ is fast to run, but slow to modify; on the other hand, the interpreted scripting language Object Tcl is flexible and very dynamic. There are interactive ways to define network topologies, dynamic configuration of protocol objects and specification and placement of traffic, etc. C++ implements the simulation kernel and the core parts of high performance primitives, and OTcl scripting language expresses the definition, configuration and control of the simulation. OTcl glues the network components implemented in C++.

4.2 Implementation details

This section presents the required changes and additions to the basic structure of NS2 for, towards the simulation, making possible the implementation of scenarios based on Networks Virtualization, and the implementation of the proposed mechanisms.

The first step in this process was the definition and creation of a database that could store all the required information in all the substrate nodes, as described by the schematic in Figure 21;

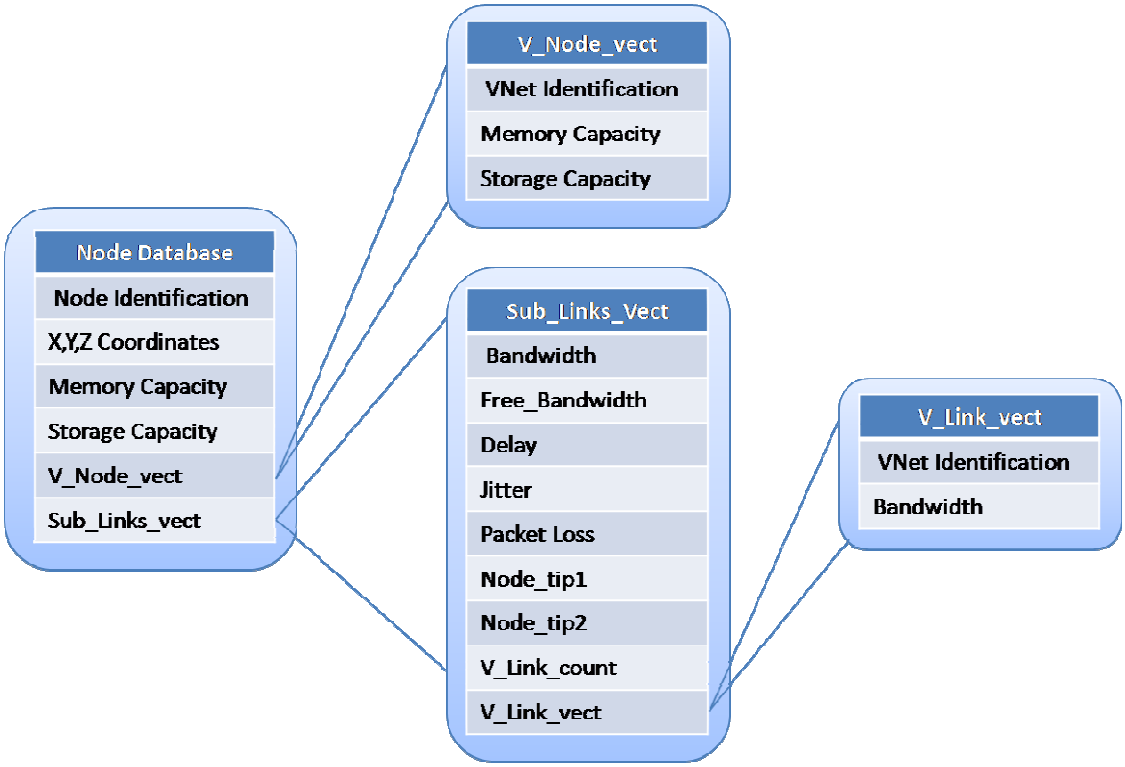


Figure 21: Substrate node database schematic

All the substrate nodes inside the network have in their databases some basic fields and pointers to different structures (Figure 21) as the ones presented:

- Node Database
    - This object is the database of each substrate node.
    - It contains the node identification, coordinates, memory capacity and storage capacity, all important to distinguish the nodes, spatially locate them and define maximum limits for their innate resources. Besides these, other fields will be presented below which were specially defined and crucial to a successful virtualization.
- V_Node_Vect:

- This is a vector of objects that represent the virtual nodes within a substrate node. It is reachable from a pointer in the substrate node, v_node_vect.
- Each time a virtual node is created inside a subtract node, it is added to this vector and some of the physical resources, from the memory and storage capacity fields, are allocated to the virtual node. Whenever an element is added to this vector, its identification is stored in VNet Identification. All this allows the simulation to relate substrate nodes, virtual nodes and their shared resources in order for the simulation to be as close as possible to reality.

- Sub_Links_vect:
  - This is a vector of objects that represent the substrate sub-links.
  - Initially, this vector is empty on all nodes database, but every time a link is created between two nodes, by the Tcl script, the vector is associated with new information: Bandwidth, Free_Bandwith, Link_Delay, Identification of Nodes. This information is in the database of both the end nodes. Furthermore, V_Link_count is responsible for counting the number of Virtual Links in a specific substrate link, while V_Link_vect contains the information of the virtual links.

- V_Link_vect:
  - Represents the virtual links created inside each substrate link.
  - Each time a virtual link is created inside a substrate link, the physical resources, mainly bandwidth, are allocated to it and the element is added to the vector with the VNet identification and the expected QoS characteristics from this V_link.

On the other hand, for simulations regarding the centralized approach, was also necessary to create a central database, capable of storing all network nodes databases (all the network information). So, at Figure 22 is presented an example of the implemented central node database, with two substrate nodes databases (nodes 3 and 4) stored in it.

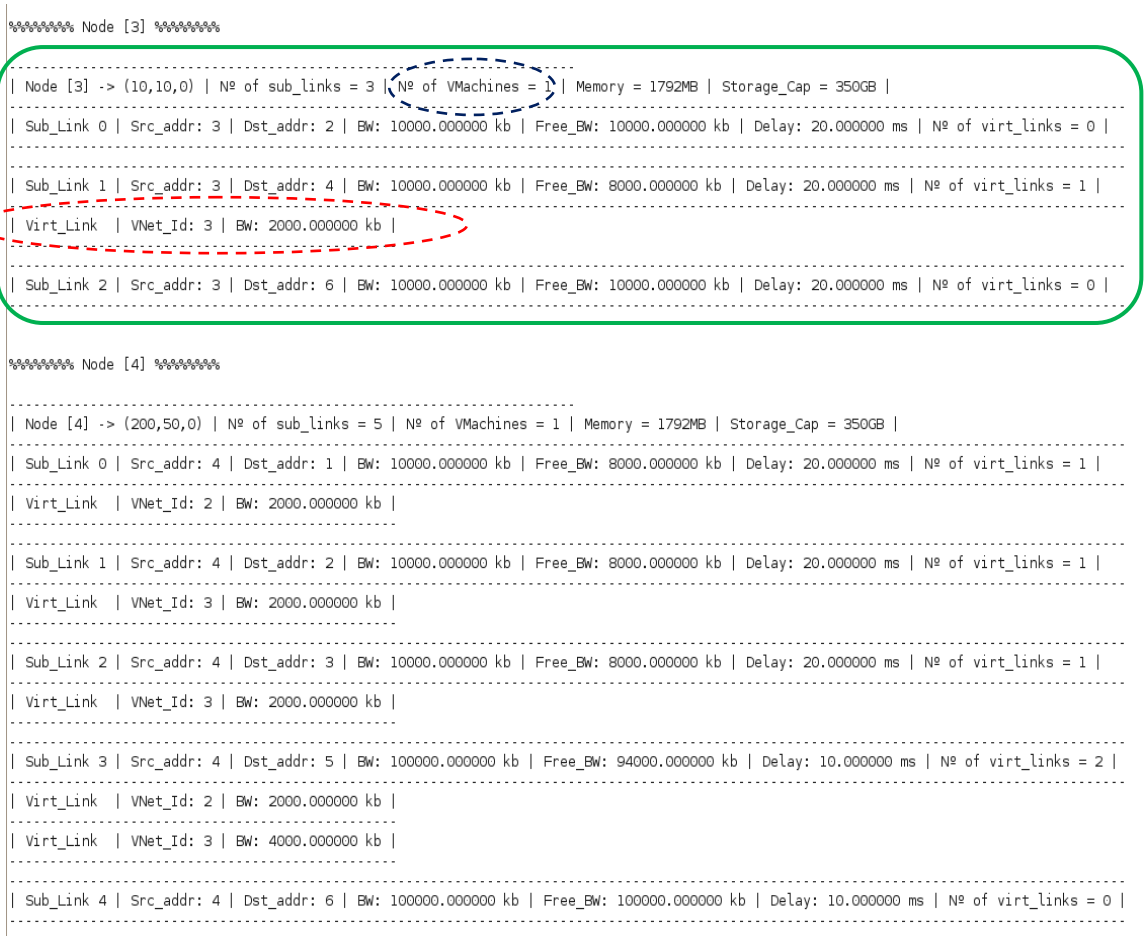


Figure 22: Example of the implemented central node database, with two substrate nodes databases (nodes 3 and 4) stored in it.

From the database of substrate node 3 (green rounded rectangle) it is possible to notice that this node has one virtual machine established on it (blue dashed oval) and a virtual link with 2000kb of bandwidth established on the physical link with edge nodes, the substrate nodes 3 and 4 (red dashed oval). On node 4 databases there is information regarding one virtual machine and several other virtual links from different virtual networks. In this database there is also a situation, where two virtual links (one is from virtual network 2, with 2000 kb of bandwidth and another, but from virtual network 3 with 4000 kb) are established on the physical link with edge nodes, the substrate nodes 4 and 5.

.

4.2.1 Example Scenario

To better explain the details of the implementation, we'll be considering an example scenario represented in Figure 23.

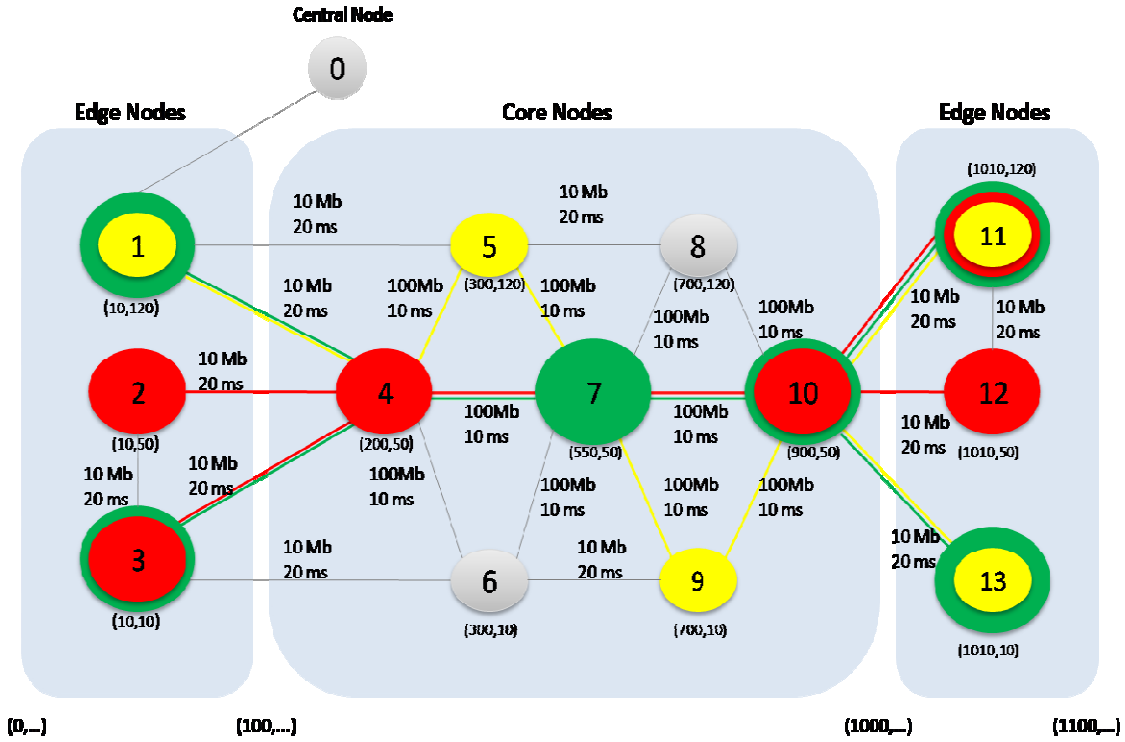


Figure 23: Predefined scenario with 3 different Virtual Networks

As shown in Figure 23 and previously referred, two levels are considered in the network. The first one is the underlying substrate network infrastructure, which includes 13 substrate nodes (represented by a circle with a number inside) and respective substrate links. There is also a node with id 0, outside the network, which represents the centralized element responsible for collecting all the information of the network (only used in the centralized approach) and for running the management algorithms. Plus, we can distinguish the edge nodes (left and right sides) and core nodes (central side). The spatial coordinates for each node are shown below these and have a key role in the distributed approach as will be seen later.

The other level is the virtualized level, also shown in Figure 23. Three colors, green, yellow and red, help to distinguish the three different virtual networks,

respectively 1, 2 and 3. All the nodes and links represented in grey aren't part of any virtual network.

Now, virtual Network 1, in green, will be used as an example to explain this scenario. Virtual Network 1 is formed by:

- Six virtual nodes integrated in substrate nodes 1, 3, 7, 10, 11 and 13 represented by green circles around respective node numbers;
- Five virtual links:
    - **VLink (1-7):** This virtual link is established between virtual nodes 1 and 7 (virtual machines from virtual network 1 established on physical nodes 1 and 7) and is formed by physical links 1 - 4, 4 – 7. In this case, node 4 acts as a passing node, since although it assists in directing traffic of virtual network 1, it has no virtual machine, from virtual network 1 (green), established on itself.
    - **VLink (3-7):** This virtual link is established between virtual nodes 3 and 7 and is formed by physical links 3 - 4, 4 – 7. Once again, node 4 acts as a passing node, since although it assists in directing traffic of virtual network 1, it has no virtual machine, from virtual network 1 (green), established on itself.
    - **VLink (7-10):** This virtual link is established between virtual nodes 7 and 10 (virtual machines from virtual network 1 established on physical nodes 7 and 10) and is only formed by physical link 7-10.
    - **VLink (10-11):** This virtual link is established between virtual nodes 10 and 11 and is only formed by physical link 10 - 11.
    - **VLink (10-13):** This virtual link is established between virtual nodes 10 and 13 and is only formed by physical link 10 – 13.

The same scenario was used for the distributed approach, with the difference that the central node (0) was removed since the objective is to bring all the intelligence and decisions back to the network elements themselves. This scenario served as a starting point for all others created in this dissertation.

4.2.2 Additional Information stored into central node, regarding virtual networks.

Using a more complex scenario (different from the scenario of Figure 23) with 48 physical nodes and several virtual networks (scenario resulting from thesis final simulations), will be presented the type of additional information concerning VNets that the central element must store, so it can perform management operations regarding virtual networks (Figure 24). This figure only focus the information related to virtual networks 1 and 2, however there may be more.

```
-|--------
[ VNet 1 ]
-------------------------------------------------------------------------------
Virtual Nodes are:
[2 (src)] [16 (core)] [6 (dst)] [25 (core)] [9 (dst)] [19 (core)]
-------------------------------------------------------------------------------
Virtual Links are:
[(2) - 20 19 17 - (16) (bw = 1000.000000kb & delay = 11.343968ms)]
[(6) - 12 16 33 - (25) (bw = 1000.000000kb & delay = 11.776426ms)]
[(9) - 14 31 10 - (25) (bw = 1000.000000kb & delay = 11.840292ms)]
[(16) - 17 - (19) (bw = 2000.000000kb & delay = 11.790793ms)]
[(19) - 30 10 - (25) (bw = 2000.000000kb & delay = 11.849712ms)]

-------------------------------------------------------------------------------



----------
[ VNet 2 ]
-------------------------------------------------------------------------------
Virtual Nodes are:
[1 (src)] [20 (core)] [2 (src)] [9 (dst)] [23 (core)] [8 (dst)] [21 (core)]
-------------------------------------------------------------------------------
Virtual Links are:
[(2) - - (20) (bw = 1000.000000kb & delay = 11.186656ms)]
[(8) - 30 7 33 - (23) (bw = 1000.000000kb & delay = 11.413007ms)]
[(20) - 9 18 - (21) (bw = 2000.000000kb & delay = 11.409028ms)]
[(1) - 32 15 29 17 19 - (20) (bw = 1000.000000kb & delay = 11.156454ms)]
[(9) - 18 21 33 - (23) (bw = 1000.000000kb & delay = 11.721810ms)]
[(21) - 33 - (23) (bw = 2000.000000kb & delay = 11.530629ms)]

-------------------------------------------------------------------------------
```

Figure 24: Additional information stored into central node, necessary to distinguish virtual networks resources.

Virtual network 1 is formed by 6 virtual nodes (2, 16, 6, 25, 9 and 19), each one is established on the physical node with the same ID (virtual node 2 is established on physical node 2), and is also formed by 5 virtual links. As an example we will consider the virtual link (2) – 20 – 19 – 17 - (16) where nodes 2 and 16, with IDs between brackets, are virtual end nodes (have virtual machines from virtual network 1) of this virtual link, and where nodes 20, 19 and 17 are “passing nodes” since although they are responsible for directing (forwarding) traffic of virtual network 1, they have no virtual machine established on them (from virtual network 1). Figure 24 also presents the information regarding virtual network 2.

Having presented all the necessary information to be collected either by a central element or even by the network nodes, will be then presented two different approaches (centralized and distributed approaches) that, by using all the stored information, enable the network management when confronted with failures (in the present thesis, link failures).

4.2.3 Centralized Link Failure Approach

This section presents the details of the implementation of the centralized approach for substrate link failures.

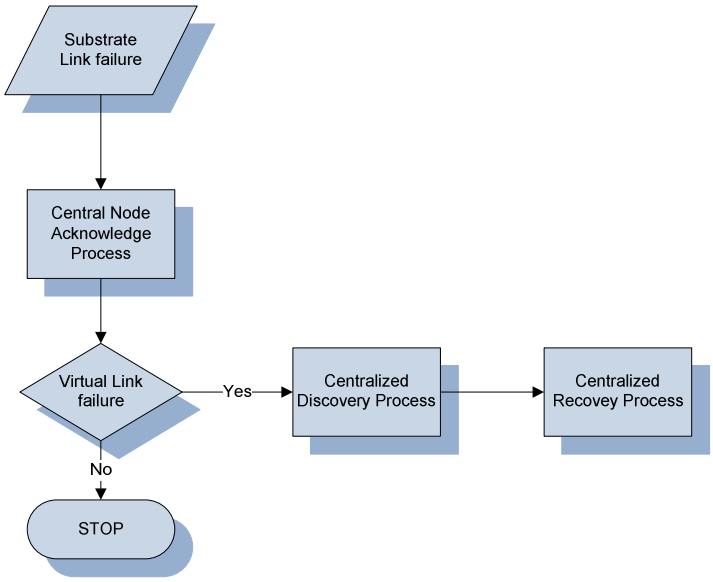


Figure 25: Centralized link failure approach scheme

After a link failure event takes place, information about this is sent to the central node, in order to decide how to handle it. Therefore, as described in the algorithm diagram on Figure 25, one of two situations can occur: the substrate link failure happened on one physical link with no virtual link on it, meaning that no virtual link is affected and virtual link recovery is unneeded (process terminates); or the substrate link failure occurred on one physical link with virtual links established on it causing also their failure. This last situation initiates the Centralized Discovery process and the Centralized Recovery process. These processes will be detailed in the next sub-sections.

Still, it is important to remember that for this thesis simulations, only situations in which one or multiple virtual links are affected by a physical link failure are considered, avoiding simulations in which the recovery algorithm of virtual links would not be necessary.

4.2.3.1 **Central Node Acknowledge process**

Central Node Acknowledge process has an extreme importance on a centralized approach, since this process provides all the information required by every algorithm that runs inside this Central Node. In this process, whenever something occurs, like a substrate link down, the affected nodes (Nodes into substrate link tips) send messages towards the central node in order to provide it with essential information to work.

4.2.3.2 **Centralized Discovery process**

As explained before, this process only occurs if the substrate link which failed has Virtual links running on it. After the central node has gathered all the information available, this process of Discovery is now ready to begin.

As shown in Figure 26, the Discovery process is sequential and responsible for:

- Discovery of Virtual Networks which were affected by the specific physical link failure (returns as much identifications as the number of VNets which have virtual links established in this physical link).
- Therefore, begins another identifying process responsible for the identification of the IDs of all VLinks affected, from each one of those VNets. When a physical link failure occurs, may affect one or multiple Virtual Links that may belong to one or multiple virtual networks;
- After both identifications, this process also provides all the information related with each of those virtual links, essential for the recovering process (IDs of both Virtual Nodes in the ends of the virtual link, Virtual link bandwidth, Virtual link maximum delay).

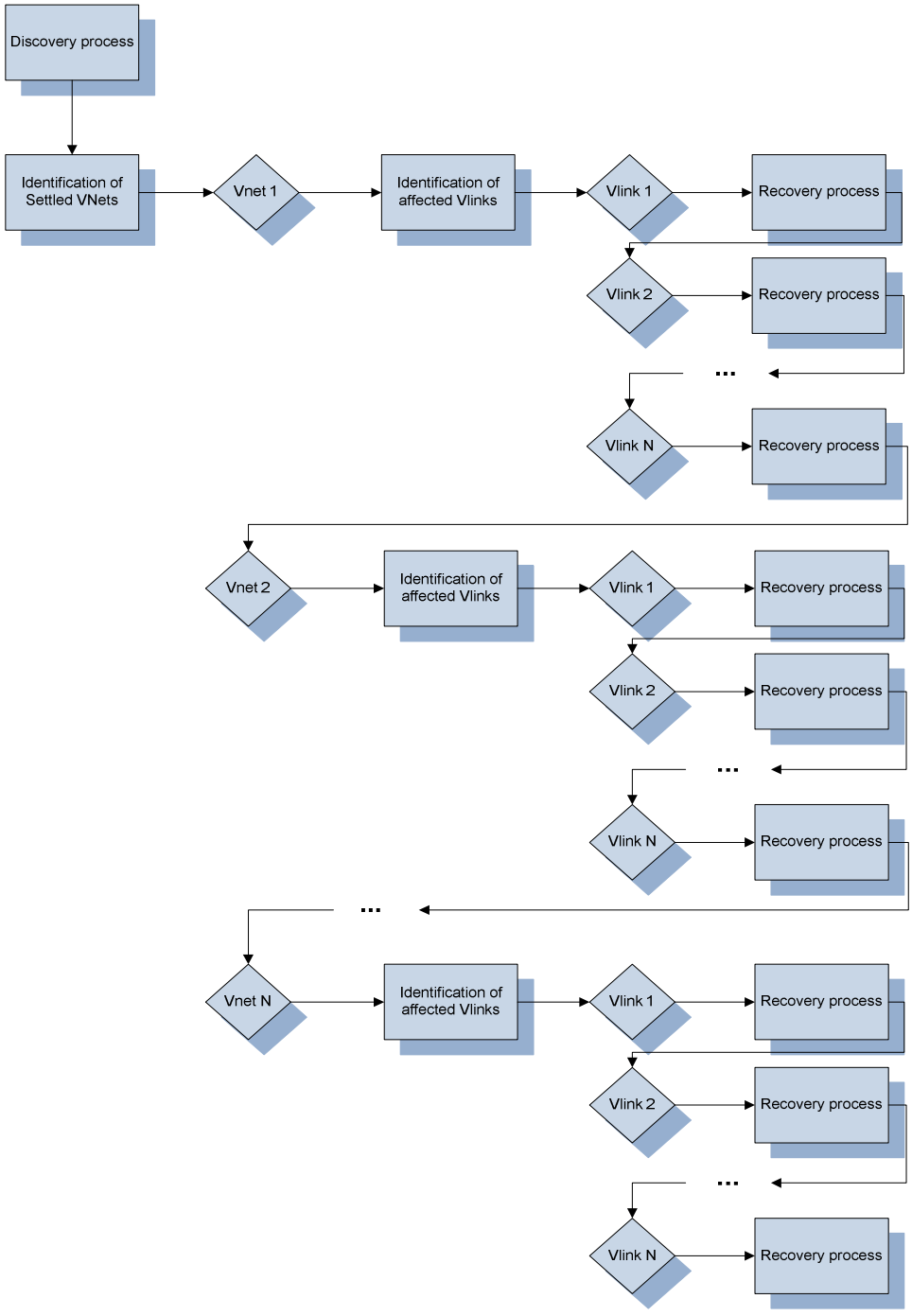


Figure 26: Centralized Discovery process scheme

4.2.3.3 **Centralized Recovery process**

The recovery process is triggered every time a Virtual link goes down. Thus, this process is responsible for recovering all failed Virtual Links and can be divided in three

different phases. The first phase is responsible for removing the information related with the failed virtual link in the central node, and also responsible for storing into auxiliary variables, the necessary information (stores information about all the resources required for this Virtual link, such as the spatial location of Virtual Nodes in the ends of the virtual link), in order to provide content for the following phase to work.

Then, begins the second phase, where all the information obtained from the central node database and from auxiliary variables is filtered and prepared to allow the initiation of the path selection algorithm. For this, a modified version of Dijkstra algorithm was defined and implemented, even if there was already one available on ns2.

The Dijkstra code had to be re-written in order to make possible the interaction of this algorithm with all the information accessible by the central node, to make this algorithm capable of finding paths for failed virtual links that meet all the requirements (bandwidth and maximum delay between the ends of the virtual link). Plus, the algorithm has to be capable of finding a new path for all the affected virtual links, since a Virtual network can only be re-established when all the virtual links can be re-established. If, for some reason, during a virtual link recovery the algorithm does not return a new path with the necessary requirements, means that the virtual link recovery has failed and virtual network recovery has also failed.

After the new paths have been obtained in the central node (Dijkstra was successful), it is necessary to communicate with network nodes in order to inform them about all the changes on routes and allocated resources caused by virtual link recovery. So this phase can be seen as a network refresh phase. Moreover, if one Virtual Network fails to recover indefinitely (happens when one or multiple virtual links were impossible to repair), it is also necessary to refresh all the nodes belonging to it, in order to erase all the information related with this Virtual network and to free allocated resources.

4.2.4 Distributed Link Failure Approach

In opposition to the centralized approach, all the intelligence and decisions are distributed between the network nodes, significantly increasing the complexity of all this network processes.

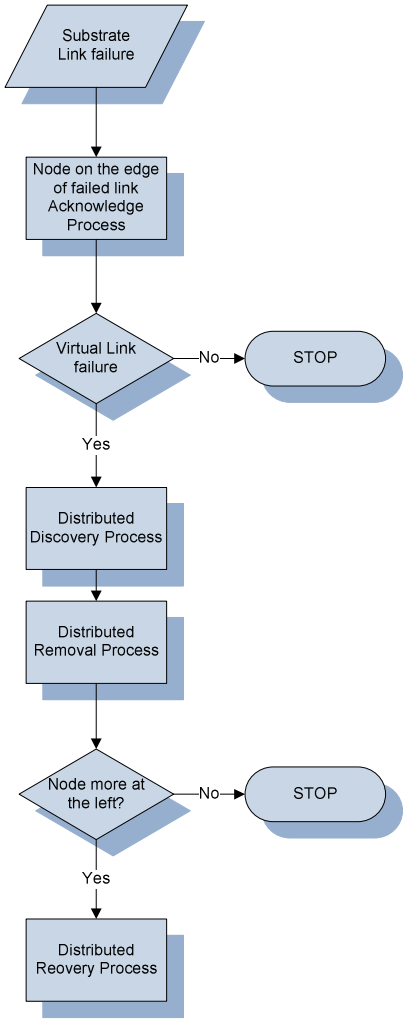


Figure 27: Distributed link failure approach scheme

When a substrate link failure occurs, both the nodes that were connected through this link must be aware of this failure, in order to initiate all the processes presented in Figure 27 (theoretically, all the processes presented in Figure 27 should occur simultaneously on both end nodes, but NS2 only allow scheduled events). Again, one of two different situations can occur: a substrate link failure on a link with no virtual link established on it, or a substrate link failure on a link with one or several virtual links established on it. While the first situation does not initiate any process and terminates at this stage, the last situation initiates the Distributed Discovery process plus the

Distributed Removal and Distributed Recovery processes (Figure 27). These three processes will be explained in the next sub-sections.

For this thesis, only the situations, in which one or multiple virtual links were affected by a physical link failure, have been considered. This avoids occurring simulations in which the recovery algorithm of virtual links is not necessary.

4.2.4.1 **Node on the edge of failed link Acknowledge process**

Due to the nature of the distributed perspective, both the nodes on the edges of the failed substrate link are responsible for gathering information about the failure, thus required to initiate the distributed Discovery process.

4.2.4.2 **Distributed Discovery process**

This process is similar to the Centralized Discovery process. The significant important difference is that, instead of occurring into the central node, now it occurs on both nodes in the edges of the failed links: every time a link failure takes place, this process occurs simultaneous in both nodes (theoretically, since in reality NS2 does not allow simultaneous events, only scheduled events).

4.2.4.3 **Distributed Removal process**

For this process, a new agent was created as well as different types of messages in order to enable the communication between all the substrate nodes of the network. The distributed removal process occurs for each virtual link affected and begins with each edge node (of the failed physical link) verifying if it itself is a virtual node for the virtual link in cause (Figure 28).

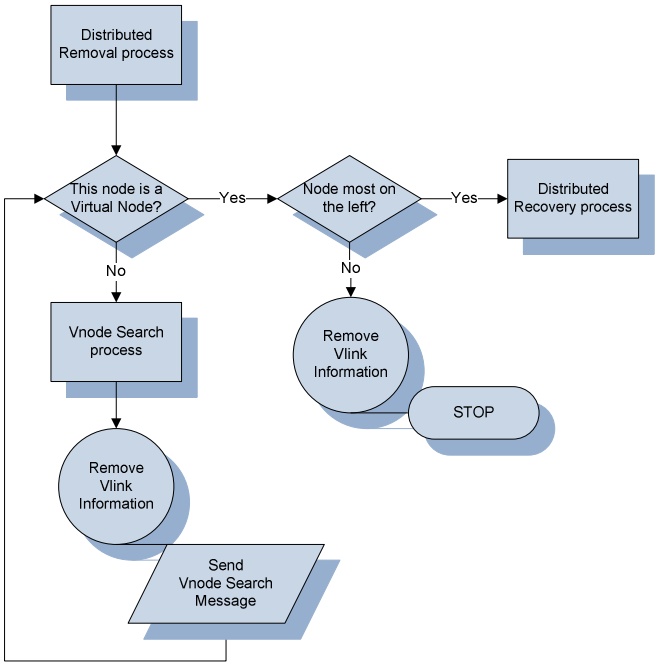


Figure 28: Distributed removal process scheme

As presented in Figure 28, if the result was "YES" (i.e. it is a virtual node of the virtual network in cause) and:

- If this node is also the virtual node most on the left (since all this processes occur simultaneous on both virtual nodes on the edges of the Virtual link, this condition ensures that only one of them, initiates the recovery process) it is responsible for gathering all the information related with this Virtual link and starts the Distributed Recovery process (section 4.2.4.4).
- If this node is not the virtual node most on the left, the information related with this virtual link is removed, and the process terminates here.

Instead, if the result was "NO" (i.e. it is not a virtual node of the virtual network in cause) this node is responsible for sending a message to the near available node of this Virtual link aiming to find a virtual node.

After receive the message, receiver node has to verify if it itself is a virtual node for the virtual link in cause, repeating all the process again (Figure 28).

4.2.4.4 **Distributed Recovery process**

This process is initiated in each virtual node that at the same time is the Vnode most left of the failed virtual link. So, this process is responsible for finding alternative paths for each failed Virtual link, taking into account the information stored about each one of them, such as: the ID of the Virtual Nodes in the edges of the virtual link in cause; maximum delay between them; and bandwidth required in every physical link which form the virtual link.

Then, begins the searching for a new path aiming to reestablish the Virtual link. In order to make this process possible, several sub-problems had to be solved, such as:

1. The communication of a specific node with all its neighbours nodes, therefore the communication of these nodes with their neighbours in order to reach the destination node;
2. Ensure that all the Virtual link pre-established conditions are met (Conditions such as bandwidth and delay, agreed on Virtual Network first establishment, have to be respected).

In order to answer and solve these problems, it was necessary the development of a flooding algorithm as a basis for exchanging messages between neighbours, and in simultaneous, it was also necessary the development of a distributed algorithm capable of ensuring Virtual Links pre-established conditions, otherwise, Virtual network conditions should be renegotiated (this last solution was not explored on the simulations of this thesis).

4.2.4.4.1 Flooding Mechanism

The flooding algorithm was developed and implemented to operate in two phases using the information related with the Virtual link (flooding algorithm used in this Thesis is a base and a non-optimized step to perform a first analysis of the distributed approach).

In order to simplify the explanation, the node responsible for the initiation of this searching process will be denoted as initiator node. The initiator node will have to search for a path to other specific node in the network which will be denoted as destination node.

In the first phase, in order to reduce the number of sent messages, the initiator node knows the spatial location of the destination node (the initiator and destination nodes, since they both have a Virtual Node and belong to the same Virtual Network, have information about the spatial location of each other) and also knows the spatial location of its neighbours (information stored into node databases). Thus, the initiator node only sends messages for neighbour nodes which are into the destination node direction. Therefore, the node which receive this message also initiate a similar process, sending messages only for neighbour nodes which are into the destination node direction and so on, until the destination node is reached (see example of Figure 29).

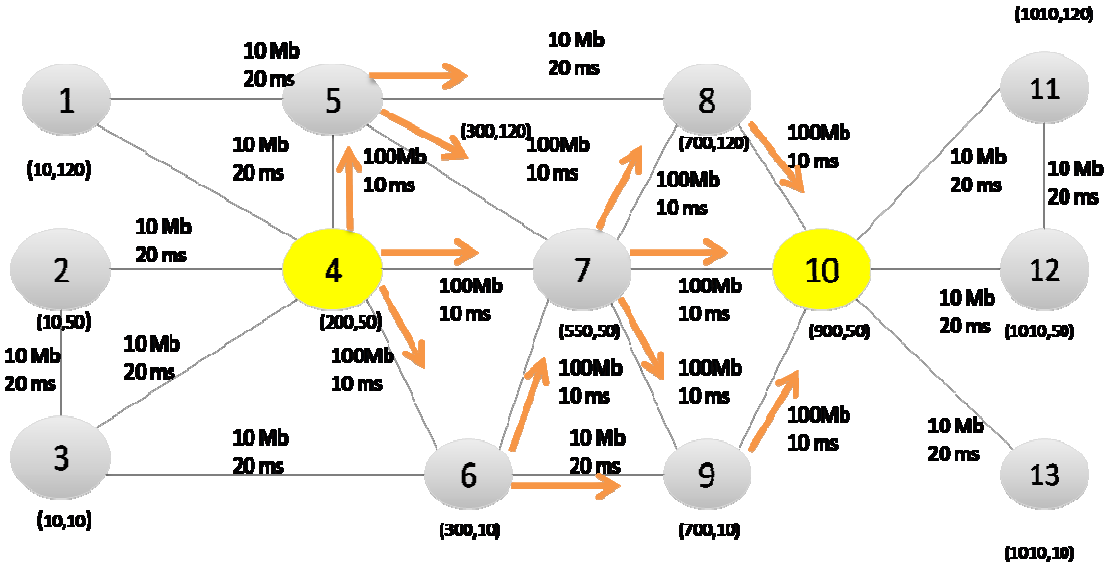


Figure 29: Example of flooding with spatial restrictions example

In Figure 29, assuming that the node initiator 4 is located at X= 200 and destination node is located at X=900, this means that the initiator Vnode will only send messages for neighbour nodes located at X>=200. All the receiver neighbour nodes do the same thing (once the message they receive, contains destination node coordinates), until the message reaches node 10, like shown in Figure 29.

If the destination node is not reached in this process, a second phase begins, where the initiator node sends messages for all its neighbours without any spatial restrictions (usual flooding).

For both phases presented above it is important to notice that, maximum end-to-end delay and bandwidth were the only parameters taken into account. Thus, for both

phases, in order to simplify the algorithm (reducing also the number of exchanged messages) and by knowing that the quicker path available between two nodes is usually the one from where the flooding message arrives firstly, all nodes only flood once for each Virtual link recovery process. In both approaches, all nodes only flood messages to neighbour nodes connected by physical links which respect the Bandwidth needed (bandwidth previously established whether the creation of the virtual link). With this approach, it is possible to guarantee that the chosen path, if there is one, meets the bandwidth requirements.

4.2.4.4.2 Mechanism to take in account the delay of the path

In parallel with this flooding mechanism, it was also implemented another mechanism in order to account the delays for different paths. The developed mechanism was based on Dijkstra algorithm but oriented for a distributed perspective.

To more easily understand the operation of the mechanism implemented, it will be presented with the help of a specific example (recovery of the virtual link 12-17; initiator node 12 and destination node 17). This mechanism starts when the initiator node (node with the value contained inside of the blue dashed oval in Figure 30) starts flooding. After that, in every receiver node it is created a table like the one shown in Figure 30.

Node 8 receives one of the messages sent by node 12 (initiator node), into this node it will be created a table with all its neighbours (in the case of normal flooding), or the neighbours which are into the direction of the destination node (in the case of flooding with spatial restrictions). As can be seen in Figure 30, all neighbour nodes of node 8 (IDs of these nodes are represented into the green oval) have the value of 9999 (=infinite) on the value of the delay with the exception of the value for node with ID 12. This means that node 8 has received a message from node 12 and the cost/delay from node 8 to this node is 0.260855.

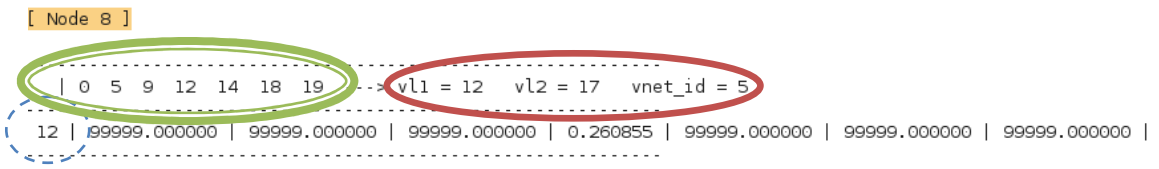
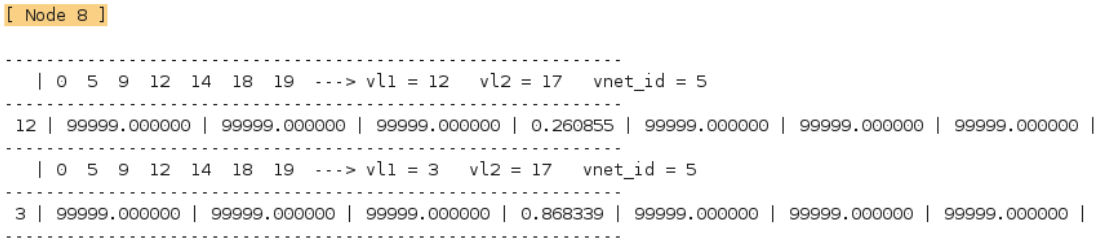


Figure 30: Example of the first entry on node 8 table

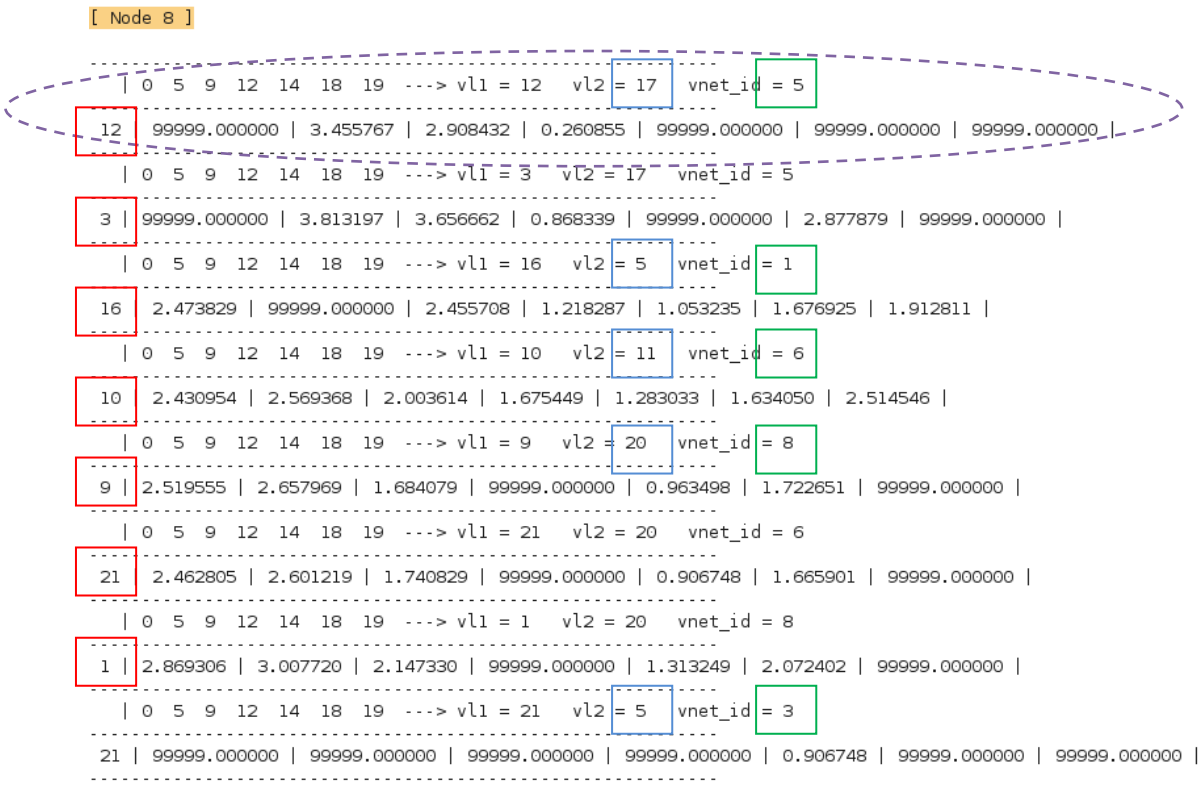
In Figure 32 in the nodes 8 table, it is possible to see in the two first lines (inside dashed purple oval), that this node has received more messages from other nodes such as node 5 and node 9. Now, it is possible to state that there are two other possible paths for node 12, the first one is through node 5 with the cost/delay of 3.455767, and the second one is through node 9 with the cost/delay of 2.908432.



```
[ Node 8 ]

-----------------------------------------------------------
   | 0  5  9  12  14  18  19  ---> vl1 = 12   vl2 = 17   vnet_id = 5
-----------------------------------------------------------
 12 | 99999.000000 | 99999.000000 | 99999.000000 | 0.260855 | 99999.000000 | 99999.000000 | 99999.000000 |
-----------------------------------------------------------
   | 0  5  9  12  14  18  19  ---> vl1 = 3   vl2 = 17   vnet_id = 5
-----------------------------------------------------------
 3 | 99999.000000 | 99999.000000 | 99999.000000 | 0.868339 | 99999.000000 | 99999.000000 | 99999.000000 |
-----------------------------------------------------------
```

Figure 31: Example of the second entry on node 8 table



```
[ Node 8 ]

-----------------------------------------------------------
   | 0  5  9  12  14  18  19  ---> vl1 = 12   vl2 = 17   vnet_id = 5
-----------------------------------------------------------
 12 | 99999.000000 | 3.455767 | 2.908432 | 0.260855 | 99999.000000 | 99999.000000 | 99999.000000 |
-----------------------------------------------------------
   | 0  5  9  12  14  18  19  ---> vl1 = 3   vl2 = 17   vnet_id = 5
-----------------------------------------------------------
 3 | 99999.000000 | 3.813197 | 3.656662 | 0.868339 | 99999.000000 | 2.877879 | 99999.000000 |
-----------------------------------------------------------
   | 0  5  9  12  14  18  19  ---> vl1 = 16   vl2 = 5   vnet_id = 1
-----------------------------------------------------------
 16 | 2.473829 | 99999.000000 | 2.455708 | 1.218287 | 1.053235 | 1.676925 | 1.912811 |
-----------------------------------------------------------
   | 0  5  9  12  14  18  19  ---> vl1 = 10   vl2 = 11   vnet_id = 6
-----------------------------------------------------------
 10 | 2.430954 | 2.569368 | 2.003614 | 1.675449 | 1.283033 | 1.634050 | 2.514546 |
-----------------------------------------------------------
   | 0  5  9  12  14  18  19  ---> vl1 = 9   vl2 = 20   vnet_id = 8
-----------------------------------------------------------
 9 | 2.519555 | 2.657969 | 1.684079 | 99999.000000 | 0.963498 | 1.722651 | 99999.000000 |
-----------------------------------------------------------
   | 0  5  9  12  14  18  19  ---> vl1 = 21   vl2 = 20   vnet_id = 6
-----------------------------------------------------------
 21 | 2.462805 | 2.601219 | 1.740829 | 99999.000000 | 0.906748 | 1.665901 | 99999.000000 |
-----------------------------------------------------------
   | 0  5  9  12  14  18  19  ---> vl1 = 1   vl2 = 20   vnet_id = 8
-----------------------------------------------------------
 1 | 2.869306 | 3.007720 | 2.147330 | 99999.000000 | 1.313249 | 2.072402 | 99999.000000 |
-----------------------------------------------------------
   | 0  5  9  12  14  18  19  ---> vl1 = 21   vl2 = 5   vnet_id = 3
-----------------------------------------------------------
 21 | 99999.000000 | 99999.000000 | 99999.000000 | 99999.000000 | 0.906748 | 99999.000000 | 99999.000000 |
-----------------------------------------------------------
```

Figure 32: Example of the node 8 table after many table entries

By analyzing all the three figures (Figure 30, Figure 31 and Figure 32), it is possible to state the growth of node's 8 table with regard for the number of entries in this table (this is possible because almost simultaneously there may be other virtual links recoveries). So, every time a new path is being searched with a variation in at least one of these parameters:

- Initiator node (red rectangles in Figure 32),
- Destination node (blue rectangles in Figure 32),
- Virtual Network (green rectangles in Figure 32),

A new entry in the table is created.

Proceeding with the recovery process of the virtual link [12-17], via Figure 33 (which only displays the entries, related with virtual link 12 -17, on the tables of nodes 8,14,17,18), it is possible to see part of this recovering. As explained before, node 8 sends messages for all neighbours and among them, two nodes (nodes 14 and 18) are evidenced in Figure 33.

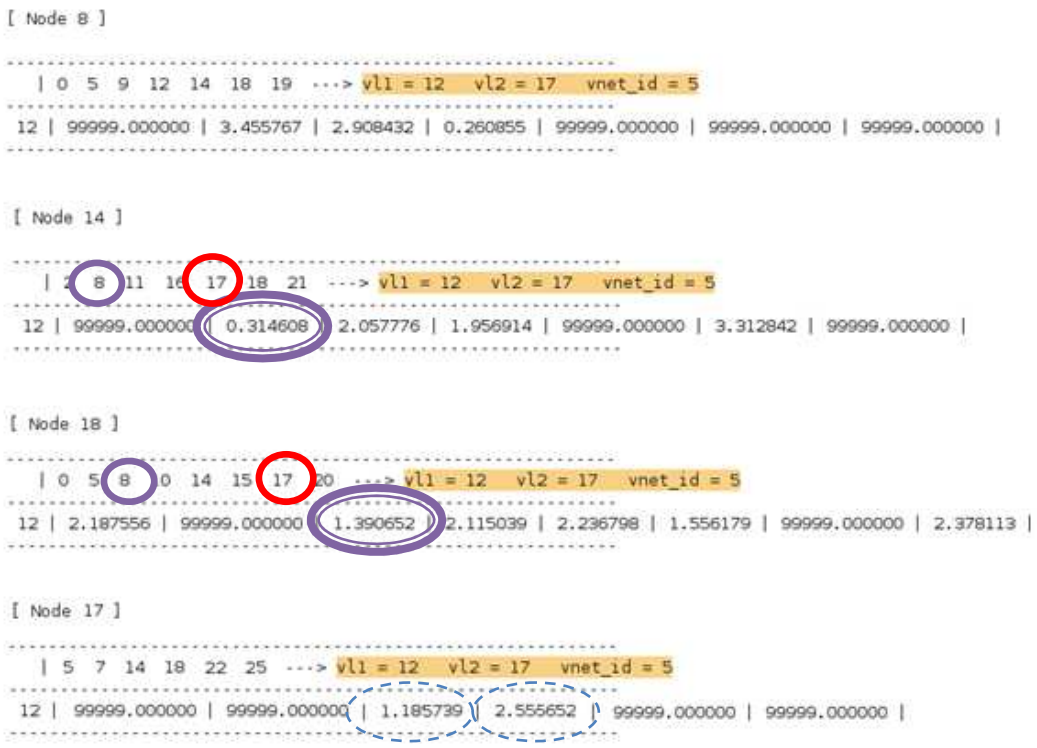


Figure 33: Presents part of the cost tables from nodes 8, 14, 17 and 18

Each one of those two nodes receives a message from node 8 with the cost of 0.314608 and 1.390652 respectively (14 and 18), but they also receive many other messages with different and larger values of cost/delay for node 12 from other nodes. This means that the first message received from node 8 has the lower cost (once, cost = delay).

On the other hand, it is possible to verify that those two nodes are also neighbours of node 17, which is the destination node for virtual link [12-17] recovery. Thus, as stated in Figure 33 it is possible to verify that those nodes (14 and 18) also sent messages to node 17 with (blue dashed oval). Since node 12 only received messages from these two nodes and the best way to reach the source is through node 14, it is possible to conclude that the best path available is 17-14-8-12 and it has the final cost of 1.185739.

After the best path available is found, it is the time to see if its total cost/delay is lower than the delay required for virtual link [12-17] (this delay was agreed upon the formation of the virtual network).

Finally, when the destination node is reached, it sends back a message containing the ID of the initiator node. To do that, cost tables are always consulted, in order to make possible this type of messages to go through the nodes which own lower cost for initiator nodes). Then, during this course (from destination node to initiator node), resources are allocated in each link (refreshing nodes databases) through which the message passes.

To better understand this reply process will be then presented a more detailed explanation.

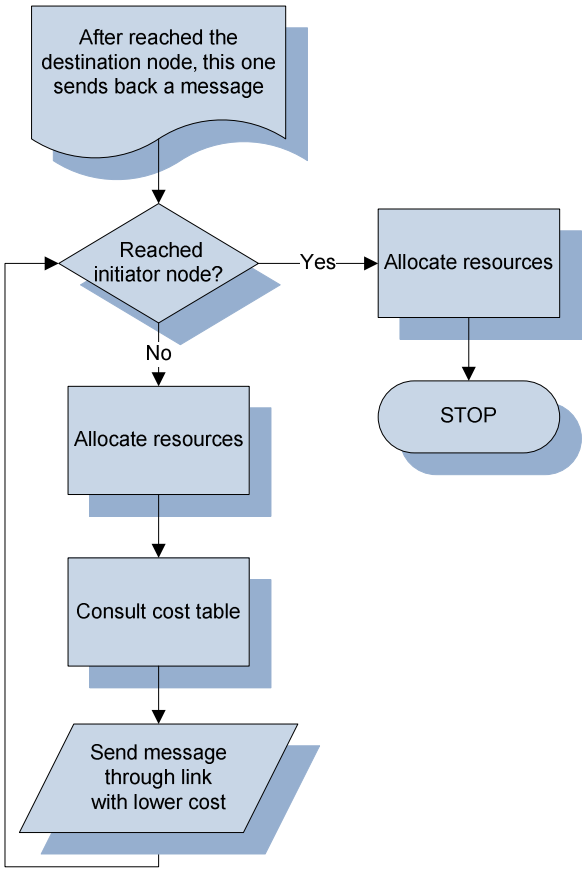


Figure 34: Scheme of messages being sending back in order to allocate resources and re-establish the Virtual link in cause with a new path

Taking as basis the example of Figure 33 and the scheme of Figure 34, node 17 sends a message, back to node 14, since node 14 was the first to reach node 17 with a message). Once this message is received by node 14, its table is consulted and it is verified that the best cost for initiator node 12 is through node 8 with a cost of 0.314608. So, node 14 sends back a message for node 8 and finally this node checks its table and realizes that it is a neighbour of the initiator node and that the best cost for it, is through node 12 itself. Then, a message is sent to the initiator node, finishing the process. As previously mentioned throughout this process, resources will be allocated on the network (for the virtual link in cause).

Otherwise, after passing some time (normally more than sufficient for reestablish a virtual link), this table entry is erased in order to refresh this table.

4.2.4.4.3 Avoiding deadlock

In order to guarantee that a deadlock does not occur in the recovery process (if the initiator node does not receive a message indicating the chosen path, and/or the destination node does not receive a flood message, or the paths do not satisfy the virtual network requirements), after a specified time, the searching for a new path ends, this network is removed, and the allocated resources are released.

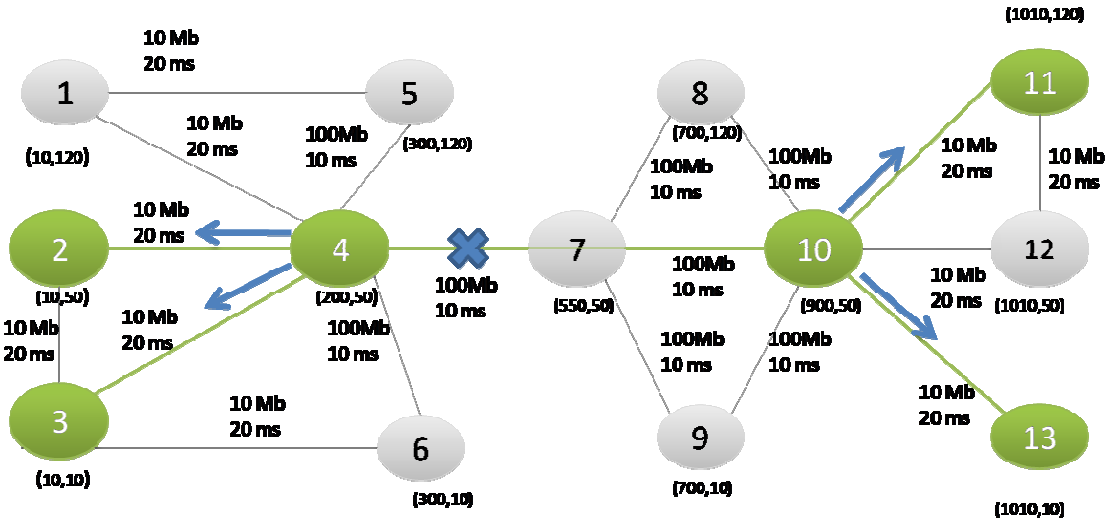


Figure 35: Link failure example without any possible alternative path example

Using Figure 35 as an example, with the failure on the physical link between nodes 4 and 7, it is impossible to repair virtual link 4-10, and consequently, it is impossible to repair the green virtual network.

In this process, physical link 4-7 does not exist and the virtual link 4-10 was removed in the physical link 7-10 in the previous process, and therefore, the completely removal of the virtual network requires the removal of virtual links 4-2, 4-3, 10-11 and 10-13. Thus, assuming that virtual nodes 4 and 10 did not receive any message, all the information related with this virtual network will be removed from them, and messages will be sent for the virtual nodes in opposite edges (node 4 sends messages to nodes 2 and 3; node 10 sends messages to nodes 11 and 13) allowing them to do the same if necessary (if they are also end nodes from another virtual link besides the one in cause). This process can be repeated if there are more virtual links, terminating only when there are not any more virtual links belonging to this specific virtual network

4.3 Summary

This chapter described the NS2 implementation of the architecture. All mechanisms and modules developed and implemented either for the centralized approach and for the distributed approach were presented in detail for a better comprehension of all the developed self-repairing architecture. The centralized approach requires the existence of a central element for which will be targeted all the network information. All the algorithms and mechanisms required for network failures resolution are performed; after solutions are found and decisions are performed on the central element, these are communicated to the network nodes. In contrast, there is the distributed approach which aims at a distributed knowledge across the network that does not require the use of a central element. Thus, the proposed algorithm, as opposed to centralized, runs and requires the contribution of all nodes of the network in order to solve network failures: now, each node only has information about its neighbours and, in the case of being a virtual node (of any virtual link), it has also some information about the virtual node in the opposite edge of the virtual link.

5 Simulation results and evaluation

This chapter presents results of the mechanisms implemented in the simulator. The results present the two different perspectives of network management, centralized and distributed through different scenarios. We include results of each perspective and then a comparison between both, assessing the benefits and advantages of each approach.

This chapter is organized in the following way. Section 5.1 describes the general scenario to be used in the simulations. Then, section 5.2 and section 5.3 present results of simulations of both centralized and distributed approaches, where the number of physical nodes and virtual networks is increasing, respectively. Finally, section 5.4 resumes the most relevant results.

5.1 Scenario overview

This section explains the principles behind the scenarios and the parameters that were assumed during their creation towards the evaluation of the simulation results from the centralized and distributed approaches. In order to obtain results in different situations more easily, it was created a generic scenario with input parameters such as:

- Edge_L
    - This parameter represents the number of physical nodes available in the edge (most on the left) of the network. This zone most on the left is restricted to the spatial coordinates between X > 0 and X < 120 (Figure 36);
- Core
    - This parameter represents the number of physical nodes available in the core of the network. This zone is the center of the network and is restricted to the spatial coordinates between X > 150 and X < 280 (Figure 36);
- Edge_R

- This parameter represents the number of physical nodes available into the edge (most on the right) of the network. This zone most on the right is restricted to the spatial coordinates between X > 300 and X < 500 (Figure 29);

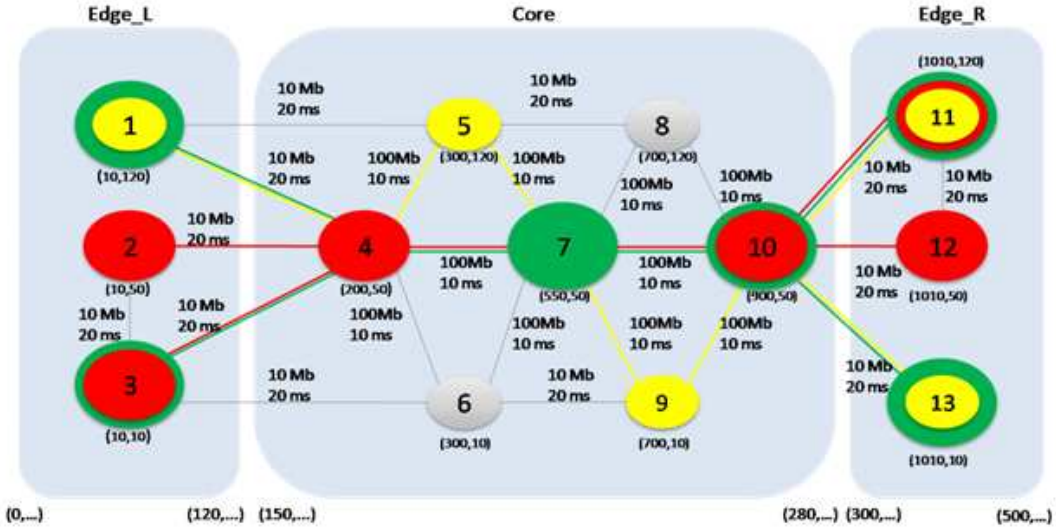


Figure 36: Example with multiple virtual networks established over a physical network.

- VNets
  - This parameter represents the number of virtual networks which will be available over the physical network;
- V_Edge_L
  - This parameter represents the maximum number of virtual nodes available over the physical nodes in the Edge_L zone for each virtual network (for each VNets);
- V_Core
  - This parameter represents the maximum number of virtual nodes available over the physical nodes in the Core zone for each virtual network (for each VNets);
- V_Edge_R
  - This parameter represents the maximum number of virtual nodes available over the physical nodes in the Edge_R zone for each virtual network (for each VNets);

- Links_Down
    - This parameter represents the number of physical links failures, which will occur all long the simulation.

To obtain different results for the same scenario in order to treat them statistically (average values and confidence intervals), NS seed is changed. Thus, in order to get more reliable results, for every scenario with the same input parameters, 10 simulations with different seed values were made. Scenario seed is used to change almost all values of it, such as a change in the number of virtual links from each virtual network, variations on the links selected for failures occur, and so on. However, the topology code implemented automatically generates a scenario based on input parameters and configuration rules, easing the evaluation of different situations.

In both scenarios, "Increasing the node count scenario" and "Increasing the Virtual Networks count scenario" physical links between nodes were configured with a random delay between 0 and 1 second and with a bandwidth between 5 and 10 Mb/sec in the case of links between edge nodes and core nodes, and with a bandwidth between 50 and 100 Mb/sec in the case of links between core nodes.

The following sections present the results obtained. For all these results, the corresponding confidence interval at 95% will also be displayed.

5.2 Increasing the node count scenario

This scenario represents a series of simulations where the number of nodes is increasing. It is tested for 4, 8 and 12 link failures on section 5.2.1 (number of exchanged messages) and tested for 4 and 12 links failures on sections 5.2.2 (virtual link re-establishment success times) and 5.2.3 (virtual link re-establishment success rate). First, it will be shown the number of messages being exchanged in both approaches (centralized and the distributed) while facing multiple physical link failures and physical network size. The initial parameters used in this simulation are shown in Table 2.

Edge_L	Core	Edge_R	VNets	V_Edge_L	V_Core	V_Edge_R	Links_Down
4	14	4	8	2	4	2	4

Table 2: Initial parameters used in simulation

Therefore, in order to make possible the growth of the physical network in number of nodes, the variables Edge_L, Core, Edge_R will be increased. Similarly, in order to make possible an increase on the number of physical links failures in the simulation, the values of Links_Down will also be increased.

The following results refer to the centralized and distributed approaches.

5.2.1 Number of exchanged messages

These results refer to the number of messages (independently of the type of message is being sent - a message on these graphics represents the exchange of a message between two nodes) being exchanged while re-establishing the virtual links affected by failures in physical links.

- Centralized approach

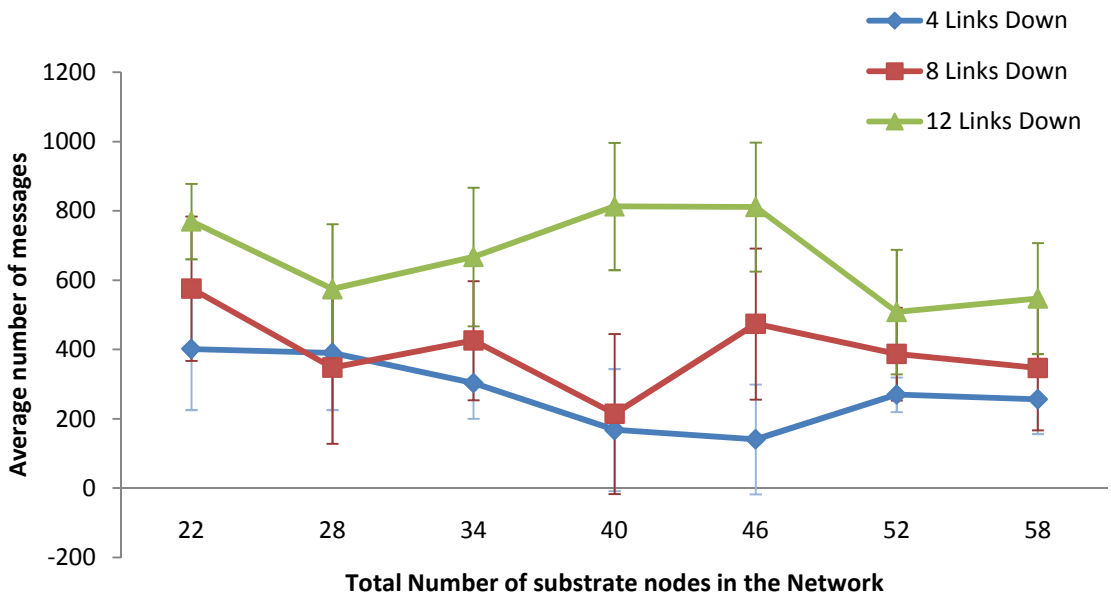


Figure 37: Centralized Average number of exchanged messages as the number of physical nodes into the network grows, tested for 4, 8 and 12 physical links failures

- Distributed approach

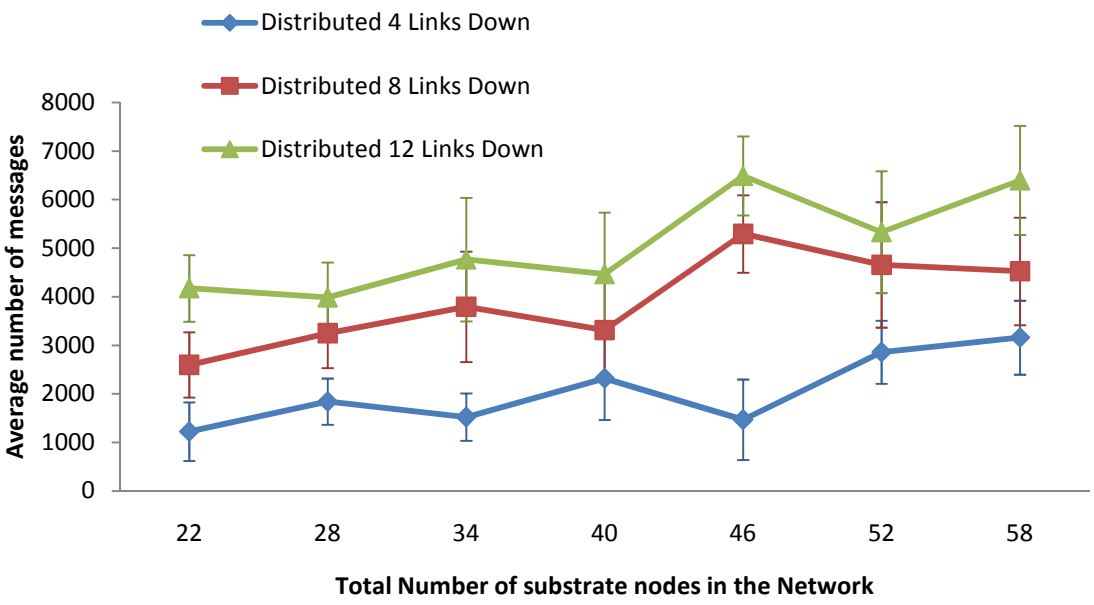


Figure 38: Distributed Average number of exchanged messages as the number of physical nodes into the network grows, tested for 4, 8 and 12 physical links failures

An increase of exchanged messages can be observed in Figure 37, as the number of physical link failures increases, since it triggers an increase of affected virtual links, and consequently an increment of exchanged messages for re-establishment. From Figure 37, it is also possible to verify that there is a slight decrease on the average number of exchanged messages as the network grows, since the impact of the link failures in the overall network is decreased.

In the distributed approach, a slight increase of exchanged messages can also be observed (Figure 38), as the number of physical links down increases. It is also possible to verify a trend on an increase in the average number of exchanged messages as the network grows. An increase in the number of nodes causes an increase on the number of messages exchanged, as it increases the number of nodes to do flooding into the networking (mechanism of flooding occurs in the process of finding a new path).

Comparing the results of centralized and distributed approaches, the average number of messages exchanged during the re-establishment of virtual links is always higher on the distributed one.

This is mainly due to the use of mechanisms of flooding, when searching for alternative paths to re-establish virtual links. Moreover, on the central approach, it is considered that the nodes communicate directly with the central node (and only one message is accounted for between each node and the central point) which is not true in real scenarios.

5.2.2 Virtual link reestablishment success times

This section presents the duration times of re-establishments for centralized approach vs distributed approach while facing multiple physical link failures (4 and 12 physical links down), and the physical network growing in terms of nodes.

SIMULATION WITH 4 LINKS DOWN

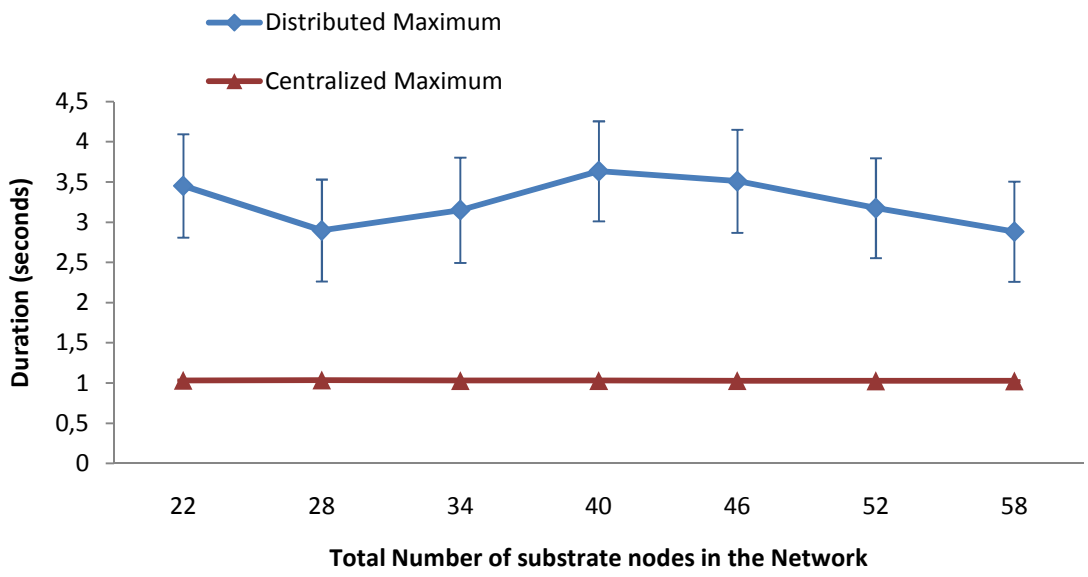


Figure 39: Average durations of re-establishments as the number of physical nodes into the network grows, tested for 4 physical links failures

SIMULATION WITH 12 LINKS DOWN

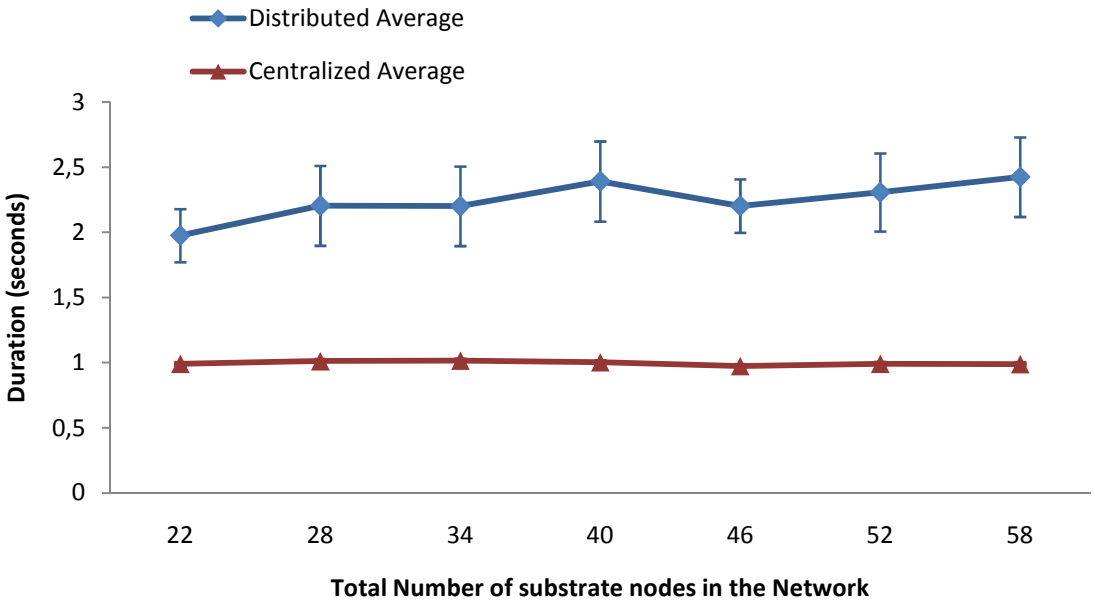


Figure 40: Average durations of re-establishments as the number of physical nodes into the network grows, tested for 12 physical links failures

From the simulations of Figure 39 and Figure 40, it is possible to verify that for both approaches, the number of physical link failures has small impact on the times obtained for the durations of re-establishments. These conclusions can be explained, given the fact that the network does not grow in terms of spatial location, and instead it grows in number of nodes becoming a network more compact with more alternative paths, allowing lower times in communications.

It is shown that the reconfiguration times are always higher for the distributed perspective. This is partly due to the fact that in the distributed algorithm there is a significant communication between the various nodes of the network which in turn implies a large exchange of messages involving a huge accumulation of delays in the trade of these messages. This accumulation of delays appears to be much higher in the distributed approach than in the case of the centralized approach where it is only required to exchange messages to refresh nodes. Notice also that the algorithm

processing time in the central node is not accounted, and it is assumed that only one link is required to get to the central node.

5.2.3 Virtual link reestablishment success rate

This section presents the virtual links reestablishment capability of the centralized approach vs distributed approach. It compares the number of Virtual Links failures, the number of Virtual Links re-establishments and also the number of Virtual networks impossible of being repaired (scenarios with no feasible solution, e.g. no physical path between two nodes) in different simulations while growing the number on nodes in the network for multiple physical link failures (4 and 12 physical link failures).

SIMULATION WITH 4 LINKS DOWN

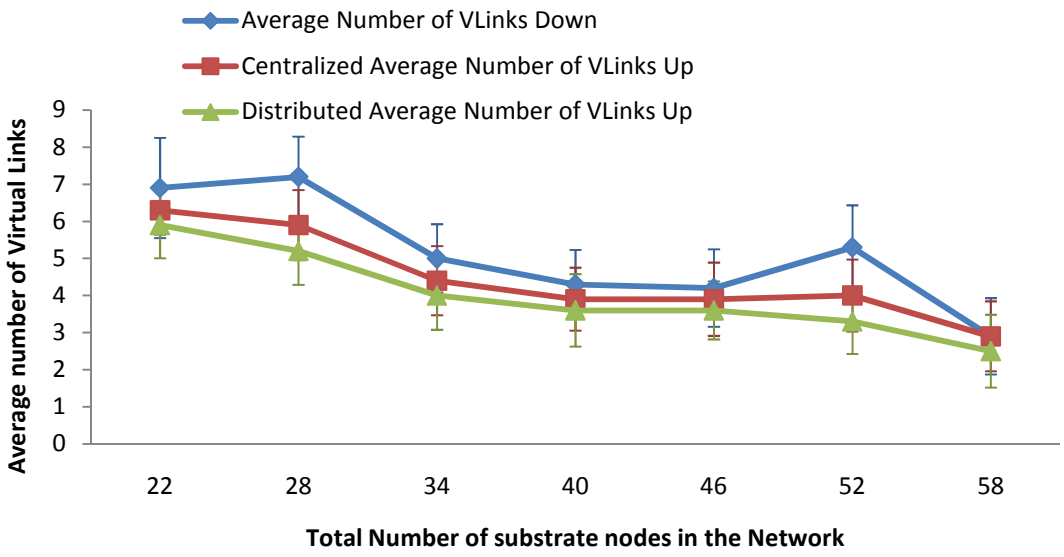


Figure 41: Average number of Virtual Links up achieved for centralized and distributed approaches for the same average number of Virtual Links down as the number of physical nodes into the network grows, tested for 4 physical links failures

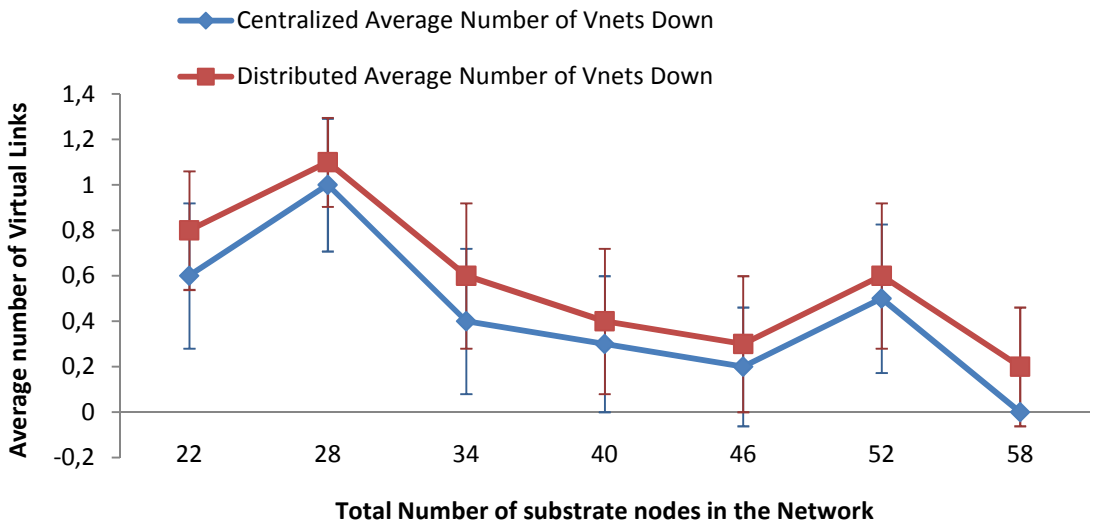


Figure 42: Average number of Virtual Networks impossible of being repaired achieved for centralized and distributed approaches as the number of physical nodes into the network grows, tested for 4 physical links failures

SIMULATION WITH 12 LINKS DOWN

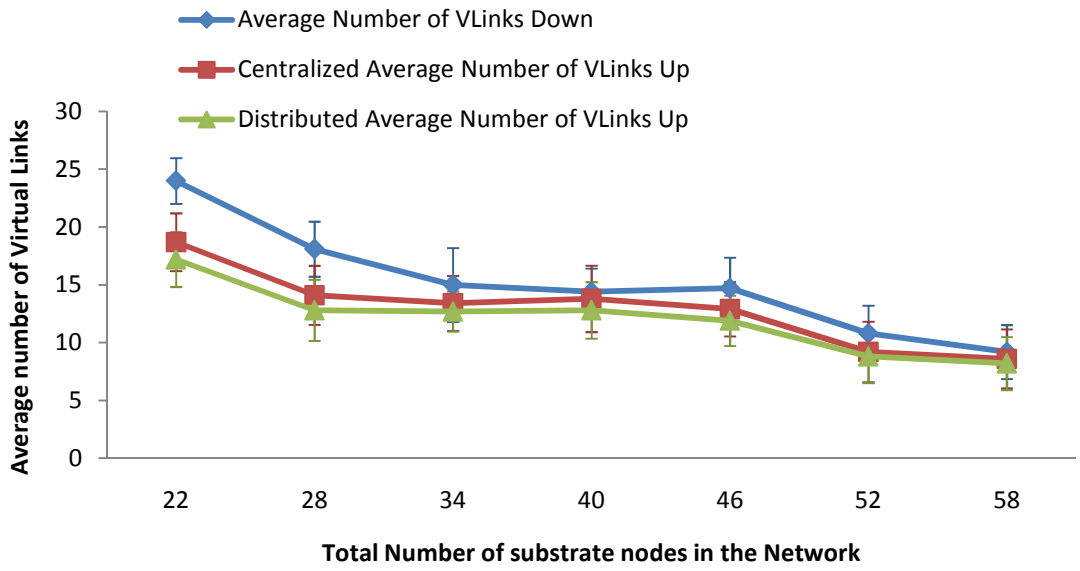


Figure 43: Average number of Virtual Links up achieved for centralized and distributed approaches for the same average number of Virtual Links down as the number of physical nodes into the network grows, tested for 12 physical links failures

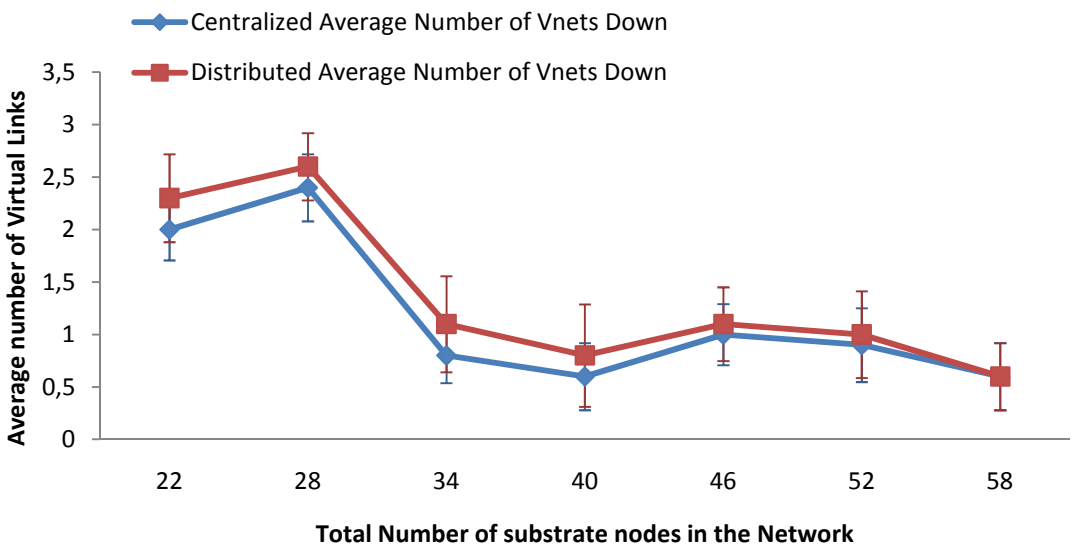


Figure 44: Average number of Virtual Networks impossible of being repaired achieved for centralized and distributed approaches as the number of physical nodes into the network grows, tested for 12 physical links failures

From the simulations of the situations presented by Figure 41 and Figure 43, for both approaches it is possible to verify that the average number of Virtual Links up (re-establishments) achieved for centralized and distributed approaches are growing as the average number of Virtual Links down grows while incrementing on number of substrate link failures. This increase is due to the fact that the growth of physical links failures also increases the number of virtual links that can fail (more physical links failures implies more virtual links failures over those physical links). The number of virtual links re-established also increases with the increase on the number of substrate links failure: since more virtual links will fail, more can be repaired.

The difference between the correspondent average number of link failures and the average number of links up is the average number of virtual links that could not be repaired by each algorithm. The more these two lines are separated, higher is the number of virtual links that could not be repaired. On the other hand, for both approaches, the increase of the number of nodes increases the number of physical links, and consequently decreases the number of virtual links failures and re-establishments. The increase in the number of links while keeping constant the number of virtual

networks and virtual links, implies a decrease on the probability of a physical link failure to provoke a virtual link failure.

The increment on the number of nodes also represents a decrease on the number of virtual links that cannot be re-established, since the lines in Figure 43 are coming closer towards the end of the graph, almost overlapping. The increment on the number of the nodes also increases the number of physical links, which allow more possibilities of paths with the required resources (bandwidth, delay) to re-establish Virtual Links.

In Figure 42 and Figure 44, which represent the number of virtual networks that were impossible to repair, it is possible to observe, for both approaches, a slight increase on the average number of Virtual Networks unable of being repaired with the increment on number of Link failures. It is important to keep in mind that, for a VNet to be impossible to repair, it means that one or more virtual links were unrecoverable. Even if the number of unrepaired VNets is not the same as the number of unrepaired VLinks, these are related and present the same decreasing trend while the number of nodes increase.

In both approaches there is a decrease in the average number of failures and re-establishments of virtual links with the increase of the number of nodes. More nodes means more links, which implies a higher spreading of virtual networks among them. Therefore, the probability of affecting a physical link with many virtual links is lower which causes a reduction on the number of virtual link failures, and consequently, a reduction on the number of virtual links able to be repaired. The number of virtual networks impossible of being repaired is always higher on distributed approaches than in centralized approach. This is due to the fact that we consider that the central element has all information, without incurring scalability concerns.

5.3 Increasing the Virtual Networks count scenario

This scenario represents a series of simulations where the number of Vnets is increasing. It is tested for 4, 8 and 12 link failures on section 5.3.1 and tested only for 4 link failures on sections 5.3.2 and 5.3.3. First, it will show the number of messages being exchanged in both approaches (centralized and the distributed) while facing

multiple physical link failures and physical network growing. The initial parameters used in this simulation are shown in Table 3.

Edge_L	Core	Edge_R	VNets	V_Edge_L	V_Core	V_Edge_R	Links_Down
6	22	6	2	2	4	2	4

Table 3: Initial parameters used in simulation

To provide the increments on the number of Virtual Networks over a same physical network, the variable VNets will be increased. Similarly, to provide an increase on the number of physical links failures, the variable Links_Down will be decreased.

The following results refer to the centralized and distributed approaches.

5.3.1 Number of exchanged messages

The results refer to the number of messages being exchanged while re-establishing the virtual links affected by failures in physical links.

- Centralized approach

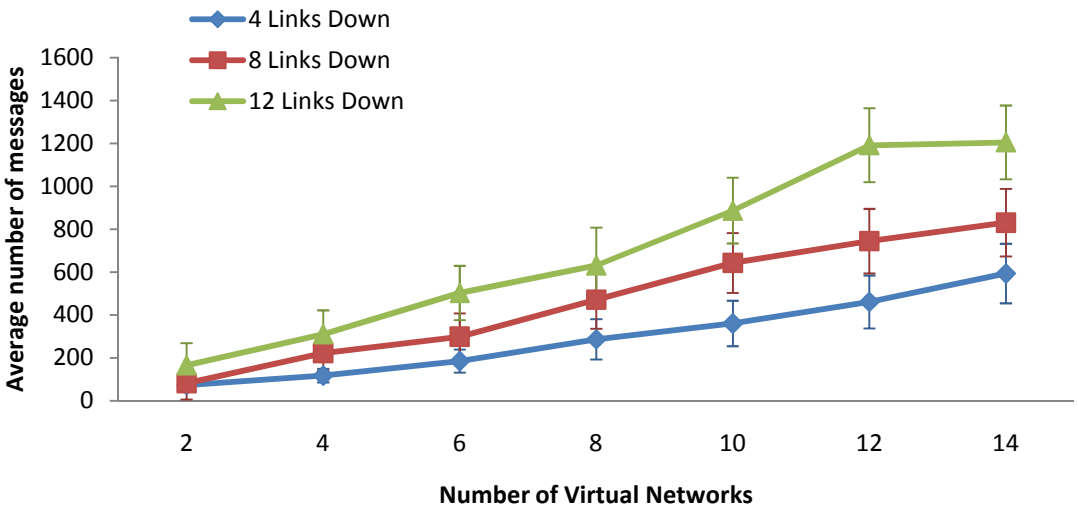


Figure 45: Centralized average number of exchanged messages as the total number of virtual networks into the physical network grows, tested for 4, 8 and 12 physical links failures

- Distributed approach

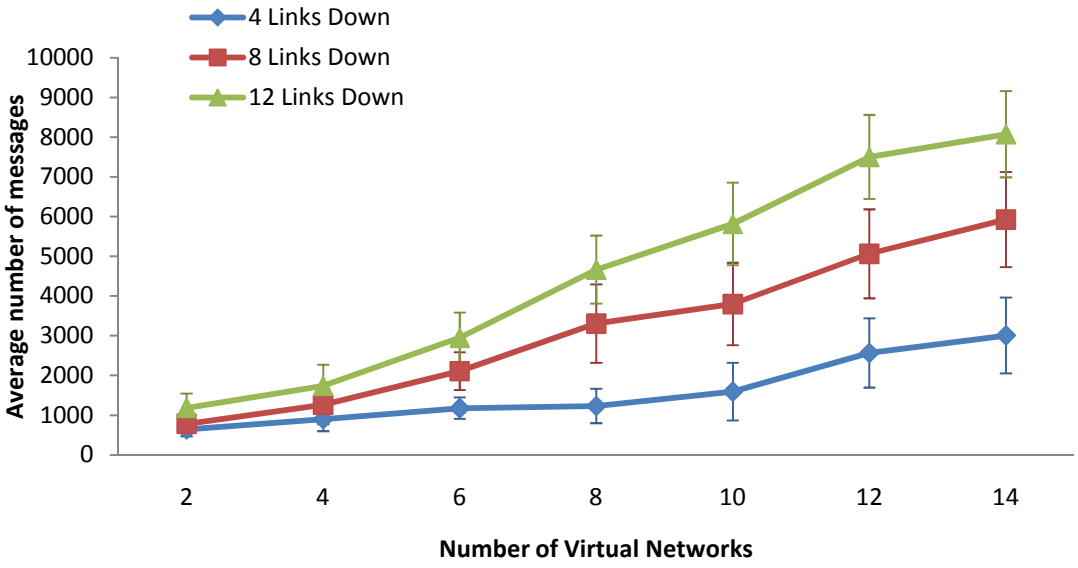


Figure 46: Distributed average number of exchanged messages as the total number of virtual networks into the physical network grows, tested for 4, 8 and 12 physical links failures

An increase of exchanged messages can be observed in Figure 45, as the number of physical link failures increases, as was the case in the previous section.

From Figure 45 and Figure 46, it is possible to verify an increase on the average number of exchanged messages as the number of virtual networks increases. More virtual networks into the same physical network means more virtual links per physical link; therefore, there is a higher probability of affecting physical links with more virtual links, causing an increment on the number of exchanged messages while trying to re-establish them.

In this simulation, as shown on Figure 45 and Figure 46, the average number of messages exchanged during the re-establishment of virtual links is always higher on the distributed approach. This is again due to the use of a mechanism of flooding in the search for alternative paths in the distributed approach, and the one-hop message assumption on the centralized approach.

5.3.2 Virtual link re-establishment success times

This section presents the performance of centralized approach while increasing the number of virtual networks.

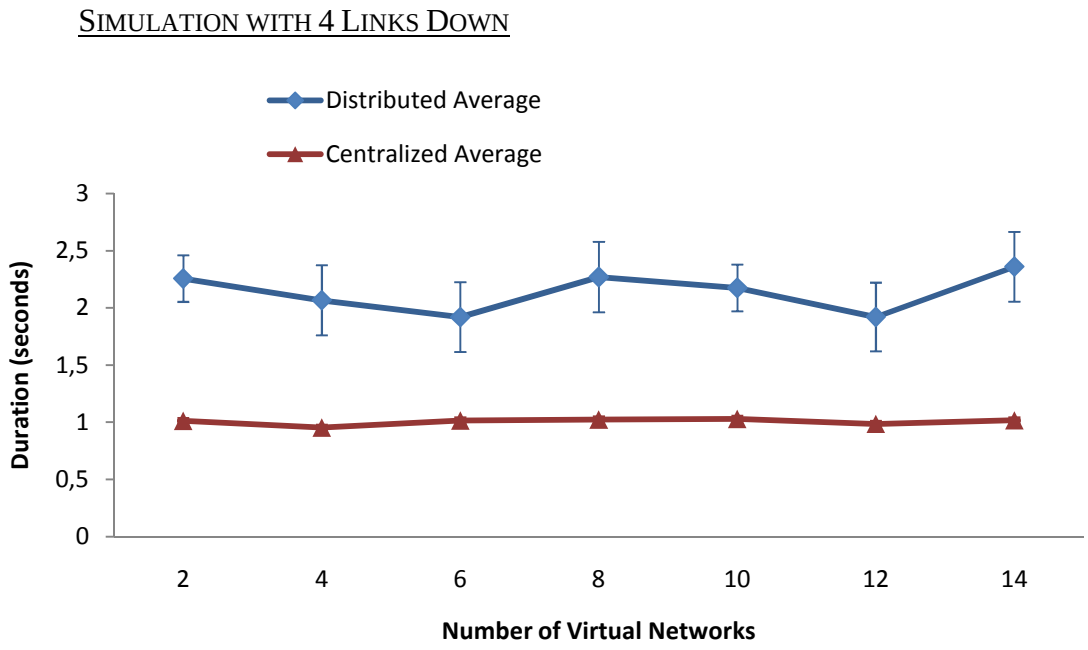


Figure 47: Average durations of re-establishments as the total number of virtual networks into the physical network grows, tested for 4 physical links failures

As presented in Figure 47, for both approaches, the number of virtual networks established on the physical network has small impact on the times obtained for the re-establishments. As was the case of the previous section, this is partly due to the fact that in the distributed algorithm there is a significant communication between the various nodes of the network which in turn implies a large exchange of messages involving a huge accumulation of delays in the trade of these messages. This accumulation of delays appears to be much higher, in distributed approach than in the case of the centralized approach, where it is only required to exchange messages to refresh nodes.

5.3.3 Virtual link re-establishment success rate

This section presents the virtual links re-establishment capability of the centralized and distributed approaches for different numbers of virtual networks.

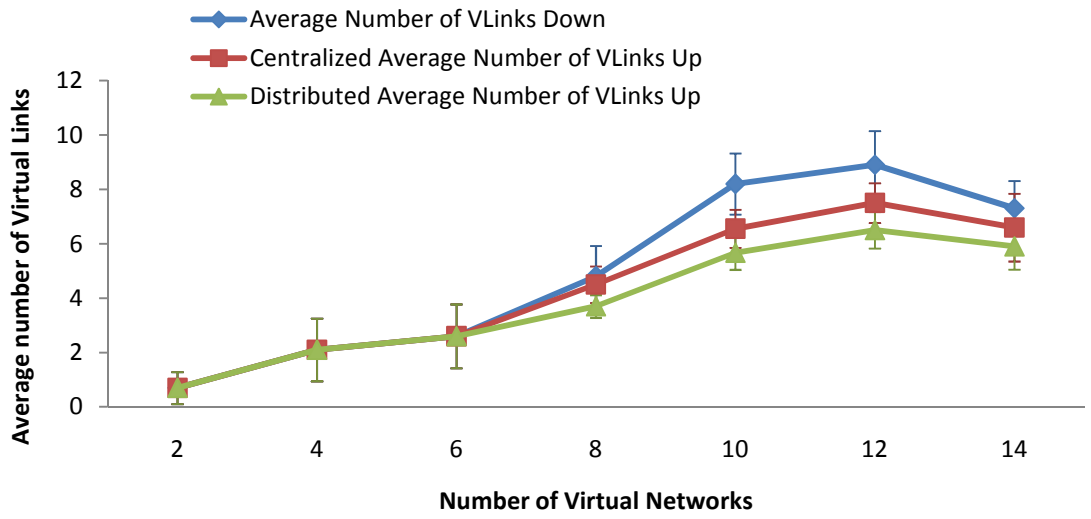


Figure 48: Average number of Virtual Links up achieved for centralized and distributed approaches for the same average number of Virtual Links down as the total number of virtual networks into the physical network grows, tested for 4 physical links failures

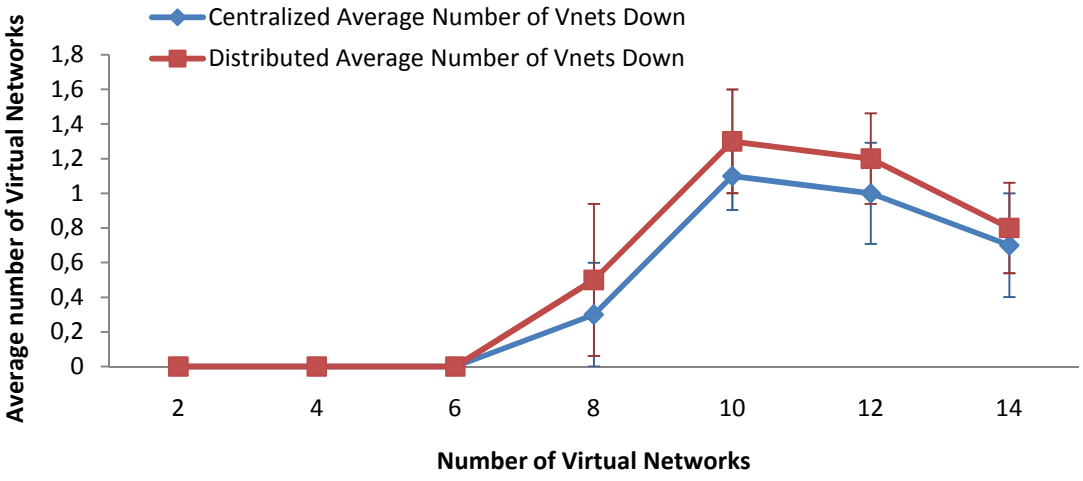


Figure 49: Average number of Virtual Networks impossible of being repaired achieved for centralized and distributed approaches as the total number of virtual networks into the physical network grows, tested for 4 physical links failures

Figure 48 represents the number of virtual networks that were impossible to repair for both approaches; it is possible to see an increase on the average number of Virtual Networks unable of being repaired with the increment on the number of virtual

networks, since more virtual networks increase the probability of more unrepaired networks.

As presented in Figure 48 and Figure 49, both algorithms show a decrease in their effectiveness at the increase in the number of virtual networks. This reduction in efficiency is however more evident in the distributed algorithm. More virtual networks imply a higher probability of being affected more virtual links at the failure of a physical link. Thus, there are more virtual links to be repaired which implies a higher probability that there are no enough resources to restore them all.

Figure 48 also presents differences on the number of Vlinks that were not possible to restore between both approaches, as a result there are also differences on the number of VNets that were not possible to restore between both approaches (Figure 49). This fact is related to the establishment of maximum time for virtual link recovery, thus if this recovery time was beyond the timeout, the algorithm would stop the execution on each node and the attempt to re-establish the Vlink was cancelled.

5.4 Conclusions

The results achieved through the different simulations, in several scenarios, meet, in general, the expectations. In what concerns to the number of messages exchanged, the increase of the total number of physical nodes in the network causes different reactions in both algorithms, since for distributed approach there is an increase in the number of exchanged messages, due to a consequent increase on the number of nodes to do flooding into the networking, which in turn is not observed in the centralized approach. However, this variation in responses is not verified in scenarios where the number of virtual networks grows, since both approaches have the same tendency of increase in the number of exchanged messages. As expected, both scenarios show that the distributed algorithm requires larger exchange of messages, which is related with the need to flood the network, as mentioned above.

Centralized and distributed approaches showed that, both the overall number of physical nodes in the network and the increase in the number of virtual networks in the physical network, do not cause significant changes in recovery times of virtual connections. However, as expected, the recovery times provided by the distributed

algorithm have always been higher, since this algorithm requires a cooperative development between the nodes, which implies the exchange of messages between them and a consequent accumulation of delays. This is not the case in centralized algorithm that only exchanges messages to update the database of each node of the network and the central element database.

In both scenarios, as expected, the centralized approach was more efficient in the recovery of virtual connections and consequently on the recovery of virtual networks, since the proposed distributed algorithm has a timeout established (for the recovery process) that may reduce the number of virtual links repaired, once if a recovery takes longer than this time, the repairing is cancelled.

6 Conclusions and future work

In this thesis centralized and distributed management approaches have been presented, tested and compared. Centralized and distributed algorithms were developed in order to enable the self-repairing of Virtual Networks when one or multiple virtual links are affected by one or multiple physical links failures. An algorithm that aims for centralized approach, is based on the assumption that there is a central element (usually outside the network) knowing all the information of the network and that it is capable of take all the decisions to manage the network. On the other hand, an algorithm based on a distributed approach, aims for the distribution of the intelligence across the network. Thus, every network entity is responsible for gathering essential information from all its direct neighbours, in order being able to take decisions in cooperation with its neighbours.

Both approaches were developed and tested into a simulation environment using NS2. Thus, the algorithms were submitted to multiple simulations with different scenarios that were created in order to test the behaviour and effectiveness of both approaches.

The implemented distributed algorithm was capable of recovering almost all failures in virtual connections, which were possible to be recovered and almost as effective as the centralized algorithm. This difference may be due to the static timeout established for this algorithm where it would stop trying to repair. The distributed algorithm usually takes longer to restore Vlinks and to avoid possible non-convergence situations a maximum time for the recovery was established. If this recovery time goes beyond the timeout, the algorithm stops the executing on each node and cancels the attempts to re-establish the Vlink, possibly resulting in the difference of the number of Vlinks that restored between both approaches.

Nevertheless the success ratio of the distributed approach, both the algorithm's running time and message overhead metrics were favourable to the centralized algorithm. Although the centralized approach has presented better results, it also benefits from certain considerations that may have improved its results. It was decided that, to carry out simulations, it would be possible to have a central element capable of

clustering all the information of the network, growing in capacity, memory, etc., as the network grows, capable of take all the decisions. This centralized characteristic could bring huge problems in scalability (may even be impossible), as the network number of nodes grows, that were not taken into account in these simulations.

Plus, it was decided to carry out the simulations and ignore the times taken by the algorithms during recovery with storage and processing on each node. These were not taken into account. With an increase on number of network nodes that consequently would provoke a higher flow of messages of information towards the central node would translate to an increase in storage time, as well as the processing time of any algorithm for this central node. Therefore we weren't able to verify an increase in its response times when restoring virtual links, with a large on the number of nodes and verify some advantages of the distributed processing.

If the previous assumptions were not taken into account, the centralized approach, would present serious problems of scalability and performance. On the other hand, on the distributed approach, the increase in the number of nodes should not affect significantly its performance, since every node only have to store essential information from its neighbours and since the intelligence is now distributed all over the network elements (enabling the execution of cooperative and parallel mechanisms).

Given the tremendous growth of telecommunications networks and according to the presented results, new studies should be performed in order to improve the distributed algorithm behaviour.

The number of messages exchanged by the algorithm, as well as the time spent, could be reduced through the study and development of more effective mechanisms for communication and discovery of paths between nodes. The reduced time spent on recovery could consequently allow the reduction of timeout which also may allow an increase in algorithm robustness (more virtual links recovered).

7 References

[1] FP7 4WARD, project website, http://www.4ward-project.eu/, Visited 2008/09

[2] Carapinha, J. and Jiménez, J. "Network Virtualization – a View from the Bottom", VISA09, August 2009

[3] Internet Engineering Task Force (IETF), "Management Information Base (MIB) for the Simple Network Management Protocol (SNMP)", RFC 3418, 2002

[4] Internet Engineering Task Force (IETF), "Structure and identification of management information for TCP/IP-based internets", RFC 1155, 1990

[5] Internet Engineering Task Force (IETF), "Introduction to version 2 of the Internet-standard Network Management Framework", RFC 1441, 1993

[6] Internet Engineering Task Force (IETF), "An Architecture for Describing Simple Network Management Protocol (SNMP) Management Frameworks", RFC 3411, 2002

[7] Internet Engineering Task Force (IETF), "The Common Management Information Services and Protocols for the Internet (CMOT and CMIP)", RFC 1189, 1990

[8] Internet Engineering Task Force (IETF), "Remote Network Monitoring Management Information Base", RFC 2819, 2000

[9] Internet Engineering Task Force (IETF), "NETCONF Configuration Protocol", RFC 4741, Dez 2006

[10] Westerinen, A., Schnizlein, J. et all "Terminology for Policy-Based Management", RFC 3198, http://www.ietf.org/rfc/rfc3198.txt, Nov 2001

[11] D. Durham et al. The COPS (Common Open Policy Service) Protocol. RFC 2748

[12] K. Chan et al. COPS Usage for Policy Provisioning (COPS-PR). RFC 3084

[13] S.. Herzog et al. COPS usage for RSVP. RFC 2749

[14] CTIT Technical Report, "Introduction to TMN" April 1999

[15] ETSI ES 282 001: "Telecommunications and Internet converged Services and Protocols for Advanced Networking (TISPAN); NGN Functional Architecture Release 1".

[16] ETSI ES 282 003: “Resource and Admission Control Subsystem (RACS); Functional Architecture”

[17] 3GPP TS 23.203: “Policy and charging control architecture – Release 7”

[18] Franzke, M., Hasslinger, G et all, “4WARD D-4.2 In-Network Management Concept”, March 2009.

[19] European Future Internet Portal, http://www.future-internet.eu/

[20] National Science Fondation Network Technology and Systems Future INternet Design Initiative (NSF NeTS FIND), http://www.nets-find.net

[21] Trossen, D., Briscoe, B., Mahonen, P., et all, “EIFFEL Report: Starting the discussion“, Draft 6 published 13 February 2009, http://www.fp7- eiffel.eu/publications.html

[22] Feldmann, A. “Internet clean-slate design: what and why?”, ACM SIGCOMM Computer Communications Review (CCR), volume 37, number 3, pages 59-64, July 2007.

[23] Schreck, G, “The Virtualization Imperative: Why Virtualization Is The Key To IT Flexibility”, Forrester Research, Inc, November 2007.

[24] Baucke, S. et al. Virtualization Approach: Concept, 4WARD project Deliverable 3.1.0., 2009

[25] Carapinha, J., Melo, M., Sargento, M. “Network Virtualisation from an Operator Perspective", Institute of Telecommunications, Aveiro, Portugal and Portugal Telecom Inovação, Aveiro, Portugal, 2009

[26] Ricci, R., Alfeld, C., Lepreau, J. “A Solver for the Network Testbed Mapping Problem”, SIGCOMM Computer Communications Review,Volume 33, No 2, April 2003.

[27] Gibbons, A. , Algorithmic Graph Theory, Cambridge University Press, 1985.

[28] Bellman, R. “On a Routing Problem”, Quart. Appl. Math., 16:87-90, 1958

[29] Dial, R. “Algorithm 360: Shortest Path Forest with Topological Ordering. Comm”. ACM, 12:632-633, 1969

[30] Dijkstra,E. “A Note on Two Problems in Connection with Graphs.” Numer. Math., 1:269-271, 1959.

[31] Ford, L., Fulkerson, D. “Flows in Networks. Princeton Univ. Press, Princeton”, NJ, 1962.

[32] Moore, E., “The Shortest Path Trough a Maze.” In Proc. Of the Int. Symp. On the Theory of Switching, pages 285-292. Harvard University Press, 1959

[33] Pape, U. “Implementation and Efficiency of Moore Algorithms for Sortest Root Problem”. Math. Prog., 7:212-222, 1974

[34] Abram, J. and Rhodes, I., “A decentralized shortest path algorithm” in Proc. of the 16th Allerton Conf. on Communication, Control, and Computing (Monticello, Ill.), pp. 271-277, 1978

[35] Ravichandran, A. Menon,S. and Shyamasundar, R., “A distributed algorithm for finding the shortest paths in an undirected graph”, Technical Report CS-86-13, Department of Computer Science, Pennsylvania State University, May 1986

[36] Chandry, K. and Misra, J., “Distributed computation on graphs: shortest path algorithms”, Comm. ACM 25, pp. 833-837, Nov. 1982

[37] ns-2, website, http://www.isi.edu/nsnam/ns/index.html, Visited 2008/09

[38] Fall, K and Varadhan, K “The ns manual (formerly ns notes and documentation),”

[39] VINT -Virtual InterNetwork Testbed, project website, http://www.isi.edu/nsnam/vint/

[40] Juniper, “Virtualization in the Core of the Network” retrieved from http://www.juniper.net/us/en/local/pdf/whitepapers/2000299-en.pdf

[41] Ros, F. and Ruiz, P. “Implementing a new manet unicast routing protocol in ns2,” Dept. of Information and Communications Engineering University of Murcia, Tech. Rep., 2004.

[42] Vmware software (virtual machine),” http://www.vmware.com, Visited 2008/09