Network Virtualization in Future Home Environments

Andreas Berl[1], Roman Weidlich[2], Michael Schrank[1], Helmut Hlavacs[2], and Hermann de Meer[1]

[1] Computer Networks and Communications, University of Passau, Germany
[2] Institute of Distributed and Multimedia Systems, University of Vienna, Austria

Abstract. Home environments have a great potential of resource sharing and energy saving. More and more home computers are running on an always-on basis (e.g. media-centers or file-sharing clients). Such home environments have not been sufficiently analyzed regarding their energy-efficient operation, yet. This paper discusses network virtualization methods that are needed in future home environments to enable the energy-efficient cooperation of home networks. End-users share their available hardware resources (e.g. CPU, disk, or network resources) with other users in an energy-efficient and balanced way. To achieve such an envisioned future home environment, an architecture is suggested that combines different virtualization methods. In this paper, virtualization related requirements of the suggested architecture are discussed in detail. Network virtualization methods and concepts are compared to each other with respect to their usability in the architecture. In addition, initial virtualization approaches are simulated and evaluated with regard to benefits and complexity in the suggested architecture.

Key words: Home networks, energy efficiency, resource sharing, virtualization, peer-to-peer

1 Introduction

Increased costs of energy and the desire to reduce CO_2 emissions make energy-efficient computing a more and more important topic. Koomey [1] reports a doubling of energy consumption from 2000 to 2005 of volume, mid-range, and high-end servers in the U.S. and worldwide. Although this is related to data centers, a similar tendency can be expected for computers in home environments. End devices in the home are contributing to a large portion of the electricity consumption growth according to a 2006 survey commissioned by the EU [2].

Following energy-saving concepts of data-centers, an energy-efficient Virtual Home Environment (VHE) architecture is suggested and evaluated in [3–6] and presented in Section 3. Energy wastage in data centers is mainly caused by underutilized hardware. To increase energy-efficiency, services can be virtualized and consolidated (several services run on the same hardware). The VHE architecture realizes this kind of consolidation in home networks. Hardware resources

of end-hosts (e.g. CPU cycles, disk space, or network capacity) are virtualized and shared between users in an energy efficient way. *Always-on* services of users (e.g. media-servers or file-sharing applications) are consolidated on end-hosts and unused computers are turned off (or hibernated) to save energy. Although this concept seems to be very similar to energy-saving concepts of data centers, there are severe differences. Services in data centers are located in controlled closed environments with high bandwidth networks. Distributed home environments have no central management and require different virtualization and management methods to share hardware resources and energy in a balanced way. To enable the envisioned hardware resource sharing within the VHE architecture, several virtualization related requirements have to be fulfilled:

- *Resource availability*: Provision of idle hardware resources in separate runtime environments;
- *Home network interconnection*: Addressing and locating of participating home networks;
- *Resource mediation*: Addressing and locating of idle hardware resources;
- *Resource allocation*: Distributed management of state and resource information;

These virtualization related requirements and possible solutions for it are discussed in detail in this paper. Different kinds of virtualization methods are needed to realize the envisioned energy-efficient resource sharing among end-users. Especially network virtualization approaches are analyzed in this paper and virtualization approaches are suggested that meet the defined requirements. Furthermore, network virtualization approaches are evaluated by simulation. Initial simulations and evaluations of the suggested VHE architecture have already been shown in [3–5]. In this paper, the simulation is extended by network virtualization. Different approaches are described and evaluated.

The remainder of this paper is structured as follows: In Section 2 the related work is discussed. Section 3 introduces the energy-efficient VHE architecture. In Section 4 virtualization methods are discussed in detail, which can be used to realize the suggested VHE architecture. Section 5 describes the overlays that are implemented in the simulation of VHE and evaluates their benefits and overheads. Section 6 concludes this paper.

2 Related Work

The 3rd Generation Partnership Project (3GPP) [7] describes a VHE as a concept for personal service environment portability across network boundaries and between terminals. Users are consistently presented with the same personalized features, interface customizations and services in whatever network and whatever terminal, whereever the user may be located. Further the Open Service Access (OSA) framework for separating network and service layers was proposed by 3GPP. The OSA specified in conjunction with the Parlay Application Programming Interfaces the base technology that was applied in the project VESPER

[8, 9]. The project aimed to define, demonstrate, and promote a service architecture for provision of VHE across a multi-provider, heterogeneous network and system infrastructure. The European Institute for Research and Strategic Studies in Telecommunications[3] (Eurescom) [10] described a VHE as an environment enabling users to receive customized and personalized services, regardless of location, access network or terminal type in a way that users will not see a difference in using services at home or while roaming in other networks. Similarly for Liotta et al. [11] the VHE concept pursues the idea of service universality, which allows users to transparently access services anytime, anywhere, with any type of terminal. This concept allows users to be consistently presented with the same personalized features and preferences, regardless of the context. Nakajima et al. [12] proposed a virtual overlay network for integrating networked home appliances while also considering media streaming and disk sharing. All of this work done in the field of VHE assumes external providers for operating services inside a home. In contrast, the VHE proposed in this paper realizes a home centric view on the network, where virtualization is the clue to aggregate and consolidate distributed hardware resources. This understanding of a virtualization based VHE architecture was already introduced in [3] and deepened with more simulation results in [5]. An economic model for fostering fair resource sharing was presented in [4]. The investigations were extended in [6] by the creation of a prototype for task virtualization for sending virtual machines between homes designed for minimal resource usage.

Future Internet Platforms like PlanetLab [13] envision open platforms for distributed end-to-end applications, similar to VHE. Network resources and hardware resources of end-hosts that are located all over the world are virtualized. A user is provided a *slice* that consists of hundreds of shells, one for each end-host. Comparable approaches towards end-to-end virtualization are done by other projects (e.g. GENI[4] or VINI [14]). These kinds of network virtualization are comparable to the VHE architecture with respect to virtualization. However, the focus of these approaches is on providing Future Internet environments, rather than sharing hardware resources among end users.

3 The VHE Architecture

The VHE architecture approach as it was introduced in [3–6] enables the energy-efficient sharing of hardware resources amongst home networks. The main goal of the architecture is to achieve a consolidation of load (e.g. in terms of bandwidth consumption, CPU usage or disk space). Especially the load generated by always-on applications is considered. The number of applications that requires always-on hardware (e.g. media-server or file-sharing client) in home networks is growing fast, leading to a high number of computers running on a 24/7 basis. Similarly to approaches in data centers, the overall load is shifted to a small number of computers, in order to relieve others. Unloaded computers can be hibernated

[3] http://www.eurescom.de
[4] http://www.geni.net

(or turned off) to save energy. To achieve this, a possibly large number of home networks is interconnected to share their resources.

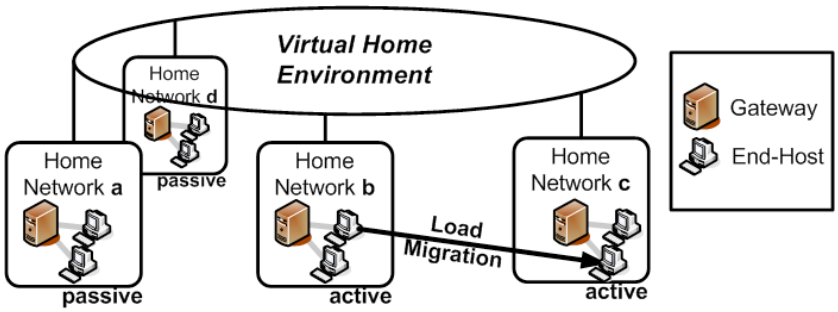

Fig. 1. VHE architecture

Consolidation of load is envisioned as follows: A user starts a task (e.g. a file-sharing client) locally on his computer. The VHE environment discovers potential of energy savings. It moves the task to another computer (probably in another home network) that is already running and has enough hardware resources left to process the task. The local computer can be turned off to save energy. When the task is finished, the local computer is turned on and the result is sent back. In this way only a small portion always-on computers is needed. A home network usually consists of a small number of computers, e.g. PCs, home servers, or laptops and a *gateway* to the Internet. In the VHE architecture, a gateway is a Linux-based diskless always-on computer that has small energy needs (e.g. an AVM FritzBox[5]). The gateway maintains a permanent entry to the interconnected home networks and represents its home network in the VHE. The interconnected home networks form a distributed pool of virtual resources, and the architecture uses a distributed management to allocate resources to home networks dynamically. To allow energy saving, a distinction is made between *active* and *passive* home networks (contributing and not contributing home networks). A home network is called active if it contains at least one computer which is turned on. In a passive home network only the gateway is online and other hardware is hibernated. The VHE architecture is illustrated in Figure 1. In this example four home networks are interconnected, two active and two passive homes. In the figure load is migrated from an end-host in the active home network b to an end-host in the active home network c. The end-host in home network b can be hibernated or turned off after the migration process. If no further computer is turned on in home network b, it can change its status to passive.

[5] http://www.avm.de/en/Produkte/FRITZBox/index.html

4 Virtualization Methods in the VHE architecture

VHE combines two different virtualization approaches to enable an energy-efficient resource sharing among home networks, *virtualization of host resources* and *network virtualization*.

4.1 Virtualization of Host Resources

To enable hardware resource sharing (e.g. CPU cycles, bandwidth or disk space) among end-hosts in home networks, it is necessary to make idle resources available for processes (guest applications) of other end-hosts (*resource availability*, see Section 1). A runtime environment is needed that processes guest applications of other users. This runtime environment has to be flexible enough to enable the processing of a wide variety of guest applications. The guest application might come from a different *Operating Systems (OS)* (e.g. Windows, MAC, or Linux) or from a different computer architecture (e.g. x86 or PowerPC). This flexible runtime environment has also to deal with privacy and security issues. On the one hand, guest applications are sent to unknown hosts within the VHE architecture. The guest applications might come together with private data (e.g. a movie that needs to be encoded). The owner of the guest application wants it to be separated as much as possible from the user environment of the host. On the other hand, a user that hosts a guest application wants his machine to be separated as much as possible from the guest application.

Nowadays, the approach of *system virtualization* is successfully used to consolidate services in data centers. Several services can run separately on top of a single hardware, saving hardware costs, space, and energy. In [6] system virtualization based on QEMU [15] has been used as initial VHE solution to virtualize idle resources of hosts. In system virtualization, a *Virtual Machine (VM)* is created, i.e. a full machine is virtualized, consisting of virtual CPUs, virtual memory, virtual hard disk, virtual Network Interface Card, etc. A VM is a perfect recreation of a real machine in such a way that an OS can be installed on it without being aware of the resource virtualization. Typical examples for system virtualization software are, e.g., XEN [16] or VMWare ESX Server[6]). In addition to providing a VM, QEMU emulates the CPU within the VM, based on dynamic binary translation [15]. This kind of emulation enables QEMU based VMs to migrate between different architectures (e.g. x86 or Mac) and makes QEMU a very flexible choice for VHE. First results with QEMU-based virtualization are described in [6].

4.2 Network Virtualization

Home networks that are a part of the VHE architecture have to be interconnected. Participating homes have to be addressable for other homes in order

[6] http://www.vmware.com/products/vi/esx

to enable communication within the connected VHE (*home network interconnection*, see Section 1). In the VHE architecture no central VHE provider is intended. Therefore, the addressing of the home networks has to be solved in a distributed way to make participating home networks accessible. Furthermore, QEMU-based virtualization makes idle resources on hosts available in separated runtime environments, however they are not yet accessible for other users. A mediation of available hardware resources has to established (*resource mediation*, see Section 1). Idle resources have to be discovered within the VHE network and they have to be made addressable to enable the allocation of idle resources to other participants. Another important requirement of the VHE architecture is the distributed management of resources (*resource allocation*, see Section 1). No central architectural element is available in VHE that manages the balanced cooperation of home networks and the access to available resources. This cooperation has to be achieved in a distributed way. Energy-efficient resource sharing has a number of constraints that have to be considered. Examples are fair distribution of energy consumption or the provision of a sufficient quality of service to users. A cost model to target such constraints in VHE is discussed in [4]. The distributed management has to be aware of the different states of the home networks (active, passive) and the resources that are available at a certain point of time. In addition, guest applications might have special needs in terms of hardware resources that have to be considered when resources are allocated.

Such requirements of the VHE architecture can be met by network virtualization methods [17, 18]. Two kinds of virtualized networks are widely used today: *Virtual Local Area Networks (VLANs)* and *Virtual Private Networks (VPNs)*. VLANs like IEEE 802.1Q[7] operate mainly on the link layer, subdividing a switched *Local Area Network* into several distinct groups either by assigning the different ports of a switch to different VLANs or by tagging link layer frames with VLAN identifiers and then routing accordingly. VPNs like IPSec[8], on the other hand, establish a network layer tunnel to either connect two networks (*site-to-site*), one network and a host (*site-to-end*) or two hosts (*end-to-end*) with an encrypted and/or authenticated channel over the Internet. However, these kinds of virtualization methods target mainly the sharing of links among users and are not sufficient for the VHE approach.

Besides the virtualization of links, also the virtualization of routers has been investigated in several approaches. In [19] system virtualization (e.g. based on XEN [16] or VMWare ESX Server)) is applied to routers to create virtualized networks with special features. In [20, 21] performance challenges are identified that have to be tackled when virtual routers are based on XEN. Other forms of router virtualization are already available in commercial products[9]. However, virtualization of routers does not solve the network virtualization issues of the VHE architecture. Such solutions mainly allow the concurrent usage of network

[7] http://standards.ieee.org/getieee802/download/802.1Q-2003.pdf
[8] http://tools.ietf.org/html/rfc4301
[9] http://www.cisco.com/en/US/docs/ios_xr_sw/iosxr_r3.2/interfaces/command/reference/hr32lr.html

infrastructure. In the VHE, a mediation of available hardware resources and their distributed management is needed within the VHE.

A further approach towards network virtualization are *peer-to-peer* (P2P) overlays [22]. In this approach logical links are defined on top of a physical infrastructure. A single logical hop in the overlay can be mapped to several physical hops in the network. P2P networks are mainly classified according to their architecture and their algorithmic features. In *pure P2P* overlays (e.g. Chord [23]) all peers are assumed to be equal. In *hybrid P2P* overlays some peers are distinguished from other peers, i.e. some peers have different capabilities than others (e.g. eDonkey [24]). P2P overlays are denoted to be *unstructured* if the algorithms establish overlay links which do not follow a regular connectivity pattern. In contrast, P2P overlays are said to be *structured* if a generic but predefined organization scheme (e.g. a ring) of the overlay exists. In contrast to the previously mentioned network virtualization approaches, P2P overlays do not only virtualize links and nodes, but they solve three main VHE issues: Home network interconnection, resource mediation, and resource allocation. P2P networks establish an addressing scheme within the overlay that enables the addressing of peers as well as the addressing of available resources. In P2P file-sharing networks (e.g. eDonkey), files can be discovered and addressed, whereas in P2P VoIP applications like Skype[10] users are locatable and addressable. This solves the problem of home network interconnection as well as the resource mediation problem. Also a management with respect to resource allocation is provided in such P2P overlays. eDonkey, e.g. establishes a complex tit-for-tat principle to enable a fair resource sharing and allows the distributed download of files from different sources concurrently. There are several P2P overlays available that can be used within the VHE architecture. Solutions have to be scalable (with a high number of home networks) and they need to be lightweight to operate on the always-on gateways (see Section 3). Pastry [25], a structured pure P2P overlay, has been used as initial network virtualization technology in [6]. Pastry is scalable and available as open source platform. Although Pastry was successfully used in simplified initial tests, it has its shortcomings. First, structured overlays like Pastry are more vulnerable to high churn-rates[11] than unstructured networks [22]. However, end-hosts might show a very dynamic behavior in the VHE architecture, concerning on-line and off-line times. Another problem is that pastry only solve the home network interconnection and the resource mediation problem, resource allocation (as described above) is not addressed in Pastry.

In this paper the unstructured and hybrid eDonkey P2P overlay is suggested as a solution for the VHE network virtualization. It has the capability to solve all of the mentioned requirements and is resistant to high churn rates. eDonkey is a very popular file sharing P2P network with a high amount of users (and traffic) [24] that has practically proven to be very scalable. Another reason for this choice is the similarity of the hybrid eDonkey structure to the VHE structure

[10] http://www.skype.com

[11] In P2P parlance, the term churn denotes the stochastic process of peer turnover as occurring when peers join or leave the system.

(end-hosts and gateways). Two kinds of nodes are participating in the eDonkey network: peers and super nodes (index servers). Peers are providing and consuming resources, similar to the end-hosts in the VHE architecture. Super nodes form a separate overlay to share information. A peer can report its available resources to the super node and request the location of hardware resources from the super node. The location of a gateway within the VHE architecture is very similar to the location of a super node — the gateway is physically the first node of each home network. This location makes the gateway the natural place for gathering statistics about the home network that are needed to enable a fair and energy-efficient resource sharing among home networks. In addition, the gateway is supposed to be always-on, which enables it to manage and distribute information among other gateways.

5 Evaluation of Network Virtualization Overhead

Simulation results concerning VHE have already been shown in [3–6]. The simulation itself is a discrete event simulation built with the general purpose programming language Java. Simulated is a meshed network of homes with certain resources (CPU time, disk space, uplink and downlink bandwidth). A home consists of at least one computer and a gateway. Intra- and interhome communication underlies certain delays and links certain latencies. Homes offer an amount of shareable resources. Load is modeled as tasks that are executed on homes and migrated between them. Homes cycle through states that indicate times of work or rest. Further details about initial parameters of simulation runs can be taken from previous works.

In this paper, the simulation is extended by network virtualization approaches. We refer to the application "Download Sharing" (DS) introduced in previous work; a network of interconnected homes exchange (share) download tasks (DS-tasks) for aggregating load on a part of the network with the aim of power saving. In this work we use DS to investigate the traffic produced by our approach under different organization schemes. We do not address power saving at all, but want to know how much traffic originate on those nodes which maintain statistics.

A DS-task is a description of desired content (music, video, etc.) and maximum allowed download bandwidth allocatable for its download. Homes play two roles; as initiator they send out DS-tasks to other homes and receive results later or as executer they receive DS-tasks, download the desired content, and then upload the completed DS-task back to initiators. There is a communication protocol between homes that can be divided into a state phase and a resource phase; in the state phase state information about homes is exchanged, whereas during the resource phase available resources are located and allocated. State information about homes includes the current state of the home (see Figure 1) and the amount of free resources available for executing DS-tasks.

Conceptually the DS application is based on virtual machines (VMs) encapsulating all necessary parts for migrating a DS-task before and after execution. These VMs are rather small on initiator side and grow to considerable file sizes

on executer side because they including now the download content. Therefore, a constraint is the possible usable bandwidth between initiator and executer.

Only for investigating the signaling traffic a server-based approach is compared to two unstructured hybrid overlays in terms of traffic overhead, caused by different strategies of information management.

Initially, a simplified centralized *Server*-based approach has been simulated in VHE. Each *Home network* (H) sends state information to one single H that acts as *Super Home* (SH), which has a global view on the system. In case of task-migrations, all resource requests must be sent to that SH.

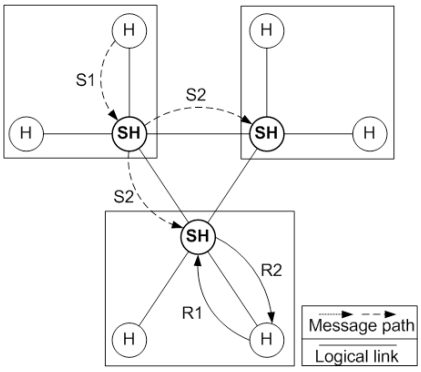

Fig. 2. Topology and information flow *Overlay 1*

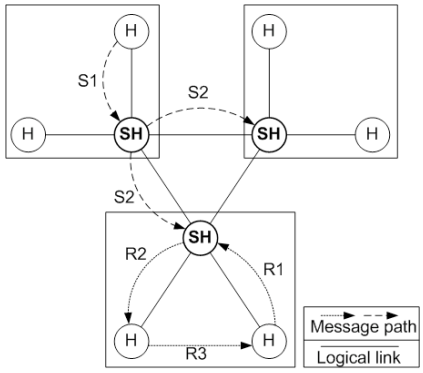

Fig. 3. Topology and information flow *Overlay 2*

The second approach (called *Overlay 1*) shown in Figure 2, is based on an unstructured and hybrid overlay (inspired by eDonkey). A number of SHs are defined, which cluster the network. A SH is a H that additionally acts as a server for a cluster of Hs (clusters are illustrated as squares in Figure 2). Every state change information within a cluster is replicated at all other SHs (SHs have a global view). Resource requests of Hs are answered by their corresponding SH. The arrows in Figure 2 illustrate two independent communication flows. With message $S1$ a H sends its state to the SH. The SH forwards the state message to all other SHs ($S2$). At this point, the state information is replicated amongst SHs. Each state change triggers a state message, therefore a considerable amount of messages is generated. To gain resources, a resource request ($R1$) is sent to to a corresponding SH. The SH replies with a list of currently active Hs ($R2$).

Both of these approaches, *Server* and *Overlay 1* create a global view on the VHE, causing overhead. However, the VHE resource management is mainly done by lightweight always-on gateways as described in Section 3 and the overhead needs to be reduced. To achieve this, a third approach, *Overlay 2* is suggested (again eDonkey inspired) that has modified communication patterns (shown in Figure 3). It has the aim of keeping resource information as local as possible. Hs send state information to their corresponding SH (message $S1$). SHs do not

replicate this information, but exchange meta information about free hardware resources within the own cluster in configurable time-intervals ($S2$). In contrast to *Overlay 1* none of the SHs has a global view on available resources. Resource information is kept local within each SH's cluster, only meta information is exchanged. This results in a different resource request scheme. The resource request first goes to the responsible SH ($R1$). The SH checks its own cluster and forwards the request to a H with free resources if possible ($R2$); this H directly answers the requester ($R3$). Otherwise, if there is no host within the own cluster that can process the task, the SH contacts other SHs, based on the available meta information. If another SH has enough idle resources in its cluster, the initial SH forwards the resource request.

Costs of resource management in the system investigated is mainly based on the number and size of signaling messages within the network. To understand the message complexity of the presented overlays the sizes of modeled messages according to the introduced communication protocol are explained. The message sizes in byte are specified in Table 1. $S = 60$ byte is the size of a state message, $R = 24$ byte is the size of a resource message, and L is the number of entries in the list of active Hs within a SH's cluster in *Overlay 1*. State update messages

Table 1. Message sizes in byte

Message type	**Overlay 1**		**Overlay 2**	
State update	S	$(S1, S2)$	S	$(S1, S2)$
Resource request	R	$(R1)$	R	$(R1, R2b)$
Resource response	$L \times S$	$(R2)$	R	$(R2a, R3)$

in byte/s for *Server* can be approximated by

$$\frac{NMS(T+E)}{Y}, \tag{1}$$

for *Overlay 1* by

$$\frac{NMS(T+E)}{Y} \times \left(2 - \frac{C}{N}\right), \tag{2}$$

and for *Overlay 2* by

$$\frac{\frac{Y}{U_H}C + 2\frac{Y}{U_{SH}}(\frac{N}{C} - 1)}{Y}S \tag{3}$$

where N is the number of Hs (network size), M is the assumed load (number of DS-tasks per H per year), T is the number of state transitions per H according to Figure 1 if homes staying in state active for executing tasks or staying in state passive for saving power, E is the number of state events per task as defined in the communication protocol of *Overlay 1*, Y is the simulation time in seconds (one year), and C is the cluster size (each SH manages C Hs). U_H

& U_{SH} indicate the delay between state updates from Hs and SHs as defined in the communication protocol of *Overlay 2*.

Accordingly, the traffic of resource messages in byte/s per SH for *Server* can be approximately calculated by

$$\frac{M(R+LS)}{Y}N, \tag{4}$$

for *Overlay 1* by

$$\frac{M(R+LS)}{Y}C, \tag{5}$$

and for *Overlay 2* by

$$\frac{2(P_aMC+P_bMC(\frac{N}{C}-1))}{Y}R \tag{6}$$

which shows that the critical parameter N dramatically increases the message complexity for *Server*, whereas *Overlay 1* only relies on the number of clusters C which is a predefined fraction of N. For *Overlay 2* additional parameters P_a & P_b are necessary. Those parameters express the probability if a task can be processed within the requester's cluster (P_a) or if it is forwarded to another cluster (P_b).

Table 2. Expected state update traffic per SH in byte/s

N	**Server**	**Overlay1**	**Overlay2**
100	0.69	1.21	1.65
1000	6.93	13.68	6.45
10000	69.25	138.34	54.45
100000	692.54	1384.91	534.45

Table 2 presents state update traffic in a network with $C = 25$ with N increasing. The *number of tasks per week per home* (λ) is fixed to $\lambda = 10$, one year is considered, U_H is 1200 s and U_{SH} is 900 s. As the table clearly shows, traffic is correlated with network size N. For verification and comparison all overlays are simulated (in a discrete event simulation [3–6]). The results are illustrated in Figures 4 and 5.

Figure 4 shows state and resource traffic in byte/s with regard to N. Generally it can be seen that state traffic is much more critical than resource traffic. For state traffic *Server* and *Overlay 2* clearly outperform *Overlay 1*. For $N > 700$ *Server* is also outperformed by *Overlay 2*; however state traffic is still linear and a matter of scalability. Further the resource traffic of Server grows with N.

Figure 5 shows the same for fixed N and increasing λ. Again, the state traffic is the more critical one. *Overlay 2* clearly outperforms *Overlay 1* for $\lambda > 6$ and Server for $\lambda > 12$. Overall, *Overlay 2* exhibits the good property of being

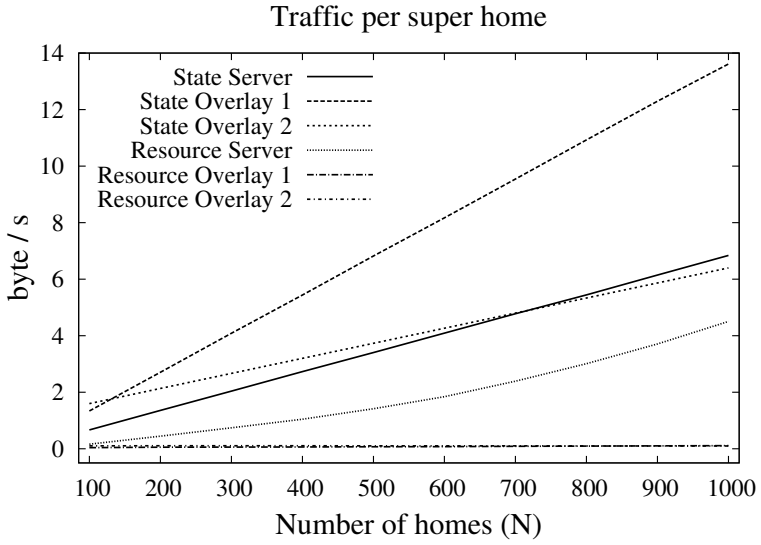

Fig. 4. Traffic in terms of network size with a fixed load of 10 tasks per week

invariant to load. Even the overhead caused by *Overlay 1* does not constrict Hs with high synchronous access bandwidth like 50 Mbit/s to act as SH.

After this worst case analysis about the feasibility of our approach the next step will be to choose a P2P-system which best fits to our approach of energy efficient resource sharing for homes but also minimizes communication overhead.

6 Conclusion and Future Work

This paper has presented an energy-efficient virtual home environment that enables resource sharing among users in home networks. Particularly, the virtualization-related aspects of the suggested environment were discussed. Therefore, virtualization-related requirements of a future home environment were defined and virtualization methods were selected to meet these requirements. In this context, an overview on network virtualization approaches was given to motivate the selection of virtualization solutions .

Simulations that are described in previous papers [3–6] have been extended by simplified network virtualization approaches. In this paper these approaches were evaluated and compared to each other concerning their effects on the energy-efficient operation of the envisioned future home environment. The overhead which is imposed by the network virtualization approaches was illustrated and discussed. The paper showed that the chosen instance of network virtualization is conceptually appropriate for the future home environment architecture. However, it has to be improved in terms of overhead and scalability in future work. Additionally, it has to be evaluated in future works, in which way the distributed

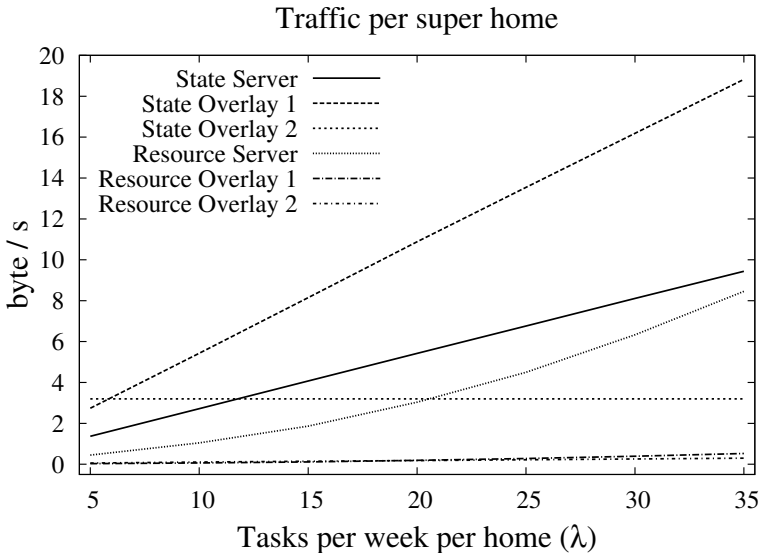

Fig. 5. Traffic in terms of load with a fixed network size of 400 Hs

file-sharing management of the suggested network virtualization approach can be adapted to enable a balanced and energy-efficient resource-sharing management in future home environments.

References

1. Koomey: Estimating total power consumption by servers in the us and the world, technical report. Technical report, Lawrence Berkeley National Laboratory Stanford University (February 2007)
2. Bertoldi, Atanasiu: Electricity consumption and efficiency trends in the enlarged european union, institute for environment and sustainability (2007)
3. Hlavacs, Hummel, Weidlich, Houyou, Berl, de Meer: Energy Efficiency in Future Home Environments: A Distributed Approach. In: 1st Home Networking Conference, Paris, France (12 2007)
4. García, Berl, Hummel, Weidlich, Houyou, Hackbarth, de Meer, Hlavacs: An Economical Cost Model for Fair Resource Sharing in Virutal Home Environments. In: 4th EURO-NGI Conference on Next Generation Internet Networks (NGI-2008), Krakow, Poland (4 2008)
5. Hlavacs, Weidlich, Hummel, Houyou, Berl, de Meer: Distributed energy efficiency in future home environments. Annals of Telecommunications - Home networking: performance and architecture challenges **63**(9-10) (2008)
6. Hlavacs, Weidlich, Treutner: Energy Saving in Future Home Environments. In: 2nd Home Networking Conference at IFIP Wireless Days, Dubai, United Arab Emirates (11 2008)
7. Pope, Meredith: The Virtual Home Environment, Release 5. Technical Report TR 22.121 V5.3.1, 3GPP (7 2002)

8. Roque, Soares, Oliveira: VESPER Project- Validation of VHE Concept (2001)
9. Roussaki, Jormakka, Xynogalas, Laikari, Chantzara, Anagnostou: Multi-terminal and multi-network access to virtual home environment. In: IST Mobile and Wireless Telecommunications, Thessaloniki, Greece (6 2002)
10. Geuna: UMTS Network Aspects. EURESCOM Project P920, VHE Trial review report, Deliverable 4 (1 2001)
11. Liotta, Yew, Bohoris, Pavlou: Supporting Adaptation-aware Services through the virtual home environment (2002)
12. Nakajima, Ueno, Tokunaga, Ishikawa, Satoh, Aizu: A Virtual Overlay Network for Integrating Home Appliances. In: 2002 Symposium on Applications and the Internet (SAINT'02), Nara, Japan (7 2002)
13. Chun, B., Culler, D., Roscoe, T., Bavier, A., Peterson, L., Wawrzoniak, M., Bowman, M.: Planetlab: an overlay testbed for broad-coverage services. SIGCOMM Comput. Commun. Rev. **33**(3) (2003) 3–12
14. Bavier, A., Feamster, N., Huang, M., Peterson, L., Rexford, J.: In vini veritas: realistic and controlled network experimentation. In: SIGCOMM '06: Proceedings of the 2006 conference on Applications, technologies, architectures, and protocols for computer communications, New York, NY, USA, ACM (2006) 3–14
15. Bellard, F.: QEMU, a fast and portable dynamic translator. In: Proceedings of the USENIX Annual Technical Conference, FREENIX Track. (2005) 41–46
16. Barham, P., Dragovic, B., Fraser, K., Hand, S., Harris, T., Ho, A., Neugebauer, R., Pratt, I., Warfield, A.: Xen and the art of virtualization. SIGOPS Oper. Syst. Rev. **37**(5) (2003) 164–177
17. Chowdhury, N.M.K., Boutaba, R.: A survey of network virtualization. Technical report, University of Waterloo (Oct. 2008)
18. Feamster, N., Gao, L., Rexford, J.: How to lease the internet in your spare time. SIGCOMM Comput. Commun. Rev. **37**(1) (2007) 61–64
19. Berl, A., Fischer, A., de Meer, H.: Using System Virtualization to Create Virtualized Networks. In: Workshops der Wissenschaftlichen Konferenz Kommunikation in Verteilten Systemen (WowKiVS2009), Kassel, Germany, March 2-6, 2009. Volume 17 of Electronic Communications of the EASST., EASST (March 2009)
20. Egi, N., Greenhalgh, A., Handley, M., Hoerdt, M., Mathy, L., Schooley, T.: Evaluating xen for router virtualization. In: 16th Int. Conf. on Comp. Commun. and Networks - ICCCN 2007. (Aug. 2007) 1256–1261
21. Menon, A., Cox, A.L., Zwaenepoel, W.: Optimizing network virtualization in xen. In: USENIX Annual Technical Conference. (May 2006) 15–28
22. Steinmetz, R., Wehrle, K.: Peer-to-Peer Systems and Applications (Lecture Notes in Computer Science). Springer-Verlag New York, Secaucus, NJ, USA (2005)
23. Stoica, I., Morris, R., Karger, D., Kaashoek, M.F., Balakrishnan, H.: Chord: A scalable peer-to-peer lookup service for internet applications. In: SIGCOMM '01: Proceedings of the 2001 conference on Applications, technologies, architectures, and protocols for computer communications, ACM Press (2001) 149–160
24. Tutschku, K.: A measurement-based traffic profile of the eDonkey filesharing service. Lecture notes in computer science (2004) 12–21
25. Rowstron, A., Druschel, P.: Pastry: Scalable, distributed object location and routing for large-scale peer-to-peer systems. In: IFIP/ACM Int. Conference on Distributed Systems Platforms (Middleware). Volume 11., Heidelberg (2001) 329–350