A Network Virtualisation framework in Operator Perspective

Márcio Melo

Abstract — **Network virtualisation is seen now from a different perspective. Instead of using it as a mere tool for testing new protocols and new architectures, it can be used as a key element of the future internet: it will enable and trigger the development of new protocols and different architectures in coexistence with the existing ones. Recently, the interest from the operators and industry mainstream in network virtualisation has grown quite significantly, as the potential benefits of virtualisation became clearer, both from an economical and an operational point of view. So far, the concept has been mainly a research topic and has been materialized in small-scale testbeds and research network environments. The challenges posed by the deployment of virtualisation in operator networks are still largely unknown and require urgent study. In this paper, we present the 4WARD architecture for network virtualisation and, based on this architecture, we propose a framework for network resource control in virtualisation-based network environments. We also present the developed virtual network testbed and an assessment is made to the framework.**

Index Terms — **Future Internet, Network Virtualisation, Virtual Network Provider and Operator, Infrastructure Provider.**

I. INTRODUCTION

In networking, new trends are emerging every day, applications are pumping up as mushrooms, sensors are invading everywhere and the network seems to be getting far away from this reality due to its ossification and lack of dynamism. Clean slate approaches are being hailed as the solution for a better network [1]-[6]. For now, the fulfilment of this discussion seems to be somehow fuzzy. In the meantime, network virtualisation has achieved an increasing prominence in networking and telecommunications fields in the last couple of years. Initially, the interest in network virtualisation was mainly pushed by Future Internet research initiatives [7]-[10], mainly with the objective to find a platform on which novel Internet architectures could be experimented and evaluated without limitations or constraints, namely those associated with the traditional IP model.

Later on, it became clear that virtualisation could constitute a key component of next-generation Internet architecture itself [11], and not just as a mere platform for experimentation. Perhaps more importantly for network operators, it also became clear that network virtualisation could provide a number of short/medium term business advantages, with potential reduction of costs and increase of revenues, as an interesting tool from an operational point of view.

However, the large-scale deployment of network virtualisation by commercial operators faces a number of challenges, most of which have not been fully evaluated up to now. One of the key issues with network virtualisation is the management and control of network infrastructure resources - more specifically, how to map, or embed, virtual resources into physical substrate resources. Although this problem has been already addressed in the literature [12]-[17], the proposed solutions have been mainly oriented to research network environments, overlooking key constraints and requirements of commercial operator networks.

Network virtualisation has followed the usual development cycle, which started with research and testbed experimentation through a number of research initiatives. Validation for deployment in commercial operator networks is still largely unaccomplished and represents a logical continuation of the research efforts so far. This paper aims at providing a contribution in this direction. Our main focus is the management and control of network infrastructure resources in a network virtualisation environment. We present both the concept and an experimental testbed that entails the management and control of virtual networks in an operator perspective. The starting point of this paper is the network virtualisation architecture and business model developed in the framework of the 4WARD project [18].

The paper is organized as follows. Section II provides a general overview of network virtualisation. The 4WARD network virtualisation business model and roles are briefly explained. In particular, the interface between the virtual network provider and the infrastructure provider is analyzed in detail. Section III examines the problems of resource control in a network virtualisation environment, mainly from the point of view of the infrastructure provider, and proposes a solution for resource negotiation and control. Section IV explains with some detail a framework for resource, allocation, monitoring and controlling and present some printouts of the framework environment. Section V briefly describes a small-scale testbed, currently under development, introducing experiments to evaluate the framework and

comments on the obtained results. Finally, section VI concludes with a summary and possible directions for future work.

II. NETWORK VIRTUALISATION BACKGROUND

A. Historical Perspectives

It is clear now the importance of reducing the energy consumption with respect to the carbon footprint. The consumption of energy due to Telecom and Broadcasting has been frenetically increased over the years. In Japan the consumption of energy due to Telecom and Broadcasting achieved astonishing proportions [19]. Several procedures can be taken for reducing the energy consumption such as putting nodes that are not being used into sleep mode or by using "green" protocols. In [20] it is discussed the impact on network protocols by putting network interfaces and components into sleep for saving energy and in [21] it is presented and evaluated two forms of power management schemes that reduce the energy consumption of networks. Network Virtualisation has been hailed for achieving energy efficiency [22].

The virtualisation of network will trigger the development of green protocols and will facilitate the load distribution among the network based on the usage of the network or even in data previsions, allowing this way to save energy by putting the unnecessary resources into idle stages. The appliance of power management schemas will be a reality. It will be possible to put the virtual components into sleep modes or even in hibernated modes.

Network virtualisation, as it is viewed in this paper, supersedes all the above variants and is based on two fundamental building blocks: node virtualisation and link virtualisation.

The main advantages of network virtualisation realized by the industry to make network virtualisation potential going beyond the one of future Internet scenarios, mainly to operators and service providers on a short/medium term, are as follows [23],[24]:

- Reduction of costs: by using a single virtualised infrastructure to run multiple services, CAPEX and OPEX can be reduced, compared to the typical scenario where different types of service (e.g. voice, data, broadcast) are run in separate networks.
- Increase of revenues: by sharing infrastructure, the network operator achieves a better utilization of the network resources and optimizes profitability.
- Flexible network planning: the swift and easy establishment of virtual networks can be used as a safeguard against unpredictability of the service demand.
- Security and isolation: virtualisation can provide real isolation of network resources, with benefits in terms of fault isolation, security, and performance guarantees.
- Flexibility and programmability: virtual networks can be tuned to fulfil specific service and application requirements (e.g. security, performance, dependability), thus a "one-size-fits-all" approach is no longer required.

Business models and roles

Players in the network virtualisation model (Fig. 1) are different from those in the traditional networking model. The main distinction is the presence of three different roles: Infrastructure Providers (InP), Virtual Network Providers (VNP) and the Virtual Network Operator (VNO), in contrast with the conventional model, characterized by a single role: Internet Service Provider (ISP) in the conventional model [8, 9, 10]. From a commercial point of view, this decoupling amortizes high fixed cost of maintaining a physical presence by sharing capital and operational expenditure across multiple infrastructure providers as well as one infrastructure provider for several service providers. It should be noted that business roles do not necessarily map one-to-one to distinct business entities (i.e., any business entity can assume multiple roles).

From a business model point of view, a very significant impact of network virtualisation is the ability to cleanly decouple infrastructure from services, which has been pursued for a long time but never really accomplished. This potential separation of infrastructure and services paves the way for the creation of new business models and roles.

Network virtualisation can be deployed in a number of very different scenarios and business models but, in general, it is based on three distinct roles, as defined by the 4WARD project, and represented in Fig. 1:

- The Infrastructure Provider (InP) deploys and runs the network physical resources, and partitions them into isolated virtual resources using some virtualisation technology. These resources are typically offered to virtual network operators and not to end users (but the customer of the InP might as well be a corporation using the virtual network for its internal use, rather than to build commercial end user services). The InP has visibility into what resources are leased to each virtual network (VNet), but not into the protocols running inside.
- The virtual network provider (VNP) is responsible to find

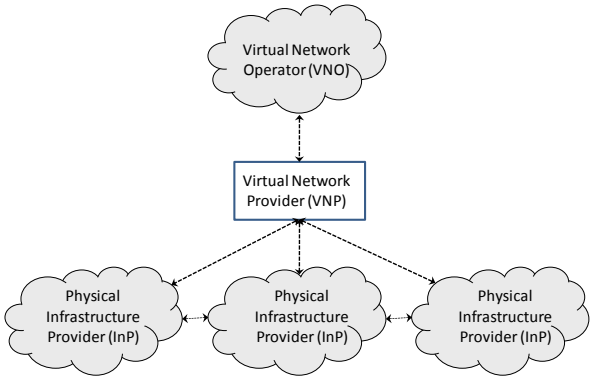


Fig. 1. Network virtualisation business roles.

and compose the adequate set of virtual resources from one or more infrastructure providers, in order to fulfil the virtual network operator request. The VNP leases slices of the virtualised infrastructure from one or more InPs and puts them together. In reality, what the VNP provides is not a network, but just an empty container where the virtual network operator builds the protocols that will make the VNet to come alive. The role of the VNP is particularly important in scenarios where multiple InP domains are involved, but may be redundant in the case where a VNet is limited to a single network infrastructure domain.

- In each isolated network partition, the virtual network operator (VNO) is, in principle, free to deploy any protocol stack and network architecture, independently of the underlying network infrastructure technology. The VNO operates, maintains, controls and manages the virtual network. From a functional viewpoint, the role of the VNO should be indistinguishable from that of any operator running a native network infrastructure. Ideally, the fact that resources are virtual, rather than physical, should not imply any major impact from an operational point of view. VNOs have a unified view of the network, regardless of the multiple infrastructure domains on which it is built.

It should be noted that this model does not preclude the possibility of more than one role being played by a single entity. In a vertically integrated scenario, the three roles would be typically played by the same operator. Yet, even in this case, a functional separation of roles based on the model above should make sense.

B. The VNP-InP interface

The VNP/InP interface is a key aspect of the network virtualisation architecture. Through this interface, the VNP is able to request the establishment, modification or removal of virtual networks (supposedly, as a result of a corresponding request from the VNO). In its turn, the InP is supposed to acknowledge the VNP requests and notify any relevant event (e.g. a network error). The split of responsibilities and the information flow between the VNP and the InP are therefore of the utmost importance. Ultimately, this will depend on the information flowing through the VNP/InP interface in both directions, as illustrated in Fig. 2.

In principle, one of two basic approaches could be taken:

- The InP announces the resources which are available to be leased by VNPs, i.e. the internal structure of the InP infrastructure (or a virtual representation thereof) and the current state of resources. The InP is supposed to publish this information in some way, e.g. by means of a specific notice-board, such as proposed in [25]. It is up to the VNP to pick one or multiple InPs amongst all possible candidates, that would be able to provide the required resources, while fulfilling any applicable constraints (e.g. performance guarantees, cost). Since the relationship between the VNP and the InP is quite straightforward, the complexity of the VNP-InP interface is quite low in this case. This approach is appropriate for research testbeds, or whenever there is a trust relationship between the InP and all potential VNPs; in a commercial environment this is not likely to be the case, except perhaps in a vertically integrated scenario, in which VNP and InP have a common business affiliation.

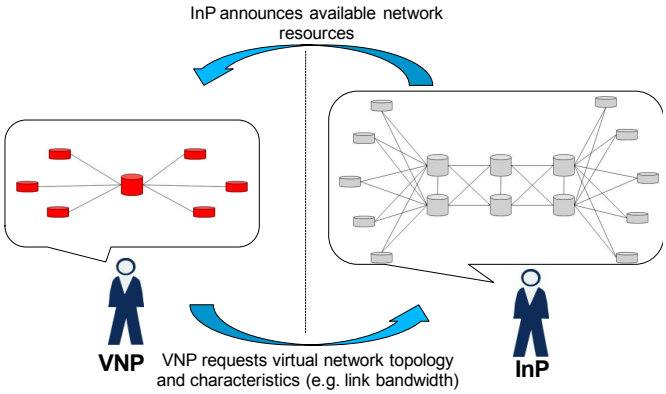


Fig. 2. VNP/InP information flow

- The InP exposes a minimal set of resources, namely the points of presence (PoPs) and hides the internal structure and the state of resources. Because the VNP does not know in advance whether the InP is able to fulfil its request, the virtual network characteristics have to be provided to the InP and a negotiation has to be carried out through the VNP/InP interface prior to VNet establishment phase, when the resources are actually reserved. In turn, the InP decides whether the request can be accommodated in the physical resources and, if so, maps virtual nodes into substrate nodes and finds the substrate path between every pair of directly connected virtual nodes. This is likely to be the approach followed in a commercial environment, where a relationship of trust between VNPs and InPs is not expected.

As stated before, in this paper we are mainly interested in commercial network environments; therefore, in the following sections we are mainly focused on the second approach.

III. CONTROLLING VIRTUAL NETWORK RESOURCES

This section presents an architecture for automatic virtual network creation and the corresponding approach for control of virtual network resources. It comprises the main building blocks of the network and their functionalities, and the communication required to provide the virtual network creation.

A. Building blocks

Prior to the creation of a new virtual network, the InP should find the adequate physical resources, taking into account the current state of the network and the level of occupancy of the network resources, at that moment, in the case of an "on-the-fly" reservation, or at the requested future time, in the case of an advance reservation.

Several theoretical approaches have been proposed to handle this problem [12]-[17]. However, these solutions are mainly oriented to small-scale networks or research testbeds, and do not take into account the constraints that usually apply in a commercial environment.

In practice, the mapping of virtual nodes into physical nodes is often constrained by physical location, in which case the selection of the physical node to associate with a specific virtual node is fixed, or limited to a small set of choices. This is the case of edge nodes, or Points of Presence (PoPs), which for economical reasons are supposed to be physically close to customers or end-users. Typically, at least one virtual node should be located in each geographic area (e.g. city, region) where the service is to be deployed. By contrast, for other types of virtual nodes, physical location is not relevant from the VNP point of view – this is usually the case with core nodes, with no direct connection to end users. The mapping of virtual nodes and links into physical nodes and links should follow a set of constraints and optimization criteria to be defined by the InP (e.g. minimum cost, resource load balance, segregation of resources according to the service type), and can be materialized in a complex algorithm.

Physical resource control and virtual resource embedding include three basic components (Fig. 3):

VNet admission control: the InP verifies whether there are available resources to fulfil the virtual network request made by the VNP, and decides whether it can be accepted, or not. VNet admission control does not necessarily find an optimal solution for a VNet yet – this is supposed to be the role of resource mapping, as described below – it only verifies that, at least, one solution can be found.

Resource mapping: the InP identifies the set of possible substrate nodes and links to host the requested virtual network and selects the optimal solution.

Re-optimization: the network state keeps changing as new VNets are setup and others are torn down, or as a result of node or link failure conditions. This often leads to inefficient utilization of resources, in which some parts of the network infrastructure (either link resources or node resources) become excessively loaded, while others are under-utilized. Therefore, the capability to re-optimize the allocation of virtual resources across the substrate network without traffic disruption, either on a periodic basis or triggered by a specific event (e.g. when a specific resource availability threshold has been reached) is a key VNet requirement.

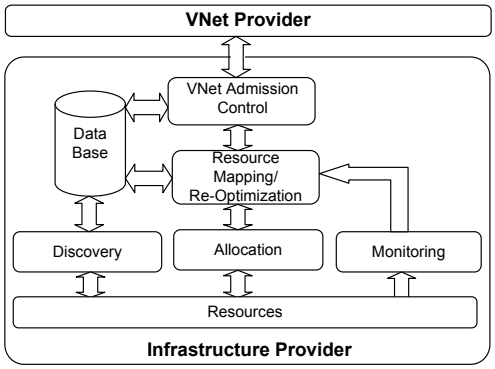


Fig. 3. InP block diagram

In addition, the resource management process typically makes use of two auxiliary components:

- *Discovery*: this function is responsible for discovering network resources and providing them available for the admission control and mapping functions.
- *Monitoring*: this function collects real-time information from nodes and links, and signals any significant deviation from the expected network behavior.

B. VNet setup negotiation process

As explained before, it is likely that in most cases the VNP has limited knowledge of the physical resources provided by the InP. On the other hand, there will be multiple candidate InPs to provide the required network resources in many cases. Therefore, the VNP must be able to inquire a set of candidate InPs and, based on their responses, select one or more that will actually provide the network resources simultaneously and cooperatively. This requires the VNP/InP negotiation to be divided in two stages, as depicted in Fig. 4:

Query: the VNP inquires the InP about the availability of resources to build a specific VNet. The InP is expected to provide a yes/no reply, possibly with additional information, e.g. cost, QoS parameters.

Commit: the VNP requests the reservation of network resources and the InP enforces the corresponding resource reservation, after establishing the mapping between virtual and physical resources. It should be noted that virtual networks can be created "on-the-fly", i.e. just before the resource is required, or in advance, i.e. at some future point in time. In either case, a time may be optionally specified for

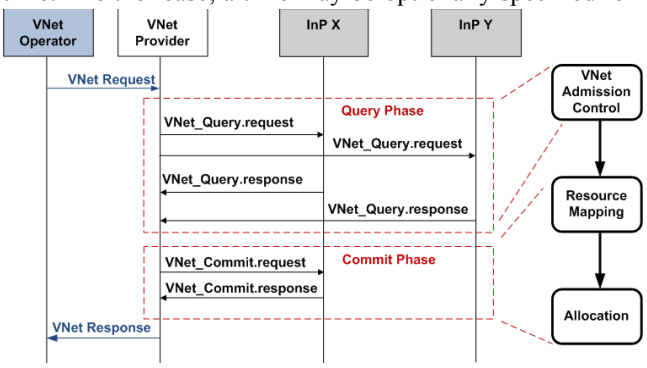


Fig. 4. VNet creation sequence chart and flow diagram

TABLE I
VIRTUAL NETWORK CHARACTERISTICS

Network virtualisation components	Parameters
Virtual network	- Virtual network ID
	- Start time of the service (optional)
	- End time of the service (optional)
	- Class of service / reliability
	- Preemption level
Virtual node	- Node ID
	- Physical location
	Physical node ID
	Don't care
	- Minimum capacity
	CPU
	Memory
	Storage
Virtual link	- End points (source / destination node IDs)
	- Traffic characteristics

resources to be released; otherwise, the VNet will only be torn down through explicit signalling.

From the InP point of view, a relevant issue is how to map the blocks represented in Fig. 3 into these two phases. The right hand side of Fig. 4 suggests a possible approach, but this will be further discussed in the next section.

The VNP is expected to build the virtual network topology and define resource capacity requirements, namely link bandwidth and node computational capability. As discussed previously, other characteristics such as geographical location of the edge nodes will be needed in most cases. The information provided by the VNP to the InP must contain a model of the virtual network topology as a graph, with a set a virtual nodes and virtual links and including the applicable constraints (e.g. link bandwidth, node computational capacity, physical location). Each virtual node and virtual link must be characterized by a number of parameters. A tentative list of parameters to characterise virtual networks, nodes, and links is shown in Table I.

C. Signaling and control

As explained earlier the creation of a VNet involves two phases, query and commit.

Fig. 4 depicts the VNet creation process: the left hand side represents the message flow between the VNO and the VNP when a new VNet is requested. In this example, the VNP contacts two candidate InPs to accommodate that VNet, InP X and InP Y, and then decides to opt for InP X, based on some criterion, e.g. InP X provides the requested resources at lower cost than InP Y. Then, the process continues with the commit phase, in which the resources are actually reserved.

1) Query Phase

The process is started when the VNO sends a *VNet Request* to the VNP, including the VNet topology and its constrains, node constrains (i.e. physical location, CPU) and link constrains (i.e. bandwidth, delay).

The VNP is then in charge of assigning a VNet ID and an ID for each virtual resource. Then a *VNet_Query.request* message is sent to one or more InPs. This message must contain the VNet ID, the nodes/links IDs, the VNet topology and its specifications, according to Table I.

At a first stage, the InP will perform a VNet Admission Control, checking that every requested virtual node and virtual link can be accommodated by at least one substrate node and substrate link, respectively. If one or more virtual nodes and/or virtual links cannot be accommodated due to lack of resources in the substrate (i.e. insufficient bandwidth or computational resources), the process should be stopped here and the *VNet_Query.response* is sent to the VNP, indicating that the request cannot be fulfilled (optionally, indicating the reason of the failure).

Otherwise, if every virtual resource can be accommodated by the substrate, then a *VNet_Query.response* message with a positive reply, including the VNet ID, should be sent to the VNP. It should be noted that the VNet Admission Control is not expected to find the optimal solution for the virtual-to-physical resource mapping problem yet, but only whether at least one solution exists. This is understandable, since the resource mapping is the most complex step of this process, and in many cases, the query will not be followed by a corresponding commit.

However, this may not be the case in all circumstances, and the InP may decide to go further than just performing admission control. So, optionally, at a second step, the InP may perform a pre-reservation by mapping the VNet into the network infrastructure, making use of an optimization algorithm, knowing *a priori* that every requested virtual resource can be accommodated. The choice of the first available or optimal solution may be based on different criteria, such as: preferring substrate nodes with more plentiful resources, selecting substrate links with more available bandwidth, link aggregation (i.e. virtual link that maps into 2 substrate links) and link segmentation (i.e. virtual link spanning through multiple substrate links).

After obtaining the best solution, the InP must perform a reservation (at this stage, at logical level only) of the concerned substrate resources. This reservation will be cancelled if a specific timeout expires without any effective reservation being made by a corresponding VNet_Commit.request from the VNP, and the reserved

resources will be released again. Optionally, this timeout should be included in the *VNet_Query.response* message.

2) Commit Phase

If the VNP receives one or more positive responses from the candidate InPs in the query phase, the process will typically continue with the selection of the InP (if more than one candidate InP answered positively), followed by a *VNet_Commit.request*, with the corresponding VNet ID.

After receiving the *VNet_Commit.request*, the InP verifies whether a pre-reservation exists for the given VNet ID and if it is still valid. If so, it proceeds with the allocation of the virtual resources. After allocating each virtual resource, it sends a *VNet_Commit.response*, including the VNet ID and the ID of each virtual resource. If the VNet ID is not valid or the pre-reservation has expired, or if for some reason it cannot allocate any particular virtual resource, the InP should send a *VNet_Commit.response* indicating the reason of the negative response.

If a pre-reservation has not been performed beforehand, then the complete process has to be executed. A potential issue in this case is that, because no resources have been reserved, it may be the case that when the commit request arrives, the resources are no longer available. Thus, from the point of view of the InP, there is a trade-off between increasing the complexity of the query phase and improving the reliability of the whole process.

IV. RESOURCE ALLOCATION MONITORING CONTROLLING FRAMEWORK

In this section we present and describe the Resource Allocation, Monitoring and Controlling (RAMC) framework, which provides the build up, management and control of virtual networks using a small-scale network virtualisation testbed. In the envisioned network virtualisation environment, the infrastructure provider is responsible for managing and controlling physical network resources. Virtual networks are established as a result of a VNet Provider explicit request (following 4WARD business model), or through the network management console. Whenever a request to establish or modify a virtual network is received, the network resource controller, based on specific resource utilisation policies, should decide whether or not the request can be accepted and, if it can, how to map the virtual resources into physical resources. The main functionalities of the RAMC framework are as follows:

1. *Virtual Network creation*: Creates a new virtual network, based on a specification in XML file;
2. *Virtual Network deletion*: Removes a virtual network and releases all associated resources;
3. *Resource discovery*: Discovers the topology of the physical substrate and identifies the complete set of virtualisable resources;

```
-----------------------------------------------
| Resource Allocation Monitoring and Control |
-----------------------------------------------
a) List Virtual Networks
b) Show Virtual Network
c) Show Substrate Network
d) Show Substrate Node
e) Create Virtual Network
f) Delete Virtual Network

u) Update DataBase
x) Exit
```

Fig. 5. RAMC framework main menu

```
| Virtual Networks          |
| VNetId     | VNodes | VLinks |
| Grey       | 6      | 9      |
| Vegas      | 4      | 4      |
------------------------------
```

Fig. 6. Output of 'List Virtual Networks' function

```
-------------------------------------------------------------
| Virtual Network: Vegas                                    |
-------------------------------------------------------------
| Nodes:                                                    |
| VNodeId    | SNodeId   | Status   | MEM (MB) | HD(GB)|
| Sara       | Bree      | Running  | 500      | 1     |
| Catherine  | Lynette   | Running  | 500      | 1     |
| Grissom    | Eddie     | Running  | 500      | 1     |
| Nick       | Susan     | Running  | 500      | 1     |
-------------------------------------------------------------
| Links:                                                    |
| SourceId  | DestId    | Physical Path                     |
| Sara      | Catherine | Bree-Lynette                      |
| Sara      | Grissom   | Bree-Eddie                        |
| Grissom   | Nick      | Eddie-Mary-Susan                  |
| Catherine | Nick      | Lynette-Gabrielle-Susan           |
-------------------------------------------------------------
```

Fig. 7. Output of 'Show Substrate Network' function

4. *Monitor virtual resources*: Provides overall information about all VNets that share the same substrate network. Provides the current status of the resources allocated to a specific VNet, uniquely identified by VNetID: virtual machines, virtual links (network path), storage capacity, link capacity.
5. *Monitor physical resources*: Provides information about the physical resources:
    a. *Nodes*: static parameters (CPU, OS, RAM, storage, capacity [in terms of Virtual Machines]) and dynamic parameters (occupancy [# VMs that can still be accepted], available storage, available memory);
    b. *Links*: static parameters (link technology, capacity in Mbit/s), available bandwidth per physical link.

A. Network Virtualization Testbed

To demonstrate the network virtualization concept and to test the functionalities provided by the RAMC framework, a small-scale testbed was implemented. This tesbed is composed by 6 substrate nodes disposed accordingly to fig.

4. On the top of the substrate network, 3 different VNets were created on-demand and on-the-fly.

The instantiation of the virtual nodes is performed using XEN hypervisor [26]. To make this process faster and easier, it was used clone techniques, where it has been pre-configured one or more (default) virtual nodes to be replicated. On this testbed, each virtual node needs to have its own filesystem, where we can refer the creation, or to be more accurate, the cloning of the new filesystem. This is the process that takes longer time, in average it can take up to 20 seconds per virtual node depending on the substrate node characteristics. To enable the virtual links, VLANs [27] were configured in each substrate link.

Figure 5 to Figure 9 are printouts of the implemented command line interface. The main menu of the framework is shown in fig. 5. Option a) gives the identification of all virtual networks that are accommodated in the substrate and their size as depicted in fig. 6. In option b), the user is prompted to insert the identification of the virtual network which he wants to view, and the output will be the virtual network characteristics, as shown in fig. 7. Option c) provides static and dynamic parameters of the substrate resources, as presented in fig. 8. Option d) prompts the user to insert the substrate node identification, and the output will be the node characteristics and its virtual nodes, as demonstrated in fig. 9. Finally, options e) and f) can be used to create and delete a virtual network, respectively.

V. EVALUATION

This section describes a small scale testbed which was implemented to demonstrate the network virtualisation concept. Then, it presents some initial results of the virtual network creating process and tear-down through the use of the RAMC framework.

To demonstrate the concept of network virtualisation, it

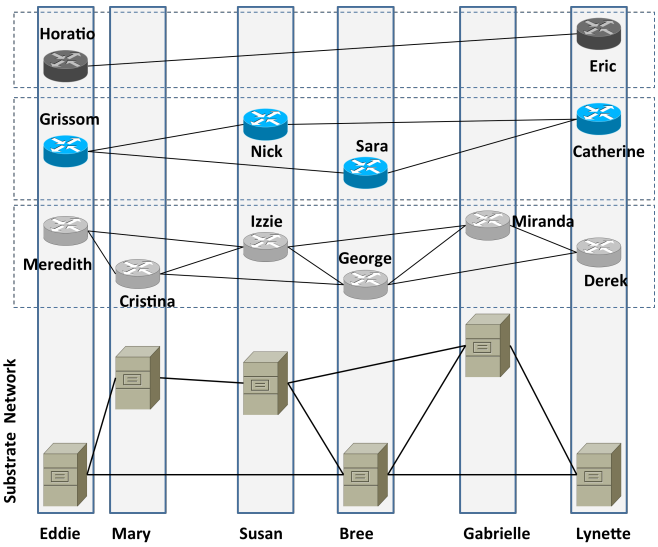
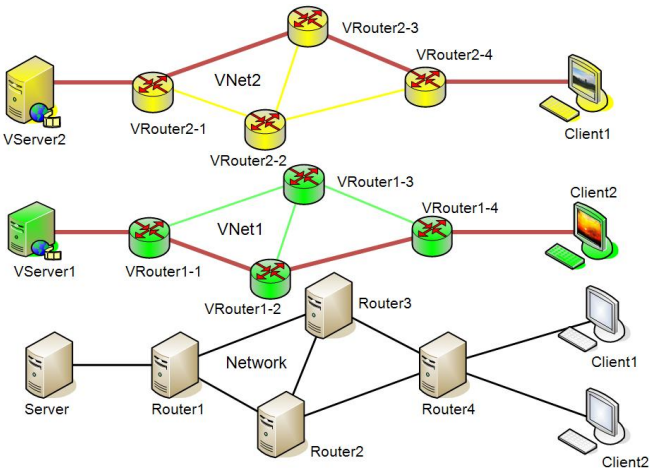


Fig. 4. Network Virtualization Tesbed

was implemented a small scale testbed with 7 substrate nodes, 1 server, 4 routers with no routing protocol running and 2 clients. Fig 10 depicts the implemented testbed. On top of the substrate network, 2 VNets were created manually, VNet 1 and VNet 2; XEN hypervisor [26] was used for the creation of the Virtual machines. Regarding that some virtual machines will act as routers, for instance VRouter1-1, the XORP open source router software [28] was installed in each



of them (i.e. virtual router) and the Open Shortest Path First (OSPF) protocol was configured as a routing protocol. In the virtual servers, a video stream will be activated through VLC media player [29], and the virtual servers will stream a video across the VNets; VLC was also installed in the clients. To

```
|Substrate Network                                                                                                  |
| Nodes:                                                                                                            |
| NodeId     | MEM (MB)   | HD(GB)     | ARCH   | Cores | CPU (Mhz) | CPU idle(%) | VMs |
| Eddie      | 3018/6616  | 390/450    | 64     | 4     | 2400.17   | 100         | 2/8 |
| Susan      | 67/2955    | 117/224    | 64     | 2     | 3400.23   | 86.57       | 2/3 |
| Bree       | 2938/6616  | 287/450    | 64     | 2     | 3000.02   | 100         | 2/4 |
| Lynette    | 66/3046    | 112/224    | 64     | 2     | 3400.29   | 100         | 2/4 |
| Gabrielle  | 121/2533   | 168/224    | 64     | 2     | 2137.04   | 100         | 1/3 |
| Mary       | 100/7616   | 386/450    | 64     | 4     | 2666.67   | 100         | 1/8 |
| Links:                                                                                                            |
| SLinkId        | SourceId     | DestId       | Speed (Mbps)   | Status                                            |
| 10.10.101.0    | Eddie        | Bree         | 1000           | Link Up                                           |
| 10.10.104.0    | Susan        | Bree         | 1000           | Link Up                                           |
| 10.10.106.0    | Bree         | Lynette      | 1000           | Link Up                                           |
| 10.10.107.0    | Lynette      | Gabrielle    | 100            | Link Up                                           |
| 10.10.105.0    | Bree         | Gabrielle    | 100            | Link Up                                           |
| 10.10.103.0    | Susan        | Gabrielle    | 100            | Link Up                                           |
| 10.10.102.0    | Susan        | Mary         | 1000           | Link Up                                           |
| 10.10.100.0    | Eddie        | Mary         | 1000           | Link Up                                           |
```

Fig. 8. Output of 'Show Substrate Network' function

```
| Substrate Node: Eddie                                                |
| MEM=3018/6616 MB | HD=390/450 GB | Number of Cores=4 |
| Virtual Nodes:                                                       |
| VNodeId    | VNetId     | Status   | MEM (MB)  | HD(GB)  |
| Meredith   | Grey       | Running  | 500       | 1       |
| Grissom    | Vegas      | Running  | 500       | 1       |
```

Fig. 9. Output of 'Show Substrate Node' function

enable the virtual links, VLANs [27] were configured in each substrate link, VLAN2 for VNet1 and VLAN3 for VNet2. The VNets are similar in terms of topology, and the virtual

resources belonging to VNet1 have the same IP address as the corresponding resources in VNet2.

Our experiment starts with virtual server (VServer) 1 and 2 streaming two different videos. With the OSPF protocol, stream 1 is forced to use Virtual Router1-2 (VRouter) by giving lower port cost to it in VRouter1-1, and the stream 2 is forced to use VRouter2-2. The clients will immediately start receiving the corresponding stream. After a while, the connection between Router1 and Router2 is broken, causing the breakdown of the corresponding virtual links, between VRouter1-1 and VRouter1-2 and between VRouter2-1 and

Fig. 10. Network Virtualisation Testbed.

VRouter2-2. Therefore, stream 1 will be interrupted, and stream 2 will continue without problems since it is not using that connection. After a couple of seconds, stream 1 will be restored and will be using the virtual link between VRouter1-1 and VRouter1-3 and between VRouter1-3 and VRouter1-4 in a different virtual network from the stream 2. The recovery time is just due to OSPF converging process.

Stream 1 will have no influence in stream 2: the measured round trip time of the packets in stream 2 did not change with the recovery of stream 1. Both streams will be using the same path but in different virtual networks. This demonstrates the isolation between VNet1 and VNet2 in terms of performance and IP addressing.

This testbed will be the basis for more complex experiments in the future, such as resource management algorithms, and the support of automatically controlled VNets.

To make a preliminary evaluation of the RAMC framework, and regarding that this framework was not built considering performance aspects, two experiments were designed where the performance metrics considered are the following: time that takes to instantiate a virtual network and the time that takes to tear down a virtual network, in respect to the number of nodes of the virtual network and regarding the number of physical nodes available. The virtual networks created were composed by 2, 4 and 6 nodes as depicted in fig. 3. The experiments were performed 10 times for each virtual network, and the graphics correspond to the mean and 95% confidence interval.

The first experiment measures the instant when the RAMC framework receives the request for a new virtual network until all virtual nodes and virtual links have been allocated. The second considers the instant when the RAMC receives the request to remove the virtual network until all virtual nodes and links are completely released.

From Figure 11 we can see that the virtual network creation process grows linearly with the number of nodes. To create a virtual network with 2 nodes and 1 link, it takes less than 1 minute. If we try to extrapolate this value to a virtual network with medium size of 200 nodes, it will probably take more than 100 minutes. This time can be acceptable in some conditions, but it can be decreased if the virtual nodes were created in parallel and not sequentially as it is done in the framework.

In fig. 12 it is presented the virtual network tear down process. This process also grows linearly with the number of nodes. Picking the example of the virtual network with 2 nodes, it takes less than 10 seconds to remove it.

VI. CONCLUSION AND FUTURE WORK

A new architecture for Network Virtualisation based on the 4WARD project has been presented: it promotes the deployment of new protocols and enables the emergence of new players in the telecommunications market. A new role has been defined, the one of the Virtual Network Provider; this element will trigger the competition and/or cooperation among different Infrastructure Providers to provide substrate resources.

As can be seen from the testbed description, the base testbed is running with intervention of the RAMC framework. Currently, we are working on the deployment of an optimal algorithm for resource management between different virtual networks, taking into account the Infrastructure Provider constraints.

Regarding the performance aspects of the framework, we are working on a different approach that will allow us to create virtual nodes in parallel and automatically.

ACKNOWLEDGMENT

The author wishes to thank Prof. Drª Susana Sargento and Eng. Jorge Carapinha for sharing their knowledge and by helping with the reviewing of the paper.

REFERENCES

[1] A. Greenberg, G. Hjalmtysson, D.A. Maltz, A. Myers, J. Rexford, G. Xie, H. Yan, J. Zhan, and H. Zhang, "A clean slate 4D approach to network control and management," ACM SIGCOMM Computer Communication Review, vol. 35, 2005, p. 54.

[2] S.M. Bellovin, D.D. Clark, A. Perrig, and D. Song, "A Clean-Slate Design for the Next-Generation Secure Internet," NSF Workshop Report, 2005.

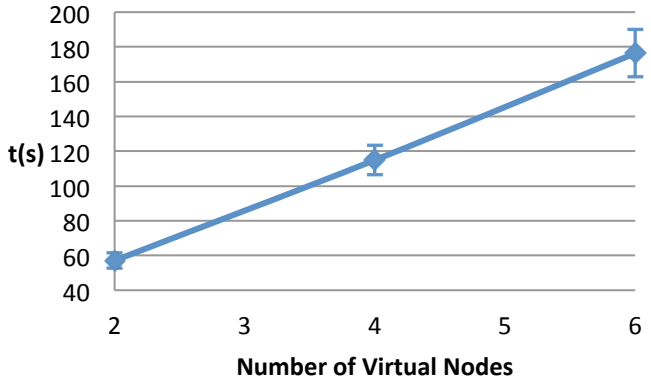


Fig. 11. Virtual Network Creation

[3] A. Feldmann, "Internet clean-slate design: what and why?," ACM SIGCOMM Computer Communication Review, vol. 37, 2007, p. 64.

[4] T.H. Davenport and D.B. Stoddard, "Reengineering: business change

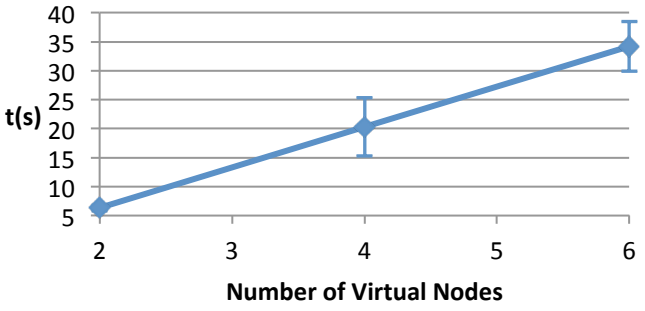


Fig. 12. Virtual Network Removal

of mythic proportions?," MIS quarterly, vol. 18, 1994, pp. 121–127.

[5] M.C. Love, "Starting over with a Clean Slate: In Praise of a Forgotten Section of the Model Penal Code.," Fordham Urban Law Journal, vol. 30, 2003, pp. 1705–1742.

[6] J.W. Newstrom, "The Management of Unlearning: Exploding the" Clean Slate" Fallacy.," Training and Development Journal, vol. 37, 1983, pp. 36–39.

[7] L. Peterson, T. Anderson, D. Culler, and T. Roscoe. A blueprint for introducing disruptive technology into the Internet, HotNets, 2002.

[8] T. Anderson, L. Peterson, S. Shenker, and J. Turner. Overcoming the Internet impasse through virtualization, IEEE Computer Magazine, vol. 38, pp. 34-41, 2005.

[9] N. Feamster, L. Gao, J. Rexford. How to lease the Internet in your spare time, ACM Computer Communication Review, January 2006.

[10] Y. Zhu, R. Zhang-Shen, S. Rangarajany, J. Rexford. 2008. Cabernet: Connectivity Architecture for Better Network Services, Workshop on Rearchitecting the Internet, December 2008.

[11] J. Touch, Y. Wang, L. Eggert, G. Finn. A Virtual Internet Architecture, ACM SIGCOMM Workshop on Future Directions in Network Architecture, Karlsruhe, Germany, 2003.

[12] Y. Zhu, M. Ammar. Algorithms for Assigning Substrate Network Resources to Virtual Network Components, INFOCOM, Barcelona, Spain, 2006.

[13] J. Lu, J. Turner. Efficient Mapping of Virtual Networks onto a Shared Substrate, Technical Report WUCSE-2006-35, Washington University, 2006.

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

[15] I. Houidi, W. Louati, and D. Zeghlache, "A distributed virtual network mapping algorithm," Proceedings of IEEE ICC, 2008, pp. 5634–5640.

[16] J. Lischka and H. Karl, "A virtual network mapping algorithm based on subgraph isomorphism detection," Proceedings of the 1st ACM workshop on Virtualized infrastructure systems and architectures, Barcelona, Spain: ACM, 2009, pp. 81-88.

[17] N.M. Chowdhury, M.R. Rahman, and R. Boutaba, "Virtual network embedding with coordinated node and link mapping," IEEE INFOCOM, 2009.

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

[19] T. Asami and S. Namiki, "Energy consumption targets for network systems," Optical Communication, 2008. ECOC 2008. 34th European Conference on, 2008, pp. 1-4.

[20] S. Nedevschi, L. Popa, G. Iannaccone, S. Ratnasamy, and D. Wetherall, "Reducing network energy consumption via sleeping and rate-adaptation," Proceedings of the 5th USENIX Symposium on Networked Systems Design and Implementation, San Francisco, California: USENIX Association, 2008, pp. 323-336.

[21] M. Gupta and S. Singh, "Greening of the internet," Proceedings of the 2003 conference on Applications, technologies, architectures, and protocols for computer communications, Karlsruhe, Germany: ACM, 2003, pp. 19-26.

[22] G. Lovász, A. Fischer, and H. de Meer, "Network Virtualization and Energy Efficiency.", unpublished.

[23] Virtualization in the Core of the Network, Juniper White Paper, http://www.juniper.net/us/en/local/pdf/whitepapers/2000299-en.pdf.

[24] Router Virtualization in Service Providers, Cisco White Paper, http://www.cisco.com/en/US/solutions/collateral/ns341/ns524/ns562/ns573/white_paper_c11-512753.html.

[25] E. Rosen, Y. Rekhter. 2006. BGP/MPLS IP Virtual Private Networks (VPNs), IETF RFC 4364.

[26] XEN hypervisor - http://www.xen.org/download/.

[27] 802.1Q VLAN implementation for Linux - http://www.candelatech.com/~greear/vlan.html.

[28] XORP – Open Source Router - http://www.xorp.org/downloads.html

[29] VLC media player - Open Source Multimedia Framework and Player - http://www.videolan.org/vlc/.