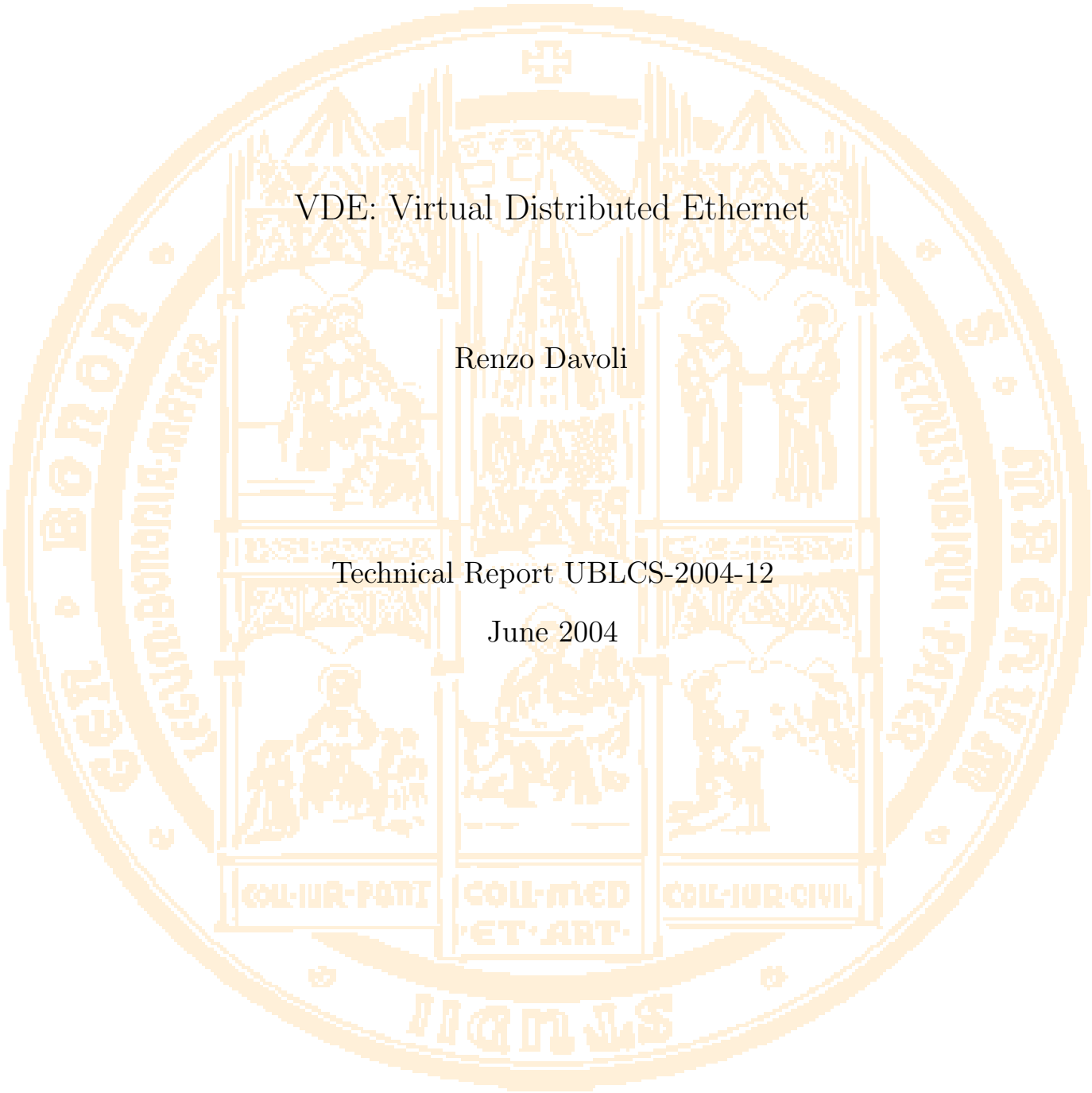

VDE: Virtual Distributed Ethernet

Renzo Davoli

Technical Report UBLCS-2004-12

June 2004

Department of Computer Science
University of Bologna
Mura Anteo Zamboni 7
40127 Bologna (Italy)

The University of Bologna Department of Computer Science Research Technical Reports are available in PDF and gzipped PostScript formats via anonymous FTP from the area ftp.cs.unibo.it:/pub/TR/UBLCS or via WWW at URL http://www.cs.unibo.it/. Plain-text abstracts organized by year are available in the directory ABSTRACTS.

Recent Titles from the UBLCS Technical Report Series

2003-9 An Object Based Algebra for Specifying A Fault Tolerant Software Architecture, Dragoni, N., Gaspari, M., June 2003.

2003-10 A Scalable Architecture for Responsive Auction Services Over the Internet, Amoroso, A., Fanzieri F., June 2003.

2003-11 WSSecSpaces: a Secure Data-Driven Coordination Service for Web Services Applications, Lucchi, R., Zavattaro, G., September 2003.

2003-12 Integrating Agent Communication Languages in Open Services Architectures, Dragoni, N., Gaspari, M., October 2003.

2003-13 Perfect load balancing on anonymous trees, Margara, L., Pistocchi, A., Vassura, M., October 2003.

2003-14 Towards Secure Epidemics: Detection and Removal of Malicious Peers in Epidemic-Style Protocols, Jelasity, M., Montresor, A., Babaoglu, O., November 2003.

2003-15 Gossip-based Unstructured Overlay Networks: An Experimental Evaluation, Jelasity, M., Guerraoui, R., Kermarrec, A-M., van Steen, M., December 2003.

2003-16 Robust Aggregation Protocols for Large-Scale Overlay Networks, Montresor, A., Jelasity, M., Babaoglu, O., December 2003.

2004-1 A Reliable Protocol for Synchronous Rendezvous (Note), Wischik, L., Wischik, D., February 2004.

2004-2 Design and evaluation of a migration-based architecture for massively populated Internet Games, Gardenghi, L., Pifferi, S., D'Angelo, G., March 2004.

2004-3 Security, Probability and Priority in the tuple-space Coordination Model (Ph.D. Thesis), Lucchi, R., March 2004.

2004-4 A New Graph-theoretic Approach to Clustering, with Applications to Computer Vision (Ph.D Thesis), Pavan., M., March 2004.

2004-5 Knowledge Management of Formal Mathematics and Interactive Theorem Proving (Ph.D. Thesis), Sacerdoti Coen, C., March 2004.

2004-6 An architecture for Content Distribution Internetworking (Ph.D. Thesis), Turrini, E., March 2004.

2004-7 T-Man: Fast Gossip-based Construction of Large-Scale Overlay Topologies, Jelasity, M., Babaoglu, O., May 2004.

2004-8 A Robust Protocol for Building Superpeer Overlay Topologies, Montresor, A., May 2004.

2004-9 A Unified Approach to Structured, Semistructured and Unstructured Data, Magnani, M., Montesi, D., May 2004.

2004-10 Exact Methods Based on Node Routing Formulations for Arc Routing Problems, Baldacci, R., Maniezzo, V., June 2004.

2004-11 Mapping XQuery to Algebraic Expressions, Magnani, M., Montesi, D., June 2004.

VDE: Virtual Distributed Ethernet

Renzo Davoli[2]

Technical Report UBLCS-2004-12

June 2004

Abstract

The idea of VDE is straightforward simple and can be applied in very many configuration to provide several services. It is a kind of Swiss knife of emulated networks. It can be used as a general Virtual Private network as well as a mobility support technology, a tool for network testing, a general reconfigurable overlay network, a layer for implementing privacy preserving technologies and many others.

2. Department of Computer Science, University of Bologna, Italy, Mura Anteo Zamboni, 7. 40127 Bologna, Italy. email: renzo@cs.unibo.it. This work has been partially supported by the Webminds FIRB project of the Italian Ministry of Education, University and Research.

1 Introduction

The acronym VDE is very self explaining: it is a Virtual network as it is built completely in software, it is Distributed as parts of the same network can run on different physical (real) computers, it is an Ethernet as the entire virtual software structure is able to forward, dispatch and route plain ethernet packets. The main feature of VDE, thus, are the following:

- VDE is Ethernet compliant.
- VDE is general, it is a virtual infrastracture that gives connectivity to several kinds of software components: emulators/virtual machines, real operating systems and other connectivity tools.
- VDE is distributed.
- VDE does not need specifically administration privileges to run.

. A virtual ethernet network facility is implemented as a complementary feature of some emulation or virtual machine software (e.g. [9],[4, 5], [21]). This feature permits the concurrent execution and communication of several virtual or emulated machines through an emulated network. These tools however fail in generality and distribution: in generality as they are tailored to one single kind of virtual or emulated machine, in distribution as everything (emulate machines and the network emulator itself) must reside and run on the same real computer.

It is possible to create a completely virtual/emulated world by using VDE together with virtual machines. This second layer of virtuality has been also named as Virtual Square [2]. Square has here the double meaning of squared (i.e. elevated to the second power) and of a virtual place where machines and humans can meet.

The remaining if the paper is organized as follows. Section 2 presents the structure and the composing entities of a VDE, section 3 explains some examples of VDE appplications. A section about Related Work and a section with final remarks and future work complete the paper.

2 Structure of VDE

VDE has the same structure of a modern Ethernet network. The main components, in fact, are vde_switches and vde_cables.

- A vde_switch is the virtual couterparts of a physical Ethernet switch (or hub). A switch has several ports where the user can plug in computers, routers, other ethernet-compliant equipments.
- It is also possible to interconnect two different switches together by using a so called cross-cable plugging it from a port of one switch to a port of the other. VDE has the virtual counterpart of the crossed cable: a VDE-cable. It is a software tool able to interconnect two vde-switches.

VDE is also able to integrate real computers in the emulated network. When a real computer is connected to VDE a virtual interface (based on tuntap) is seen from the operating system as it were a hardware interface and behaves as a physical ethernet interface or virtual machines. This operation however changes the network behavior of the host computer and thus need administrative privileges to be completed.

Currently it supports User-Mode Linux virtual machines, qemu, Bochs and MPS.

- User-Mode Linux. [4, 5] It is a project that realizes a complete system virtualization through system trapping . It has been released as a set of patches for the linux kernel that defines a new virtual "um" hardware architecture. A kernel for the "um" architecture is just an executable for the host computer that include the I-O virtualization routines as well as the kernel itself. It runs at user-level. It does not require a specific kernel support in the host

machine but there is patch named skas mode for the 2.4 version of Linux to increase U-ML security and performance (to reduce the number of threads and to keep the addressing space of the emulated kernel inaccessible by the emulated tasks). The support for skas mode should be included in vanilla Linux 2.6.

- Qemu. [1] Quoting its author's web site: Qemu is a FAST! processor emulation using dynamic translation to achieve good emulation speed. Qemu is able to run just as a processor or as a Complete System Virtualizer. It is possbile to run executables compiled for different processor architectures in a linux environment or it is possible to start a virtual machine and boot an entire operating system. It runs on a number of different hardware architectures, it is currently able to run i386, ppc, arm and sparc executables and provides virtual machines emulating i386 and ppc based architectures. This project is very active: new ports and features are announced on daily base. It runs completely at user-level and virtualizes completely the processor architecture.

- Bochs [14] is an historical virtual machine. It runs on several host architectures (supported host OS: Linux, MacOS 9/X, Windows) where it is able to create complete system virtualization of an i386 architecture. It relies on standard emulation techniques thus it is quite slow when compared to modern virtual machines. It runs completely at user-level and virtualizes completely the processor architecture.

- MPS and uMPS (micro MPS) [16] have been designed for educational purposes. Like Qemu and Bochs, MPS is a complete virtual system of an Mips based computer (user-level, complete processor virtualization). It is a workbench for computer science students to run their experimental operating systems in a real-world consistent virtual computer while stripping off unnecessary complexities.

A vde_switch operates as a real switch: is a fast bridge able to manage the dynamical association between hardware (MAC) address and port. The switch learns from the headers of the packets exchanged on the network which is the mapping each MAC address and the corresponding port. As a real switch a vde_switch implements the network traffic separation that leads to a higher aggregate bandwidth. Vde_switch also implements the Mac to Port mapping aging to allow a graceful convergence to a new configuration when the topology changes. When several switches are interconnected by vde_cables only packets that have source and destination on the opposite sides of the cable as well as broadcast packets and packets sent to already unknown destinations. There is an option for the vde_switch to use it as a hub. This can be useful for debugging, to plug in a network traffic analyzer, or for educational purpose: e.g. to show the security threats that can be put in place on hub based networks.

Vde_cables are composed by three software components:

- Two vde_plugs, one at each end of the cable. A vde_plug is a program that has been designed to be connected to a vde_switch and converts all the traffic to a standard stream connection.

- An interconnection tool, that is able to bi-directionally connect the streams of two vde_plugs.

An interconnection tool can be a double pipe, i.e. a bidirectional extension to the standard Unix tool to interconnect commands. It is also possible to connect switches running on different host computers by the joint use of a double pipe and a standard remote execution tool like rsh (not secure) or ssh. Netcat is another well known program that can be used as an interconnection tool.

There is also a Slirp tool for VDE. Slirp is a tool by Danny Gasparovski dated back to 1995. At that time internet providers proposed two different kinds of contracts: a cheap remote terminal connection and an expensive ppp/slip service. Slirp was able to convert a terminal line into a ppp/slip access for client applications. Slirp runs completely at User-level: whenever a client application tries to open a new network connection, slirp catch the connect request. Slirp does the connect for the internal application and then forwards all the packets. From the Internet (and

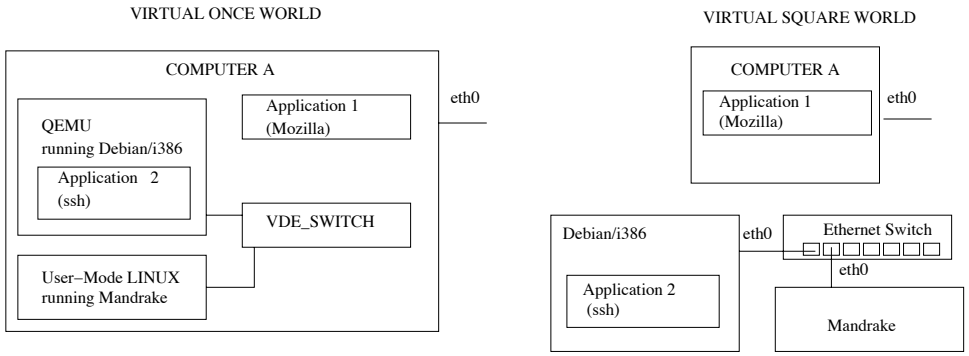

Figure 1. A simple application of VDE

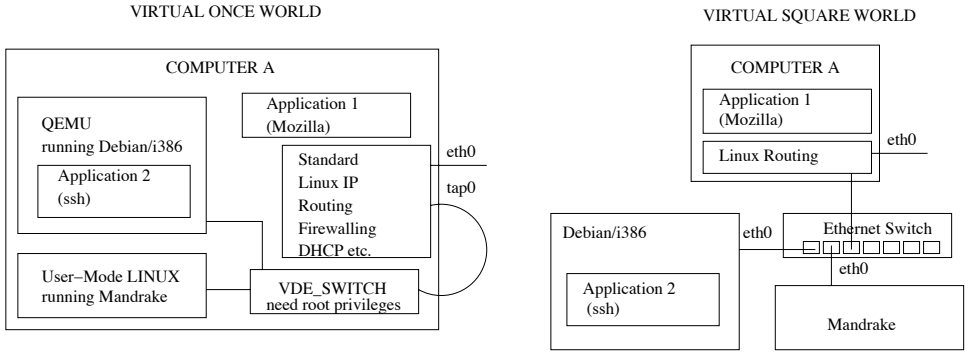

Figure 2. A VDE routed to the Internet

from the host computer operating system) point of view it is like all the connections were inited by slirp itself. It supports TCP on IPv4, currently does not work with IPv6. The VDE slirp tool has also a DHCP so that a VDE network with slirp behaves as if were an IPv4 only masquered network.

3 Detailed examples Applications of VDE

As stated in the abstract, VDE can have several applications. In this section we pass through some examples and analyze which common problems can be solved.

Figure 1 shows the simplest usage of a VDE: One or several local virtual machines can be interconnected by the virtual network. In the example the ssh client running as an application for the Linux Debian O.S. on the Qemu virtual machine can open a session to the Linux Mandrake machine running as a User-Mode Linux virtual machine. It is possible to run several vde_switch on the same computer: each vde_switch has a Unix Socket as its identificator and channel to communicate with the switch. It is also possible for several users to join the same virtual network. On the contrary it is possible for a user to keep different virtual networks running. Permission to access a specific network is granted or revoked by setting the permissions of the access socket (as it were a standard Unix file). This network structure can be used in computer science education: students can run their virtual machine as administrators and test services using other virtual machines. [3]. This configuration has no inpact on the overall network security as the virtual network and the real network are not interconnected. As a consequence it can be used to test malicious software in a confined environment. This configuration can be completely set up as ordinary users of the host computer.

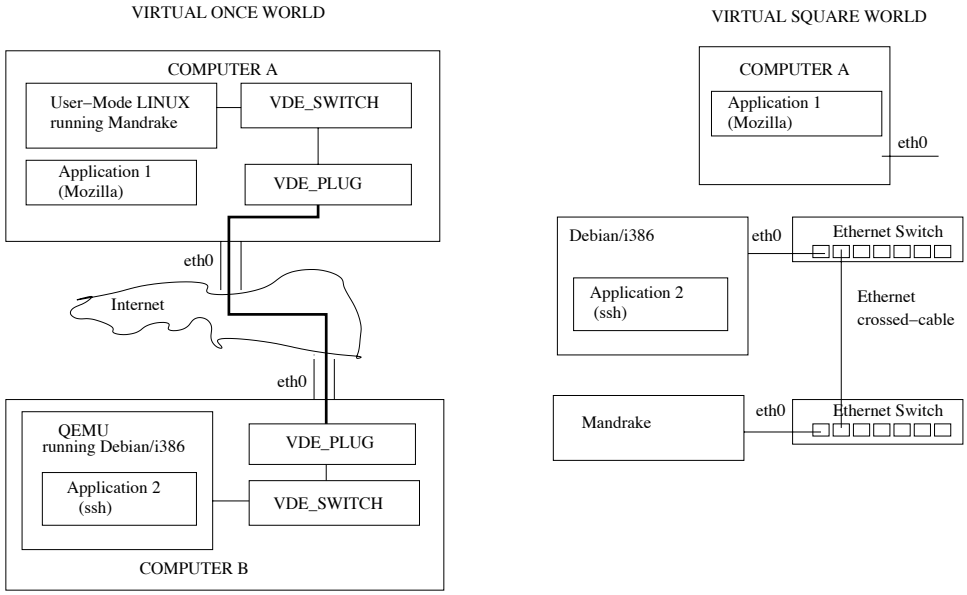

Figure 3. A VDE with remote clients

The second example (see Fig. 2) is a slight variation of the first. The virtual network infrastructure is connected to the host operating system by the tuntap support. THe host computer is then logically connected to the VDE as it had a physical interface to a real ethernet. All the methods used for routing, bridging, firewalling, packet filtering and shaping can be applied to this configuration. Is it possible to configure the host just as a packet forwarded (packer routing) or as an IPv4 NAT/masquerading [8] firewall with DHCP support [6] or as an IPv6 router with network autoconfiguration [18]. All the tools that can be installed on a real boundary machine of an internal network like DNS forwarders, service proxies, mail agents etc. can be installed on the host machine. The VDE is interconnected to the Internet (or the external network) as it were a standard real ethernet. Obviously all the installation and management of the network tools and virtual interface on the host computer operating system needs administration privileges (root access). This configuration can be used to connect virtual machines to a real network, putting in place all the security structures needed. It is possible to test network install or upgrade procedures for operating systems or applications and debug them with no need of real hardware reboots. Disk images or Copy-On-Write tecniques can be used to return to a previous checkpoint in the emulated system status in case of failures.

Figure 3 shows the usage of a vde_cable. This is similar to the first example proposed: the network topology is perceived in the same manner by the users of the two virtual machines. A user of the Debian Linux system running on the Qemu emulator open a connection on the User-Mode mandrake Linux. The two virtual machines operates as they were on the same ethernet local area network. With this configuration it is possible to run distributed system emulations on a cluster of workstation or it is possible to join and test the services running on a remote virtual network using a local virtual machine. A vde_cable can also put in place between two vde_switches running on the same host computer. This configuration can be used to interconnect clusters of virtual computer and reduce the load on a single switch (if the pattern of traffic is consistent with the virtual topology) or to test the reliability of the application in case of network partitioning. In fact starting and stopping the vde_cable leads to the simulation of a network partition. It is simple to create interconnection tools for the vde_cable able to simulate packet dropping, delays and other network behaviors very useful for tests.

The scenario depicted in figure 4 is the same of 3 but the interconnection to the host Operating System and then the possibility of routing virtual nerwork traffic to the Internet. This network

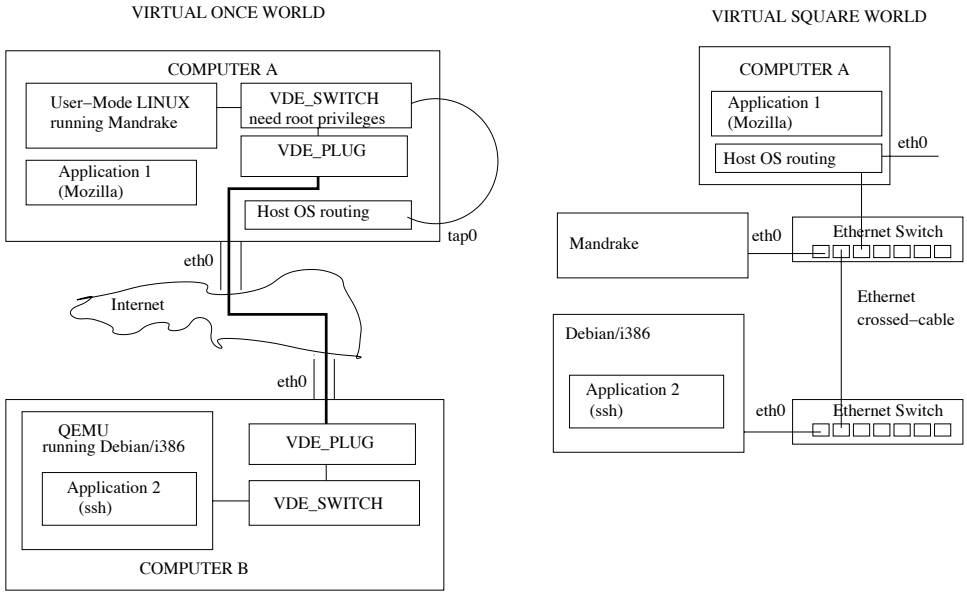

Figure 4. A VDE with remote clients routed to the Internet

configuration has been used at the School of Computer Science in Bologna to teach Operating System administration.

Each student can run her virtual machine on a workstation in the computer laboratory as well as on her own personal computer or laptop and through a single switch connected to the Internet she use her virtual machine as it were a real computer on the network. It is also possible to keep a virtual machine running at the Department and then to join the virtual network from home and test the services provided as well as practice in remote administration. A class of student can manage a virtual network as they were real system administrators of a cluster of computers each one providing different services (DNS, Mail, Web, NTP, News etc.). We give to students a direct Internet connection for their virtual machines with some packet filtering for security and confidentiality.

Remote connections opening and closing are logged for security: it is possible to find the correspondence between real IP on the Internet of the remote computer, the virtual IPv4 and IPv6 addresses used on that connection and the username that was authenticated to allow the connection.

The example shown in Fig. 5 does not make use of virtual machines, VDE is used here as a general tunneling tool for real computers. There are two vde_switches running on two different computers. Both vde_switches are connected to a local tap interface. Computer A has been configured to be a network router/firewall as in the previous examples. Computer B has its tap interface and the IP address of the tap interface at Computer A as computer B default route. (A fixed route for the Computer A on eth0 or a policy routing definition is needed for the interconnection tool of the vde_cable not to use the defult route). In this way all the applications at Computer B use VDE instead of the real network regardless of the protocol used. This configuration has a number of possible usages.

- VPN: the user has her own connectivity. If the connectivity tool is encrypted (e.g. ssh) all the network traffic is encrypted.
- Mobility: the user can terminate a vde_cable and start a new one. Nothing changes in the virtual network, if the two actions are not contemporary maybe some packets get dropped but the data-link network topology does not change. It has the same effect to change a crossed cable between two real switches. There is no need for IP reconfiguration and the

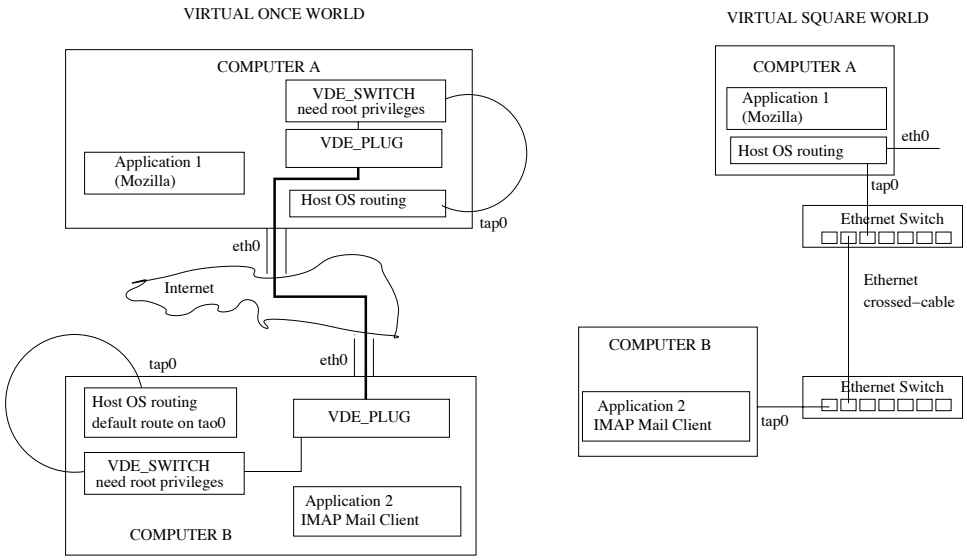

Figure 5. A VDE used as a general tunneling tool

transport level connections (e.g. TCP) do not drop. The new vde_cable can pass through a real network interface with a different IP address because it is connected to a different local area network or even use a completely different interface maybe based on a different technology. For example the traffic can be moved from a ethernet to a 802.11, a serial PPP or to a GPRS/UMTS mobile service. These intra or inter technology hand-offs do not affect the virtual network topology, thus the user can experience just a difference in performance, not in connectivity. vde_switch software allow the temporary presence of several cables between a pair of switches by setting a specific option. This can led to duplicate packets but avoids the temporary disconnections during hand-offs.

- Privacy: it is possible to change the real IP address and, if the virtual network has several distributed switches, the IP address of the corresponding host. This makes the pattern of traffic less trackable from an external spying software. It is possible to use dynamically changing addresses as in [17] or as proposed in [19].

- Unsupported protocol or family of protocols in the real infrastructure. By VDE it is possible to use protocols that are unsupported on the physical network in use. It is the case for example of the use of IPv6 with a provider which does not support that family of protocols or the use of non routable LAN protocols between geographically disdtributed systems.

Several vde_cables for several remote tunnel services can work at the same time. Computer A can be seen as a tunnel broker [7] just by using VDE.

The latter example of this set (see Fig. 6) has a single difference from the one shown in Fig. 4: the use of slirpvde instead of the tuntap interface. Slirpvde runs with user permissions thus it is not needed to have root access to interface the virtual network to the real networks at Computer A. All the network connections originated from inside the virtual network having destination outside are regenerated by the slirpvde program as they were all genuine connections initiated by slirpvde itself. All the internal nodes (virtual or real computers) are then interconnected to the Internet as they were on a NAT/masqueraded LAN. All client applications work normally, there is currently no support for servers trough port forwarding. In any case it would be an indirect support: servers appear on the internet as hosted by the host computer (computer A in the figure) at a specified port. Slirp would have to forward the request to an internal node. Slirp currently supports IPv4 only.

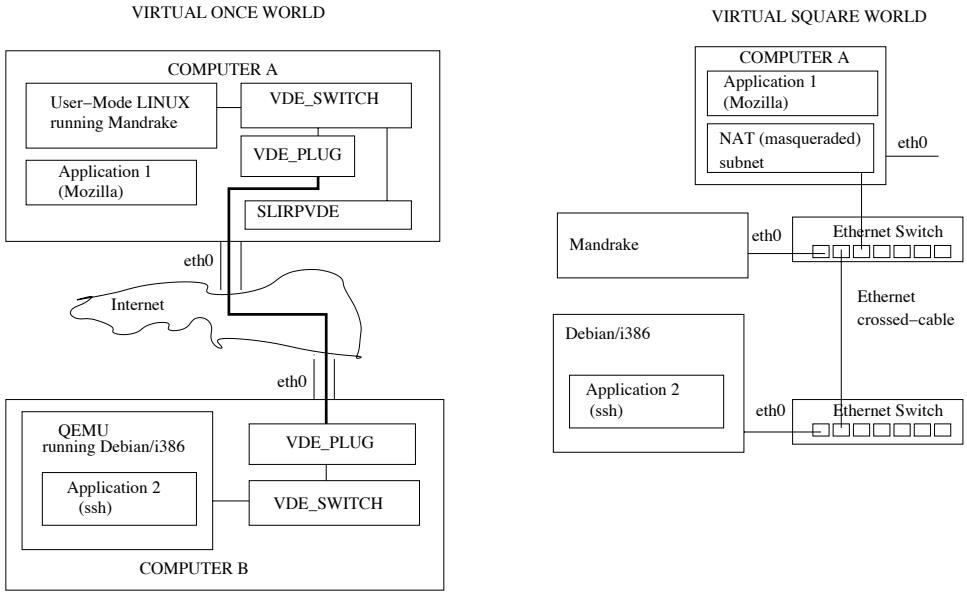

Figure 6. A VDE with remote clients and VDESLIRP

4 Related Work

Several tools do exist that are able do cover part of the features of VDE. Point to Point Tunneling Protocols (PPTP) [11] and Level 2 Tunneling Protocols (L2TP) [20] are both data-link tunneling protocols. Both emulate a point-to-point connection able to run PPP. PPTP has internal security features while L2TP needs IPsec to create secure channels. These protocols and tools have not been designed for distributed networks but just as point to point connections for VPN. There are many examples of tool for IP tunneling like openVPN [22], Zebedee [23], IPsec [12] e.g. the Free S-WAN [10] implementation. These tools are tailored to IP and thus less general with respect to VDE. Virtual Tunnel (VTUN) [13] and Virtual Private Ethernet (VPE) [15] have more similarities with VDE. In fact VTUN and VPE can create data link ethernet compliant tunnels. All the tools here listed are interfaced to the tuntap kernel driver thus always need to have root access for their installation. Moreover they have been designed for real operating systems and does not interface virtual machines.

5 Conclusions and future work

VDE is a multipurpose tool able to give a simple and generalized solution for a wide area of applications. VDE project is not complete, several other features are being designed or under development. Vde_switch is not currently supporting a loop detection and management feature. Real ethernet networks implement the Minimum Spanning Tree protocol, a different approach is under study in order to avoid inproductive network traffic. The vde_switch currently does not manage multicast (all the multicast packets are processed as broadcast). A better implementation able to deal also with IPv6 multicating methods is under development. A tool named Ale4NET (application level for networking) is already completed, now it is available in alpha version. Ale4NET is a network only virtual machine: with a simple configuration one or several applications on the host computer can use the virtual network instead of the real host machine network while running on the real processor and using the real Operating System services of the host computer.

6 Acknowledgements

The code for vde_switch is based on the User-Mode Linux networking tool by Yon Uriarte and Jeff Dike. I wish to thank dr. Nicola Mezzetti for his useful suggestions for the related work section.

References

[1] F. Ballard. Qemu project home page. http://fabrice.bellard.free.fr/qemu/.

[2] R. Davoli. Virtual square home page. http://www.virtualsquare.org/.

[3] R. Davoli. Teaching operating systems administration with user mode linux. In Proceeding of the 9th Annual Conference on Innovation and Technology in Computer Science Education. ACM, 06 2004. Leeds, UK.

[4] J. D. Dike. User-mode linux. In Proc. of 2001 Ottawa Linux Symposium (OLS), Ottawa, 2001.

[5] J. D. Dike. Making linux safe for virtual machines. In Proc. of 2002 Ottawa Linux Symposium (OLS), Ottawa, 2002.

[6] R. Droms. Rfc 2131 - dynamic host configuration protocol. Technical report, IETF, 1997.

[7] A. Durand, P. Fasano, I. Guardini, and D. Lento. RFC 3053 - IPv6 Tunnel Broker. IETF, 2001.

[8] K. Egevang and P. Francis. Rfc 1631 - the ip network address translator (nat). Technical report, IETF, 1994.

[9] Microsoft (formerly from Connectix). Virtual pc (proprietary product). http://www.microsoft.com/windowsxp/virtualpc/.

[10] J. Gilmore, H. Daniel, M. Richardson, R. G. Briggs, H. Redelmeier, C. Schmeing, S. Sgro, J. Ioannidis, A. D. Keromytisr, H. Spencer, and S. Harris. Free s/wan project web site.

[11] K. Hamzeh, G.S. Vertein, W. Vertheini, J. tarraudi, and W.A. Little. Point-to-point tunneling protocol. Internet draft, IETF, July 1997.

[12] S. Kent and B. Atkinson. Rfc 2041: Security architecture for the internet protocol. Technical report, IETF, 1998.

[13] M. Krasnyansky. Virtual tunnel: Vtun (web site). http://vtun.sourceforge.net.

[14] K. Lawton. Bochs project home page. http://bochs.sourceforge.net.

[15] M. A. Lehmann. Virtual private ethernet (vpe) web site. http://savannah.gnu.org/projects/gvpe.

[16] M. Morsiani and R. Davoli. Learning operating system structure and implementation through the MPS computer system simulator. In Proceedings of the 30th SIGCSE Technical Symposium on Computer Science Education, pages 63–67, New Orleans, 1999.

[17] T. Narten and R. Draves. Rfc 3041: Privacy extensions for stateless address autoconfiguration in ipv6. Technical report, IETF, 2001.

[18] S. Thomson and T. Narten. Rfc 2462 - ipv6 stateless address autoconfiguration. Technical report, IETF, 1998.

[19] M. Tortonesi and R. Davoli. User untraceability in next-generation internet: a proposal. In IASTED, editor, Proceeding of Communication and Computer Networks 2002 (CCN 2002), pages 177 – 182, November 2002.

[20] W. Townsley, A. Valencia, A. Rubens, G. Pall, G. Zorn, and B. Palter. RFC 2661 - Layer Two Tunneling Protocol "L2TP". IETF, 1999.

[21] Vmware inc. http://www.vmware.com/.

[22] J. Yonan. Openvpn project web site. http://openvpn.sourceforge.net.

[23] Zebedee web site. http://www.winton.org.uk/zebedee/.