Position Paper: Secure Virtual Network Embedding

Andreas Fischer and Hermann de Meer,
University of Passau, Faculty of Computer Science and Mathematics
94032 Passau, Germany,
{andreas.fischer, demeer}@uni-passau.de

Abstract. Network virtualization has been recognized as an important technique to overcome the perceived ossification of the current Internet. Several variations of network virtualization have already been discussed in the literature. These approaches use virtualization to partition and/or combine physical network resources into virtual network resources. An actual deployment of virtual networks then requires the network operator to perform a mapping of virtual resources onto physical resources. The question of how this mapping can be performed in an optimal way is commonly known as the Virtual Network Embedding (VNE) problem. Several algorithms to solve this problem have been proposed already. These algorithms, however, focus on optimizing the use of resources with regard to performance. Security constraints to the VNE problem have not been investigated in depth, so far.

Introduction

The current Internet is perceived as inflexible and difficult to change. Overwhelmed by its own success, it now poses a major hindrance to innovations concerning the network itself. This has become painfully apparent with the delayed introduction of IPv6 – but also other technologies, like DiffServ or IntServ have stumbled across the difficulty to change an integral part of the network. New paradigms are currently being investigated as a solution to this dilemma. One of these paradigms is network virtualization [1,4,5]. Taking on the recent hype on virtualization, network virtualization tries to apply virtualization methods to network elements – in particular nodes and links of a network.

Virtualization in this sense is an abstraction of the available hardware, allowing an operator to either partition or combine existing hardware to create a virtual entity that is easier to manage. In the area of networks, typically nodes are partitioned (with several virtual nodes being hosted on one physical node) and links are combined (with one virtual link being composed of a path in the substrate network). Applying these concepts, entire virtual networks can be created on top of a substrate infrastructure. These virtual networks can then be managed in a more convenient way than the underlying physical hardware. For example, the topology of a virtual network can be tailored to the needs of the services within a virtual network. It is even possible to dynamically change the topology, e.g. by introducing new nodes or links. Moreover, different networks can be conveniently separated from each other, by confining each network into its own virtual environment. This allows, for example, to test drive new network protocols on the same hardware as the production network. It is through this flexibility, that network virtualization is expected to be a major driver for a Future Internet. Indeed, several concepts to implement network virtualization have already been investigated [6]. Consequently, network virtualization is already widely used in Future Internet test-bed facilities like VINI [12], GENI [13], or the German G-Lab [11].

Virtual Network Embedding

When discussing virtual networks, the question arises, how these networks can be mapped in an optimal way onto a physical infrastructure. Every virtual resource has to be backed by physical hardware in order to operate properly. This requires algorithms that will take the demands imposed by a virtual network and calculate an appropriate mapping of the individual virtual resources to physical ones. A typical example is to associate CPU power with nodes and bandwidth with links. In a virtual network then each virtual node has a CPU demand, and each virtual link has a bandwidth demand. Likewise, each physical node has a certain CPU capacity and each physical link has a certain bandwidth capacity. The total number of virtual networks that can be mapped is thus limited by the actual physical resources provided by the substrate network.

Figure 1 shows two virtual networks that are mapped onto a substrate network. It can be seen, that a physical node can host more than one virtual node. This can be realized with system virtualization, for example. Moreover, the topology of the virtual networks does not have to follow the topology of the substrate network. Virtual links are mapped to a path in the substrate network (as indicated by the highlighted virtual link in Virtual Network 1). A technique to implement this behaviour would be tunnels (e.g. IP in IP or VPN).

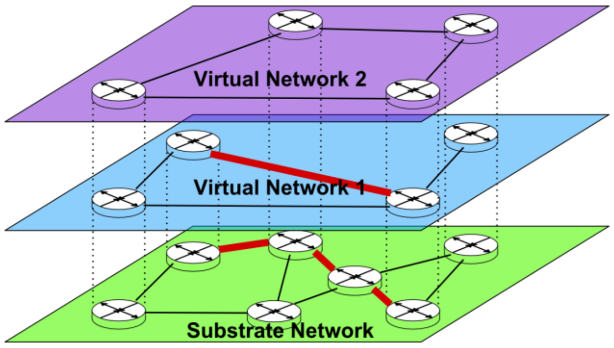

Figure 1: *A substrate network hosting two virtual networks*

Optimality, here, can be considered according to different factors. An obvious way to optimize is to minimize the individual load of each physical resource – that is to distribute the load across the network in a fair way. Another way to optimize the mapping would be to maximize the amount of virtual networks that can be hosted on a substrate network. Several variations of these optimization problems are known to be NP hard. This leads to a third optimization problem: run time of the algorithm. In particular, if requests can not be computed ahead of time – that is, if requests for new virtual networks and/or requests for modification of existing virtual networks can arrive dynamically, run time can become a major issue. Solving this mapping problem algorithmically – either precisely or by approximation – is known as Virtual Network Embedding (VNE). Several VNE algorithms have already been proposed in the literature [2,7,8,9,10]. All of these algorithms focus on the distribution of CPU and bandwidth among the virtual networks, with different target functions for the optimization.

Security problems in Network Virtualization

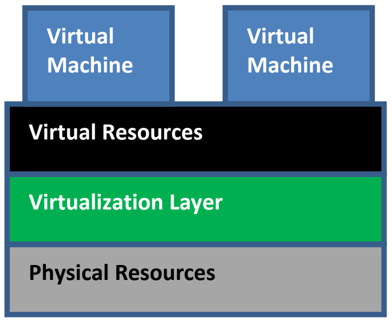

Figure 2: *The additional layers introduced by virtualization*

Network virtualization is expected to solve several problems related to the flexibility of the current network architecture. Alas, this does not come without cost. The increased flexibility comes with an increased complexity. The abstraction provided by virtualization mechanisms introduces several additional layers (see Fig. 2). This can pose a significant security risk, as additional attack vectors become available to adversaries. Note, that the source of an attack has to be an active element of the network (i.e. a network node). Problems can therefore be categorized in three areas:

1. A physical host attacking one of its virtual machines;
2. A virtual machine attacking its physical host;
3. Or a virtual machine attacking another virtual machine.

In the first case, the virtual machine has no option to defend itself. Since all computation is ultimately carried out by hardware of the physical host, the physical host is able to supervise and/or change all aspects of the virtual machine – including the result of the (potential) question by the virtual machine, whether the physical host is malicious or not. It is therefore imperative for the security of the virtual machine to be hosted on a trustworthy host.

The second case is commonly known as virtual machine outbreak – an escape from the rigid confinement created by the virtualization process. This can be triggered, for example, by bugs in the employed virtualization solution. By gaining control of

the physical host, the attacker can then exercise full control over other virtual machines. Moreover, the attacker is also able to influence the network in a negative way – e.g. by launching a Denial of Service attack. Thus, the operator of a physical machine will prefer to host virtual machines that are trustworthy to a certain degree.
The third case can be a problem, even if virtual machines are confined properly and no direct contact can be made. The fact, that virtual machines share the same hardware, creates the possibility for side-channel attacks [14]. These attacks can then be used to communicate among virtual machines, bypassing any control instances, like firewalls. Moreover, clever application of side-channel techniques may allow an attacker to spy on other virtual machines that are co-hosted on the same physical host. In order to prevent this issue, only virtual machines that can trust each other should be hosted on the same hardware.

Secure Virtual Network Embedding

As seen in the previous section, there are several security issues to consider when employing network virtualization. Solving these issues by installing additional software in the virtual machines or in the physical host is either difficult or impossible. Although virus scanners and intrusion detection systems can try to identify potential attacks, these measures are not guaranteed to be successful. An appropriate solution to these problems will, therefore, take into account the mapping of virtual resources onto physical machines. By mapping virtual resources in a security-aware way, the risk exposure of both the virtual machines and the physical machines can be minimized. A first step is to assign a security level and a security demand to both virtual and physical resources. In line with the three problem categories mentioned in the previous section, three types of additional constraints have to be considered:

- A virtual resource should not be mapped on physical resources that have a lower security level than the security demand of the virtual resource.
- A physical resource should not be used to host virtual resources that have a lower security level than the security demand of the physical resource
- A virtual resource should not be co-hosted on the same physical resource together with another virtual resource having a lower security level than the security demand of the first resource.

More formally, let *SC* be a totally ordered set of security categories. Let pl_i, $pd_i \in SC$, denote the security level and security demand of physical resource *i*. Likewise, let vl_j, $vd_j \in SC$, denote the security level and security demand of virtual resource *j*. Then, the following additional constraints have to be considered for a secure VNE:

1. $vd_j \leq pl_i$, for all virtual resources *j* mapped onto physical resource *i*.
2. $pd_i \leq vl_j$, for all virtual resources *j* mapped onto physical resource *i*.
3. $vd_j \leq vl_k$, for all virtual resources *j*,*k* mapped onto the same physical resource *i*.

Implementation of these constraints in VNE algorithms is necessary to capture the trust relationships between different parties involved in the operation of a virtualized network environment. In a multi-vendor network virtualization scenario, these constraints can, therefore, not be neglected. However, the adaption of VNE algorithms to incorporate these constraints in an appropriate manner is not without effect. The original optimality goals of modified algorithms (e.g. low resource stress) are likely to be impacted when taking security constraints into account. Moreover, the run-time of the algorithm will increase. Some algorithms may perform better than others under these additional constraints. It is, therefore, necessary to compare modified VNE algorithms on a common ground, in order to evaluate the various effects of added security constraints.

Conclusion

Network virtualization is an important concept for the Future Internet. However, being the foundation of an increasingly critical infrastructure, this paradigm must strive to ensure a certain level of security. A naive mapping of virtual resources to physical resources, only considering performance issues, can lead to severe security problems – in particular if multiple, potentially untrustworthy, players are involved. Unfortunately, algorithms solving the VNE problem have, so far, neglected security as a parameter. It is therefore necessary to investigate the performance of existing algorithms, when operating under those additional constraints.

Acknowledgements

The research leading to these results has received funding from the European Community's Seventh Framework Programme ([FP7/2007-2013] [FP7/2007-2011]) in the context of the ResumeNet and EuroNF projects (grant agreement no. 224619 and 216366, respectively).

References

1. A. Berl, A. Fischer, H. de Meer. Virtualisierung im Future Internet - Virtualisierungsmethoden und Anwendungen. Informatik-Spektrum 33(2):186–194, Apr. 2010.

2. J. F. Botero, X. Hesselbach, A. Fischer, H. De Meer. Optimal Mapping of Virtual Networks with Hidden Hops. Telecommunication Systems: Modelling, Analysis, Design and Management. Accepted - To be published, 2013.

3. A. Fischer, J. F. Botero, M. Duelli, D. Schlosser, X. Hesselbach, H. De Meer. ALEVIN - A Framework to Develop, Compare, and Analyze Virtual Network Embedding Algorithms. Electronic Communications of the EASST, 37. Pp. 1 - 12. 2011.

4. Th. Anderson, L. Peterson, S. Shenker, J. Turner. Overcoming the Internet Impasse through Virtualization. Computer 38, 4. Pp. 34-41. Apr. 2005.

5. N. Feamster, L. Gao, J. Rexford. How to lease the internet in your spare time. SIGCOMM Computer Communications Rev. 37(1):61–64, 2007.

6. N. M. K. Chowdhury, R. Boutaba, A survey of network virtualization. Computer Networks, 54. Pp. 862 - 876, 2010

7. N. M. M. K. Chowdhury, M. R. Rahman, R. Boutaba. Virtual Network Embedding with Coordinated Node and Link Mapping. In Proc. IEEE INFOCOM. Apr. 2009.

8. J. Lischka, H. Karl. A virtual network mapping algorithm based on subgraph isomorphism detection. In Proceedings of the 1st ACM workshop on Virtualized infrastructure systems and architectures. Pp. 81–88. New York, USA, Aug. 2009.

9. Y. Zhu, M. Ammar. Algorithms for assigning substrate network resources to virtual network components. In Proc. IEEE INFOCOM. Pp. 2812–2823. Apr. 2006.

10. M. Yu, Y. Yi, J. Rexford, M. Chiang. Rethinking Virtual Network Embedding: Substrate Support for Path Splitting and Migration. ACM SIGCOMM CCR 38(2):17–29, Apr. 2008.

11. D. Schwerdel, D. Günther, R. Henjes, B. Reuther, P. Müller. German-Lab Experimental Facility. Future Internet Symposium (FIS) 2010, 9 2010.

12. A. Bavier, N. Feamster, M. Huang, L. Peterson, J. Rexford. In VINI veritas: realistic and controlled network experimentation. In SIGCOMM ’06: Proc. of the 2006 conference on Applications, technologies, architectures, and protocols for computer communications. Pp. 3–14. ACM, New York, NY, USA, 2006.

13. GENI - Global Environment for Networking Innovations. http://www.geni.net. Accessed 01. June 2011.

14. T. Ristenpart, E. Tromer, H. Shacham, S. Savage. Hey, you, get off of my cloud: exploring information leakage in third-party compute clouds. CCS '09: Proc. of the 16th ACM conference on Computer and communications security, ACM. Pp. 199-212. 2009.