Electronic Communications of the EASST
Volume 17 (2009)

Workshops der Wissenschaftlichen Konferenz Kommunikation in Verteilten Systemen 2009 (WowKiVS 2009)

Control Plane Issues in the 4WARD Network Virtualization Architecture

Roland Bless and Christoph Werle

12 pages

Guest Editors: M. Wagner, D. Hogrefe, K. Geihs, K. David
Managing Editors: Tiziana Margaria, Julia Padberg, Gabriele Taentzer
ECEASST Home Page: http://www.easst.org/eceasst/ ISSN 1863-2122

Control Plane Issues in the 4WARD Network Virtualization Architecture

Roland Bless and Christoph Werle [1]

[1]Institute of Telematics, Universität Karlsruhe (TH)
Zirkel 2, 76128 Karlsruhe
Email: bless@tm.uka.de, werle@tm.uka.de

Abstract: Network virtualization technologies offer a lot of opportunities and advantages but create also new issues that need to be solved. In this paper, we discuss control mechanisms, interfaces, and protocols required in order to allow for dynamic setup of virtual networks. Finally, we describe some runtime aspects by examining control interfaces and signaling protocols necessary for the management of virtual networks.

Keywords: Network Virtualization, Virtual Networks, Signaling, Control Plane

1 Introduction

The Internet has become a rather fragile patchwork due to many new requirements that nobody could have foreseen at the time when the Internet was designed [Han06]. The effort required to change an existing network infrastructure or transition between different architectures is usually very high, as for instance the slow migration process from IPv4 to IPv6 shows. *Network virtualization* offers a possibility to smoothly test, debug, and roll out new network architectures in parallel [FGR07]. Furthermore, it provides more flexibility in network provisioning, because virtual networks can be expanded or shrunk easier than physical networks. An infrastructure provider has the advantage to use physical network resources more efficiently, because he can multiplex multiple isolated networks on top of his infrastructure and may shift virtual nodes to different infrastructure nodes depending on the current resource situation. This also offers a new mechanism for Traffic Engineering as the migration of a virtual node also results in migration of its virtual links and thereby influences the load on physical links. Furthermore, infrastructure providers may exploit the statistical multiplexing gain of virtual links, if QoS requirements of the latter are somewhat relaxed, e.g., if they require a statistically guaranteed bandwidth only. An advantage for the operator of a virtual network is the ability of virtual networks to span multiple infrastructure provider domains without the operator having to deal with interprovider issues in his virtual network. But these advantages all come at the cost of a higher management effort due to another level of abstraction: instead of managing physical resource directly, virtual resources have to be managed on top of physical resources. Therefore, network virtualization approaches need to support efficient management and control mechanisms.

In this paper we describe control plane issues of the Network Virtualization Architecture that is currently under development in the 4WARD EU project [4WA08]. We give an example of how an IP-TV application service provider may benefit from a network virtualization concept

and describe the different roles of players and stakeholders in the Network Virtualization Architecture. Although this concrete example is using an IP-based architecture running inside the virtual network we note that the 4WARD Network Virtualization Architecture is not limited to host only IP-based network architectures.

We introduce basic terminology and the roles in network virtualization in Section 2. In Section 3, we point out the phases in a virtual network's lifecycle and discuss individual interfaces in detail in Section 4. We present some considerations with respect to the virtual node architecture in Section 5 and discuss some thoughts on the attachment of end users to virtual networks in Section 6 before describing related work in Section 7. Section 8 concludes with a summary and an outlook on future work.

2 Network Virtualization Overview

Network virtualization is a concept to create logical network resources, i.e., *virtual nodes* and *virtual links*, that form a *virtual network (VNet)* from physical resources. We call the collection of physical resources the *substrate*, which is naturally divided in *substrate nodes* and *substrate links*. Some of the substrate nodes may offer virtualization support and therefore be able to host virtual nodes, whereas some of them might not. A substrate node with virtualization support may host one or more *virtual nodes* of the same or different VNets. As shown in Figure 1, a *virtual link* is a link that connects two virtual nodes. A virtual link consists of a substrate path, i.e., a path that is composed of one or more substrate links. In general, a virtual link may consist of multiple substrate paths, which can be used to increase the capacity or reliability of the virtual link. Additionally, substrate path splitting [YYRC08] can be used to efficiently map VNets to substrate resources.

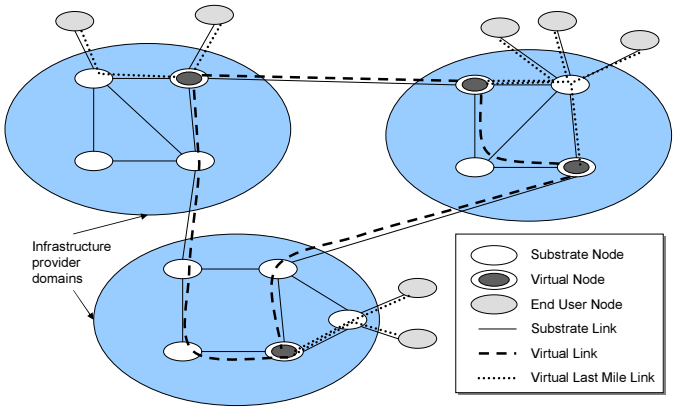

Figure 1: Overview of a virtual network topology and substrate networks

As depicted in Figure 1, a VNet may span various substrate network domains that belong to different infrastructure provider networks. Finally, *end-users* will connect to the VNet infrastructure via virtual last mile links that also consist of substrate paths. In our view end users'

devices do not belong to the VNet topology, they are rather connected as leafs to the VNet topology. Reasons for this decision will become more clear in the following, but the high dynamicity and the mobility of end users is surely one motivation to exclude them from the VNet topology description.

In the following, we start with a description of the different players in a network virtualization architecture that is currently developed within the EU-funded project 4WARD [4WA08]. The corresponding control plane provides functions to setup, control, change, and manage a VNet. A control and signaling solution must consider different roles because of the corresponding interfaces for interaction between the different parties.

2.1 Roles in Virtual Networks

Like in the Internet today, we assume that there will be multiple *infrastructure providers* (cf. Figure 2), i.e., large companies that own the infrastructure required to enable communication between different locations and which provide end users with access to their networks. In contrast to today's Internet, we additionally assume the availability of an inter-domain QoS solution in the substrate that can be used to establish links with QoS guarantees between multiple infrastructure providers. Without QoS guarantees for virtual links, any further QoS mechanisms inside the virtual network itself cannot build upon a deterministic link capacity, and consequently QoS guarantees cannot be provided inside the virtual network. Infrastructure providers may also enable the creation of virtual nodes and virtual links on top of their own resources and provide them to another party. In our architecture, this other party is the *VNet provider*. He represents an intermediate party between VNet Operators and infrastructure providers and introduces a new level of indirection. A VNet provider composes a VNet slice, i.e., a lifeless set of virtual resources—as requested by a VNet Operator—from physical resources of one or more infrastructure providers. This simplifies the VNet operator's role, because he does not need to negotiate with different infrastructure providers.

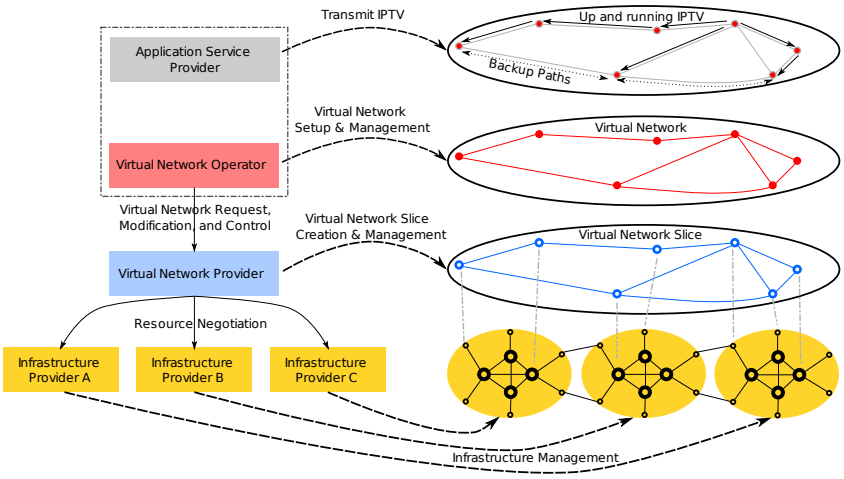

Figure 2: Relationship between roles and resources

After creation of the VNet topology, the *VNet Operator* can start to vivify his network by installing and instantiating a network architecture inside and properly configuring it. In some cases it is likely that an *application service provider* uses the VNet to provide an application specific service, which in our example is an IP-TV service. Usually, the IP-TV service provider would specify the desired network topology for the VNet operator, e.g., node locations and capacities. We note that the application service provider operates fully *inside* the VNet. One or several *content providers* may access the VNet in order to provide the necessary content, e.g., by putting TV programs and videos onto streaming servers that are connected to the VNet, too. IP-TV service customers may then attach to the VNet and use the provided IP-TV service wherever the VNet is accessible.

Due to full end-to-end control over the VNet, the VNet operator has the possibility to run protocols that support an application-specific service, e.g., in case of IP-TV an IP multicast service might be necessary and be deployed in the virtual network. In today's Internet the unclear business model and lack of deployment of inter-domain multicast are severe obstacles for wide area IP-TV providers. Today, IP-TV is already in use, but usually offered only by infrastructure providers in their own domain. An IP-TV application service provider could then use intra-domain routing protocols, like PIM-SM and PIM-DM, in the virtual network. We note that the different roles can be combined, e.g., the VNet provider and VNet operator roles may be represented by the same organisation, and in some cases the application service provider role may be combined with the VNet Operator role. A fine-grained role model, however, fosters a wide variety of business models that might not be realized otherwise.

3 The VNet Lifecycle

In this section, we start out with an overview of the VNet lifecycle and will afterwards present the interfaces we have identified in the VNet architecture so far. Therefore, we subdivide the VNet lifecycle in the steps depicted in Figure 3:

- *VNet Design:* In order to create a new VNet, the VNet Operator has to describe the required topology, resources and corresponding additional constraints, for example, QoS constraints for virtual links or geographic restrictions for virtual nodes. In our example, the VNet Operator derives such specifications from the IP-TV application service provider requirements. It is therefore necessary to estimate the amount of virtual resources required to provide the intended service, but as the VNet can be shrunk or expanded later on, this only needs to be a coarse initial presetting and can be adapted at runtime.

- *VNet Provisioning:* This description is then passed on to a VNet Provider who will construct the VNet from available physical resources. The VNet Provider's main task is the construction of a VNet as described by the VNet Operator from infrastructure providers' resources by picking a set of resources that matches the requirements [HLZ08]. Therefore, he forwards the VNet description or parts of it to one or more infrastructure providers, which reply whether they can fulfill the request. The infrastructure providers can then setup their substrate resources correspondingly and carve out the virtual resources. A spe-

cial case consists in interconnecting virtual nodes over infrastructure provider boundaries, as the VNet Provider may have to assist in this task.

- *VNet Instantiation:* If the VNet slice creation has been successful, the VNet Operator gets access to the virtual network slice. To allow the VNet Operator to enliven his share of resources he has to get access to an *Out-of-VNet Access* control interface. The functionalities offered by this interface must operate on a low level, e.g., allowing the VNet Operator to reboot the virtual machine in case of lockups and to perform installation of an operating system and to access it similar to a serial console or remote control panel. For creation of the VNet topology, i.e., a vivified VNet slice with a fully functional network architecture set up inside, or under severe failure conditions, management access is not possible from within the virtual network and therefore must be provided via an extra control plane interface, which we call Out-of-VNet Access.

- *VNet Operation:* Some modifications of a VNet, e.g., extension, shrinking, modification of QoS requirements, or tear down of the VNet, may require to contact the VNet Provider again (cf. Figure 3). Other runtime operations without VNet Provider involvment include attachment of end users, virtual node migration (usually performed transparently at infrastructure provider level) and controlled interaction with other VNets.

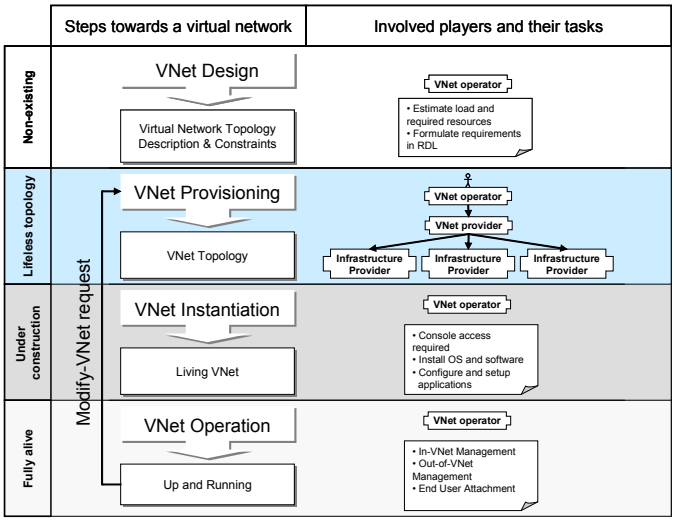

Figure 3: The VNet Lifecycle: Process Overview

4 Identification of Control Interfaces

For the following discussion on individual control interfaces, we will refer to Figure 4. This figure shows the substrate consisting of three different infrastructure providers, i.e., ISP1, ISP2,

and ISP3. On top of their resources, a VNet has been created and we will examine the single steps towards this virtual network in the following sections. In our IP-TV example, the Application Service Provider will specify the requirements of the content distribution network, e.g., node locations and link capacities.

4.1 Creation of Virtual Networks

After the VNet Operator has finished the description of its virtual network (VNet Design Phase), the description is passed on to a VNet Provider (Interface 1). This interface may be assisted by proprietary Resource Description Languages (RDLs) and toolchains that VNet Providers might offer in order to compete with each other. For the interface between VNet Providers and infrastructure providers (Interface 2) however, this possibly proprietary description will have to be mapped onto a common RDL that is used between VNet Providers and infrastructure providers. In order to keep this interface clean and to avoid that a VNet Provider has to translate the received virtual network description into multiple different RDLs, it is required to agree on a common, extensible RDL for this interface. More interesting than the actual RDL specification for this interface, however, is its signaling side. As this is a highly sensitive interface from a security and privacy aspect, it is isolated in a dedicated *Provisioning Network (PN)* that might be realized by a closed overlay network between VNet Providers and infrastructure providers. It is a closed network, because it should be accessible only by authorized VNet Providers and infrastructure providers. Furthermore, due to high availability and reliability requirements, it should provide high robustness against failures and attacks. Additionally, non-repudiation of transactions might be a desirable property.

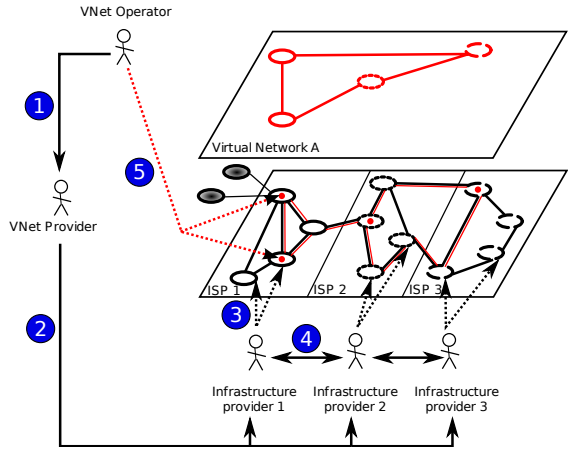

Figure 4: The VNet Lifecycle: Overview of Interfaces

Inside the PN, a VNet Provider can then negotiate resources for virtual networks with multiple infrastructure providers. After the required virtual resources have been selected by the VNet Provider, each of the involved infrastructure providers starts to configure its own physical resources correspondingly and sets up all corresponding resources for virtual nodes and links in-

Table 1: Summary of Required Control Protocols

Inter-face	Protocol Name	Who
1	VNet Provisioning Protocol	VNet Operator ↔ VNet Provider
2	VNet Resource Negotiation Protocol	VNet Provider ↔ Infrastructure Provider
3	VNet Node Setup Protocol	Infrastructure Provider ↔ Own Substrate nodes
4	VNet Link Setup Protocol	Infrastructure Providers
5	Out-of-VNet Access Protocol	VNet Operator ↔ Infrastructure Provider
6	VNet Attachment Protocol	VNet End user ↔ Infrastructure Provider, VNet End user ↔ VNet Operator

side its domain (Interface 3). As this is an interface that is likely to be completely isolated from any other traffic in the infrastructure provider's domain, e.g., by using an isolated management network, infrastructure providers are free to use their own management tools. Nevertheless these management tools must be able to put into practice all demands and constraints requested by the VNet Provider as agreed during resource negotiation.

Assuming multiple involved infrastructure providers, the so far isolated parts of the virtual networks—each hosted at a different infrastructure provider—have to be interconnected (Interface 4). We note that the setup of virtual links between different infrastructure providers requires a common signaling protocol in the substrate and, in order to support QoS for virtual links, also calls for a common inter-domain QoS solution. In case of an IP-based substrate, a path-coupled resource reservation signaling protocol (e.g., QoS NSLP [MKM08]) together with DiffServ mechanisms could serve as a basis: resources are reserved along the substrate path that interconnects two virtual nodes. An extension of the QoS signaling protocol for negotiation and exchange of VNet specific addressing information may be necessary, though. We note that it is not possible to provide sensible QoS guarantees on top of best-effort virtual links whose QoS parameters are not sufficiently predictable.

Table 1 summarizes the control interfaces and lists preliminary protocol names as well as the involved communication peers. We note that all control interfaces are sensitive and need an appropriate protection against unauthorized access. Due to space limitations we cannot discuss security issues here, but we take them into account during the design phase already.

4.2 Instantiation and Management of Virtual Networks

After the virtual network topology has been successfully created, the VNet Operator can begin with the instantiation of the virtual network. As motivated before, this requires access to the virtual nodes, which has to take place *outside of the virtual network*. For instance, the VNet operator may choose to use the same network operating system on all virtual nodes in order to simplify network management by using a homogeneous environment. Functionality of the Out-of-VNet access interface comprises low level management functions of the virtual node such as to start, to stop, to suspend, to reboot, or to install the virtual node and so on. This is an important control interface, especially if the instance running inside the virtual node cannot be contacted and managed anymore due to a failure inside the VNet, so that access from within the VNet ("In-

Vnet access") is not possible anymore. This *"Out-of-VNet access"* (Interface 5) may be provided either as direct access to the substrate node or as indirect access via a dedicated management proxy at the infrastructure provider. Currently, we assume that we a get a substrate contact address from the VNet Provider for Out-of-VNet management access to each virtual node, so direct and indirect access are possible. In case of virtual node migration and direct access this information may be updated before the virtual node is actually moved to a new substrate node, whereas migration may happen transparently to the VNet Operator and VNet Provider behind a proxy for indirect Out-of-VNet access .

Depending on the role, there are different responsibilities during this phase of the VNet Lifecycle:

- *VNet Operator* — He performs *Out-of-VNet Management* to accomplish the above listed low level management funtions and also *In-VNet Management* by using management mechanisms inside the virtual network. In case of an IP-based VNet the usual monitoring and management tools (e.g., SNMP, NetFlow, etc.) will probably be used. Monitoring of the virtual link QoS is also essential since the VNet operator must indicate any violation of the service level agreements. For instance the number of transmitted and received packets over the virtual link can be monitored and compared in order to determine its current packet loss rate. Another responsibility is the handling of end user attachment, i.e., the VNet Operator must check a user's authorization and delegate him to a suitable virtual access point.

- *VNet Provider* — He gets in the loop again if modification of the virtual network topology is required, e.g., due to adding or deleting virtual nodes. For instance, the IP-TV provider may want to increase the geographical coverage of his service and simply requests to add virtual nodes at new locations. Migration of virtual nodes between infrastructure providers also requires coordination by the virtual provider.

- *Infrastructure Provider* — He operates and manages substrate resources and optimizes physical resource usage by taking advantage of virtual resource migration. Additionally, infrastructure providers offer the aforementioned Out-of-VNet access.

After a short description of a Virtual Node architecture and touching some addressing issues in the next section, we turn our attention to the interface for end-user attachment.

5 Virtual Node Architecture

This section describes a basic node architecture for network virtualization shown in Figure 5. While describing this figure, we derive some requirements for identifiers required for signaling purposes. Figure 5 shows a substrate node with two physical network interfaces. On top of this substrate node, virtual nodes of two different virtual networks are hosted: two nodes of VNet #1 (sun symbol) and one node of VNet #2 (moon symbol). For adressing purposes in control and data planes we need a unique *VNet-Identifier (VNet-ID)*. It is required for multiple reasons:

1. *End User Attachment:* In order to attach to the desired virtual networks from any place, a globally unique identifier for virtual networks is useful, e.g., virtual access nodes of the corresponding VNet can be looked up and discovered (cf. Section 6).

2. *Accounting & Billing:* A globally unique identifier eases assignment of resource usage if multiple infrastructure providers are providing resources to a virtual network.

3. *Uniqueness across multiple infrastructure providers:* Since VNets may span different infrastructure providers, the VNet-ID also should be globally unique, e.g., for accounting purposes. One option currently under consideration is that a VNet Provider generates the VNet-ID as a cryptographic ID, e.g., as hash value of a generated public key. This could be used for improving the security: in case a VNet Provider wants to modify a VNet configuration, infrastructure providers can verify that the VNet Provider possesses the corresponding private key that belongs to the aforementioned public key. Furthermore, the VNet Provider can supply credentials to the VNet Operator and the involved infrastructure providers, so that only authorized access to control functions is possible.

If multiple virtual nodes of the same virtual network are hosted on the same substrate node—as is the case for VNet #1—it is necessary to differentiate between the virtual nodes, e.g., when setting up virtual links. Therefore, a further identifier is required, the *VNode-ID*. The VNode-ID's scope is only valid with regard to a certain VNet-ID, i.e., it is possible that VNet #1 and VNet #2 consist of virtual nodes that are assigned the same VNode-ID (Figure 5, VNode #a) as they can be differentiated by the VNet-ID. As a VNode may be connected via several virtual links to other VNodes, different virtual links may exist inside a VNode that are accessible as *virtual interface (VIf)* within a VNode. A virtual interface is thus identified by its *VIf-ID* that is unique inside the VNode. Figure 5 shows some important components:

- The *Substrate Node Control* consisting of VNode Control and VLink Control allows for setup and modification of virtual node slices and virtual links and must therefore only be accessible by the infrastructure provider.

- The *(De-)Multiplexing and QoS Mechanisms* component is responsible for demultiplexing of multiple incoming / outgoing virtual links via one substrate link.

- The *Hypervisor / Resource Control* is responsible for actual creation of virtual nodes and manages the resources assigned to them.

- *Out-of-VNet Management Access* allows VNet Operators to access each of their virtual nodes in case of initial setup, misconfiguration, or failures inside the virtual network and permits reboot, serial console access, and further management functionalities. VNet Operators are allowed to access their virtual nodes after they have been properly authenticated and authorized. The access to this interface may be proxied by a management node of the infrastructure provider. From a security perspective, this interface is highly critical and requires extremely careful engineering.

With respect to the data plane, it is not required that the new identifiers are literally carried in data packets since there could be link-specific mapping techniques using available multiplexing mechanisms, e.g., 802.1Q VLAN-tags. In analogy we want to denote such link-specific

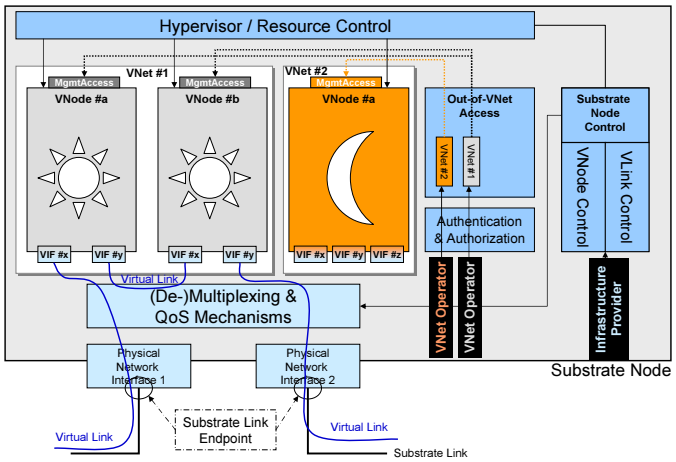

Figure 5: A substrate node hosting different virtual nodes

identifiers for VNets as *VNet-Tags* abstracting from concrete mechanisms. A VNet-Tag identifies a virtual link of a VNet in substrate link specific context, e.g., a virtual link of a certain virtual network might be mapped to Ethernet VLAN tag 42 in the substrate. This requires the presence of local mapping at the link ends. In absence of any substrate mechanism supporting (de-)multiplexing it may be required to carry an explicit shim header that carries a VNet-Tag in order to allow for proper (de-)multiplexing of virtual links across a shared substrate link.

6 End User Attachment to Virtual Networks

If an end-user (IP-TV customer) attaches to a substrate network, he probably wants to get access to the IP-TV VNet (or even more VNets) to which he subscribed. Rather than having to explicitly "dial-in" into the virtual network by establishing a tunnel connection to a VNet concentrator (analogous to a VPN concentrator), connectivity should be established automatically if possible. That is, the end-user's node will automatically discover virtual access points of the VNets it wants to connect to. We assume that there is a *Virtual Network Attachment Protocol* which is used to contact the VNet Operator of the corresponding VNet for initial authentication and authorization. We also assume that the end-user is probably attached to a domain that either doesn't support network virtualization or doesn't have any virtual nodes belonging to the specific VNet. Thus, the substrate access node may have to only support some generic backend authorization protocol. Currently, it is open whether the substrate access node locates the corresponding authorization node of the VNet Operator for the requested VNet or whether the request carries this information already. The response contains the substrate address of the VNet Access Node that can be used for the subsequent setup of the point-to-point virtual last mile link. In a simple case one could think of tunnel creation, in more complex cases the virtual last mile link may offer guarantees and a QoS reservation is required. The VNet end-user could thus access his subscribed IP-TV service from nearly any location.

7 Related Work

Network virtualization as such is not a completely new concept. For instance, the Genesis Kernel [KCC+01] is a distributed operating system allowing for creation of VNets that are aimed towards experimenting with new network architectures. Consequently, the Genesis Kernel is not aimed towards operating on a global scale in a setting with multiple competing infrastructure providers. The X-Bone's global approach to network virtualization [TWP+05] allows for virtualization in presence of competing infrastructure providers but also aims for virtualizing testbeds and does not consider the attachment of end users to virtual networks. Moreover, it is restricted, like most other existing approaches, to create VNets that run IP inside. Though we used IP-TV as an example in this paper, the VNet framework and architecture aims to support different and new network architectures.

Lots of other network virtualization approaches are briefly summarized in [CB08]. Most approaches, however, do not consider an architectural framework and control interfaces to support it as networking paradigm on a global scale. A recently published paper [ZZRR08] describes a connectivity architecture that is used to build virtual networks. The approach differentiates between infrastructure providers, connectivity providers and service providers. The approach presented here, differentiates also between a VNet provider and a VNet operator, which collapse in the single role of a connectivity provider in [ZZRR08]. In our view, the VNet provider performs mainly resource brokerage between different infrastructure providers (thereby facilitating a VNet operators work), but he is usually not in the control loop during the operation of a VNet. We also allow that several roles may be combined into a single entity, e.g., service provider and VNet operator or VNet provider and VNet operator, and so on. The finer distinction of the roles also allows for a variety of different business models and allows for a better modularisation and separation of control interfaces.

8 Conclusion and Outlook on Future Work

We presented a network virtualization architecture and its related process, with IP-TV as an application service example. In contrast to other approaches the architecture considers four involved players: the infrastructure provider, the VNet provider, the VNet Operator, and finally the VNet end-user. We motivated the need for and functionality of some control interfaces. We are currently defining signaling protocols and the involved components in more detail so that we can start to work on a prototypical implementation of the approach for further evaluation.

Acknowledgments

The authors would like to thank all members of the 4WARD project, especially those from the VNet work package, who discussed issues in the context of the paper.

Bibliography

[4WA08] T. 4WARD Consortium. The FP7 4WARD Project. Dec. 2008.
http://www.4ward-project.eu/

[CB08] N. M. K. Chowdhury, R. Boutaba. A Survey of Network Virtualization. Technical report, David R. Cheriton School of Computer Science, University of Waterloo, Oct. 2008. Technical Report: CS-2008-25.
http://www.cs.uwaterloo.ca/research/tr/2008/CS-2008-25.pdf

[FGR07] N. Feamster, L. Gao, J. Rexford. How to lease the internet in your spare time. *SIGCOMM Computer Communications Review* 37(1):61–64, 2007.
doi:http://doi.acm.org/10.1145/1198255.1198265

[Han06] M. Handley. Why The Internet Only Just Works. *BT Technology Journal* 24(3), July 2006.
http://www.cs.ucl.ac.uk/staff/M.Handley/papers/only-just-works.pdf

[HLZ08] I. Houidi, W. Louati, D. Zeghlache. A Distributed Virtual Network Mapping Algorithm. *IEEE International Conference on Communications (ICC '08)*, pp. 5634–5640, May 2008.
doi:10.1109/ICC.2008.1056

[KCC+01] M. Kounavis, A. Campbell, S. Chou, F. Modoux, J. Vicente, H. Zhuang. The Genesis Kernel: a programming system for spawning network architectures. *IEEE Journal on Selected Areas in Communications* 19(3):511–526, Mar. 2001.
doi:10.1109/49.917711

[MKM08] J. Manner, G. Karagiannis, A. McDonald. NSLP for Quality-of-Service Signaling. Feb. 2008. Internet Draft draft-ietf-nsis-qos-nslp-16 (Work in progress).
http://tools.ietf.org/html/draft-ietf-nsis-qos-nslp

[TWP+05] J. Touch, Y.-S. Wang, V. Pingali, L. Eggert, R. Zhou, G. Finn. A global X-bone for network experiments. *Tridentcom 2005*, pp. 194–203, Feb. 2005.
doi:10.1109/TRIDNT.2005.2

[YYRC08] M. Yu, Y. Yi, J. Rexford, M. Chiang. Rethinking virtual network embedding: substrate support for path splitting and migration. *SIGCOMM Comput. Commun. Rev.* 38(2):17–29, 2008.
doi:http://doi.acm.org/10.1145/1355734.1355737

[ZZRR08] Y. Zhu, R. Zhang-Shen, S. Rangarajan, J. Rexford. Cabernet: Connectivity architecture for better network services. In *Proceedings of Workshop on Rearchitecting the Internet (ReArch 08)*. ACM SigComm, Madrid, Dec. 2008.