A System for Providing A Ttraining Environment for Linux Networks using Virtual Environment Softwares

Yuichiro TATEIWA[†], *Takami YASUDA*[††], *and Shigeki YOKOI*[††]

Graduate School of Information Science, Nagoya University, Japan

Summary
We have developed a system to provide a training environment where trainees can learn Linux network construction in a safe, easy, and relaxed manner using the environment software User-mode Linux. One trainee can construct networks consisting of virtual network components on one standalone PC. Therefore, it is not necessary that trainees connect this system to the Internet; trainers do not need to prepare a variety of physical network components, and trainees need not worry about system down or the negative influence with other networks caused by operation errors. Evaluation experiments using eleven students indicated that our system is effective for training in Linux network construction, though some problems remain with the user interfaces.

Key words:
Network construction training, Virtual environment software, Open source, Linux

1. Introduction

Training to construct LANs, mainly using Linux, has been undertaken in lectures for network administrators in universities and vocational schools. Trainees construct networks consisting of actual hardware, such as Linux servers, routers, and switching hubs. In addition, there has been vigorous development of virtual environment software, such as VMware [1]. Furthermore, because of strong performance gains in conventional PCs, it is now possible to run multiple virtual environment software on just one machine.

Using the environment software "User-mode Linux" [2] (following UML), we have developed a system to provide a training environment where trainees can practice Linux network construction in a safe, and simple, and relaxed manner. With this system one trainee can construct networks consisting of virtual network components on one standalone PC. We consider such a training environment to be useful for the case where students study on their own or where lectures are performed under certain limitations, such as training time and expense.

We have previously researched a system that uses virtual networks realized by "User-mode Linux" [3]. With this system, however, trainees have not been able to safely perform tasks that are necessary for connecting to the Internet. This time, we developed a function to perform particularly important tasks such as software package management and domain management. Using this "pseudo-Internet function," trainees were able to perform training contents without sacrificing the previous merits (safe, easy, relaxed).

Relational systems [4][5][6][7][8] possess the possibility of performing network construction training and Linux-server settings training on one PC. In [4], trainees could constitute network topology and perform TCP/IP settings, such as setting IP address. The system in [5] could construct a Linux server on one PC by Internet and a Linux booted from a CD without affecting an existing system. In [6][7][8], trainees could design virtual networks and had the system simulate their actions. These systems, however, are not appropriate for the purpose of this study because they cannot construct networks including Linux servers.

2. Traditional Linux network construction training and the virtual environment software "User-mode Linux"

2.1 Traditional Linux network construction training

Traditionally in universities and vocational schools, students practice network construction by using about 10-20 actual network machines. The networks consist of Linux servers, clients, routers, switching hubs, and network cables. The procedure for this training is as follows.

(1) Linux installation onto server hardware: first of all, trainees install Linux onto their server hardware.
(2) Network component installation: Next, trainees install network components. This work requires a sufficiently large physical space to handle all the components installed by all the network builders in class. Usually, a practice room is used for this purpose.

Manuscript received August 5, 2006.
Manuscript revised August 25, 2006.

(3) System settings: Trainees perform the tasks shown in Table 1.

Table 1: Main task contents

Components	Work contents
Linux server	TCP/IP settings, Server software package management (Web, Mail, Windows sharing, DNS, etc)
Client	TCP/IP settings
Router	TCP/IP settings, Firewall settings, DHCP settings, Route settings

(4) Network status check: Trainees check whether those servers and those networks work according to their purpose. To check TCP/IP settings and server software settings, trainees perform telecommunication via a Web browser and clients' mailers. To check firewall settings and route settings, trainees perform telecommunications from a different network such as the campus LAN or their home.

2.2 Features of the virtual environment software "UML"

UML runs as Linux on a real Linux machine, consuming few computing resources. We have confirmed that at least 20 UML units can be simultaneously executed on a Pentium M 1.6-GHz PC with a memory of 256 MB. UML simulates distributions, such as debian Linux and Red Hat Linux, by reading a image data of those distributions, and can perform Ethernet-based telecommunications. Because it is a type of open-source environmental software, anyone can obtain and use it at no financial cost. This allows anyone to easily create a virtual network by running multiple UML units on one PC.

3. System structure

Figure 1 shows this system's structure. This system is an application that can work on a Linux machine. Trainees install network components, adjust system settings, and perform network status checks with the network control screen. Running simulations on the environmental software UML enables trainees to carry out the tasks mentioned above in a safe, easy and relaxed manner because the can do all this virtually on one PC that does not connect to the Internet.

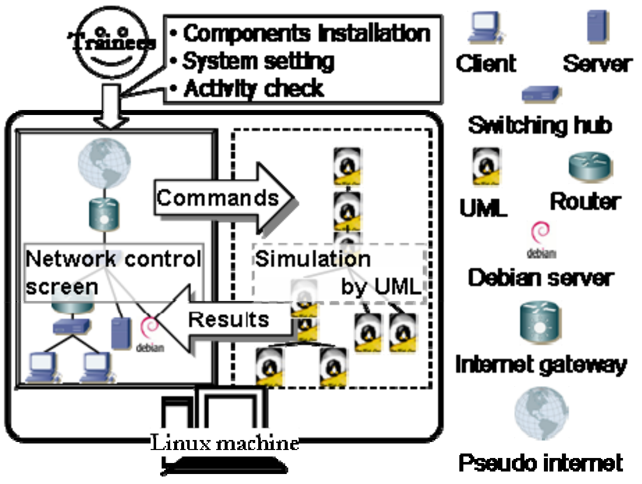


Fig. 1 System structure

3.1 Training for Linux network construction in this system

(1) Linux installation onto server hardware: Trainees can use Linux server machines without actually installing Linux. The work that is most important and on which the most time should be spent in Linux network construction training is system settings, such as TCP/IP settings and server software settings. This system provides trainees with a Linux server without having to install Linux (which takes a lot of time); thus, they can concentrate on their more important tasks.

(2) Network component installation: Trainees install network components easily by mouse operation on a GUI screen. The most important purpose in this work is to establish a connection relation between different hardware components. A large amount of space and much work time are necessary for this purpose in the case of actual hardware, but this purpose is achieved effectively in the case of this system because a virtual setup is employed instead.

(3) System settings: Trainees can manipulate training contents just as with traditional system settings (Table 1) safely and in a relaxed manner. In this system, trainees can train on one standalone PC via virtual networks realized by the virtual environmental software User-mode Linux. This means such tasks can be performed safely because this system does not need to be connect to any actual networks. Furthermore, even if trainees do make serious mistakes for actual Linux servers, such as deleting important setting files, trainees can recover their training environment by replacing them with new units. Therefore, trainees do not need to worry about their actions, thereby reducing the stress of learning.

(4) Network status check: Trainees can analyze their networks by using clients installed by them and by a client function of the pseudo-Internet. Furthermore, they can also check TCP/IP settings and server software settings with a Web browser and clients' mailers. It is also possible for trainees to examine firewall and route settings through a client function of the pseudo-Internet that is different from their network.

3.2 System implementation

3.2.1 Virtual network machinery realized by UML

Table 2 shows the network components and their main specifications that are necessary for training of the contents described in Section 2.1. Trainees can use a conventional Linux server or a debian server as the server component. The former is a Linux server with the software necessary for training fully installed, and it is useful for building networks quickly. The latter is useful for package-management training because it has been evaluated as a good distribution for server use and has a superior package-management function. Because the client is mainly used for checking network operations, we have installed client applications for its purpose on it. In addition, we have prepared a pseudo-Internet and an Internet gateway for package-management training and domain-management training. These two components will be explained in detail in the next section.

We implemented these components by UML. A root-file system is necessary to operate UML as a Linux server or a Linux client. The root-file system involves one file storing a file group that is usually installed in a hard disk. Thus, we installed the applications given in Table 2 to the root-file system. Furthermore, we realized a router by letting UML work as a Linux router and realized a switching hub by a switching-hub application that is attached to UML.

Table 2: Virtual network components

Component	Main specifications
Client	Web browser: dillo Mailer: sylpheed FTP client: gFTP SMB client: xffm Telnet/ssh client: xterm
Server	Web server: Apache Mail server: qmail FTP server: ProFTPD DNS server: Bind SMB server: Samba
Router	Firewall: iptables DHCP: udhcpd 2 Ethernet interfaces
Switching hub	5 Ethernet interfaces
debian server	debian 3.0 (lightly customized version)
Internet gateway	Domain registration DNS service 2 Ethernet interfaces (LAN/WAN)
Pseudo-Internet	Web browser, Mailer, FTP client, Telnet/ssh client (same as client) DNS root-name server debian package server
Network Cable	Ethernet cable

3.2.2 Pseudo-Internet and Internet gateway

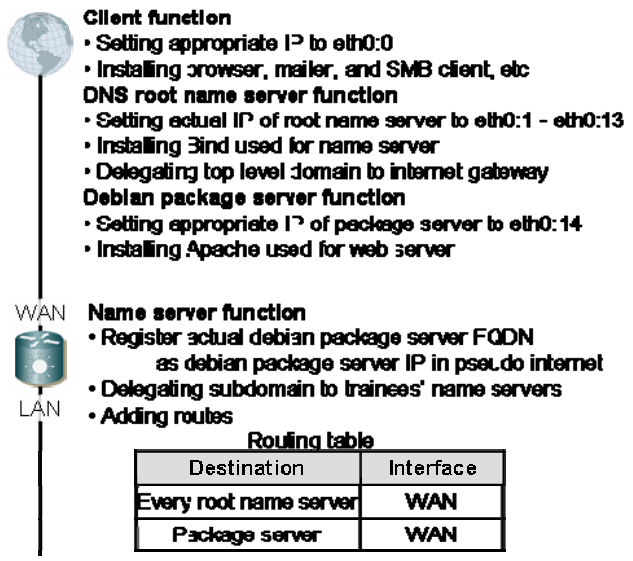
Usually, there are three cases in which the Internet is required in network construction. One is that a Linux server obtains software packages from package servers located on the Internet for installing new software or updating software. Another is that a DNS server manages domains. If a DNS server cannot resolve an inquiry received from a client, that DNS server makes a request about that inquiry at root-name servers on the Internet. The other is that trainees communicate with their network from a different network to check the network status, and this will depend on firewall settings and route settings. These checks are performed using the computers located in campus LANs and their home, which are different segments such as the Internet. We adopted the latter because we think that trainees understand it more easily than the former.

One of this system's purposes is safe training, which is why we created a "pseudo-Internet" as a substitute for the real thing. Trainees use this pseudo-Internet to perform these three types of telecommunication, thus reducing the actual Web's potential negative influence on the trainees' works. In addition, we created an Internet gateway that features a network built by trainees on the LAN side and that has the pseudo-Internet with substitute functions for actual root-name servers, actual package servers, and a client function on the WAN side. Therefore, trainees can train in procedures just as they would in traditional training because they do not need to make any special settings to their virtual machinery. Tasks for checking network status are the same as for traditional works in positioning, except that trainees must recognize this function to be telecommunication of a computer that is actually connected to the Internet.

We implemented the pseudo-Internet by using one UML and the Internet gateway, employing one UML to realize these functions effectively. Loads of the actual root-name servers and the actual package servers are dispersed by using multiple units of machinery. This act, however, is not necessary because this training scale is small. Therefore, we applied only two UMLs for effectively using the CPU and memory.

The root-name server function of the pseudo-Internet must be able to be accessed using IP addresses, just as with actual root-name servers, and that function must record registration of DNS servers that manage top-level domains, too. Therefore, we set the 13 IP addresses of all actual root-name servers to Ethernet interface eth0:1-eth0:13 of the pseudo-Internet by using Linux's alias function, and added routes to the Internet gateway.

Making these settings enabled us to realize the pseudo-Internet effectively by using only one UML. In addition, we assigned a role of the DNS server, which manages top-level domains, to the Internet gateway by setting the pseudo-Internet so that it delegated top-level domain to the Internet gateway. Consequently, trainees can install software packages and construct new sub-domains with procedures the same as those in traditional training by registering package servers and DNS servers they have constructed to this Internet gateway.


Client function
• Setting appropriate IP to eth0:0
• Installing browser, mailer, and SMB client, etc
DNS root name server function
• Setting actual IP of root name server to eth0:1 - eth0:13
• Installing Bind used for name server
• Delegating top level domain to internet gateway
Debian package server function
• Setting appropriate IP of package server to eth0:14
• Installing Apache used for web server
WAN
LAN
Name server function
• Register actual debian package server FQDN as debian package server IP in pseudo internet
• Delegating subdomain to trainees' name servers
• Adding routes
Routing table

Destination	Interface
Every root name server	WAN
Package server	WAN

Fig. 2 Implementation of pseudo-Internet and Internet gateway

4. Execution examples

Figure 3 shows the screen used to construct a virtual network. Trainees install network components appropriately after having understood the role of every part, ensuring the specifications of network that trainees will construct can be filled. Trainees can select network components by pressing the desired buttons located at the upper part of the screen, then set the selected components at the desired positions in the main area simply by dragging them to the desired and dropping them with a click of the mouse.

After finishing component installation, trainees set the installed machinery. Figure 4 shows that trainees are installing the Web server software Apache with the package management command "apt" through a terminal window of the debian server (Fig. 3 (6)). Even in the case of traditional training, trainees perform installation tasks with package management commands through terminal screens in this way. Just after executing this command, in the case of traditional training, packages are obtained from package servers on the Internet and installed. However, in the case of this system, packages are taken from the package-server function of the pseudo-Internet and installed. In this way, the pseudo-Internet enables trainees to perform training in the same way as traditional training but more safely, since work procedures are not influenced by the real Internet.

Figure 5 shows a trainee editing a setting file for the Bind of Linux server (Fig. 3 (7)) with the text editor. This server was delegated the authority of a new domain "example.com" from the DNS server (Fig. 3 (1)) managing top-level domains. In this way, the root-name server functions of the pseudo-Internet enable trainees to easily practice constructing their own domain.

Even if virtual network machinery do not work normally due to operational errors with regard to package management and editing errors in setting files, which can happen when building a network, trainees can retry their tasks easily by either deleting abnormal machinery and installing new units or by rebooting the system.

Figures 6 and 7 show that a trainee tried to view a Web page served by a debian server (Fig. 3 (6)). Figure 6 shows that a trainee succeeded in opening the Web page by inputting "www.example.com" into the URL column of a client (Fig. 3 (4)). The trainee tried the same thing and obtained the same result in another client (Fig. 3 (8)), too. In this network, one client (Fig. 3 (4)) performs name resolution by the DNS server (Fig. 3 (5)), and the client (Fig. 3 (8)) performs name resolution by the DNS server (Fig. 3 (7)). The server (Fig. 3 (7)) performs name resolution of "www.example.com" by its own effort because it is managing "example.com," while the server (Fig. 3 (5)) performs name resolution of "www.example.com" by using the root-name server function of the pseudo-Internet. As a result, both clients (Fig. 3 (4), (8)) could find the IP address of the debian server, thus the trainee could read the Web page from them. On the other hand, Fig. 7 shows that a trainee failed in viewing the same page via the pseudo-Internet (Fig. 3 (2)). Although that trainee input the same URL into the URL column (Fig. 7 (1)), the browser displayed an error message (Fig. 7 (2)). In fact, this network's specifications were that the server (Fig. 3 (6)) accepts Web access attempts only from hardware connected to the LAN. Therefore, a trainee is not able to view Web pages through the pseudo-Internet, so this error indication is the correct result. Because the trainee performed all TCP/IP settings correctly and installed Apache on the debian server, constructed his or her own domain "example.com," and set the firewall for the Internet gateway correctly, the trainee obtained these correct results.

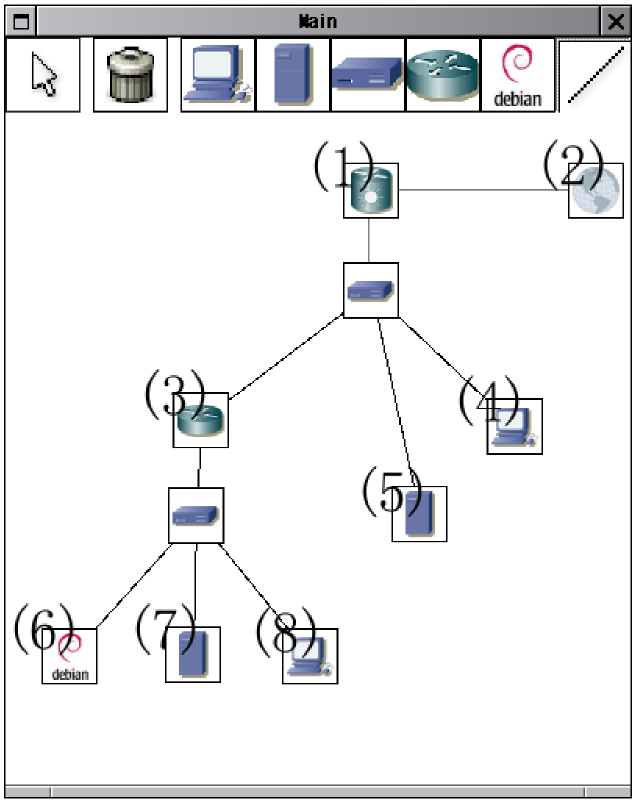
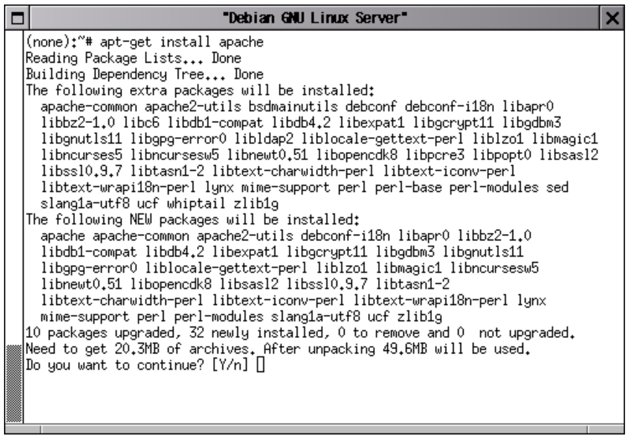
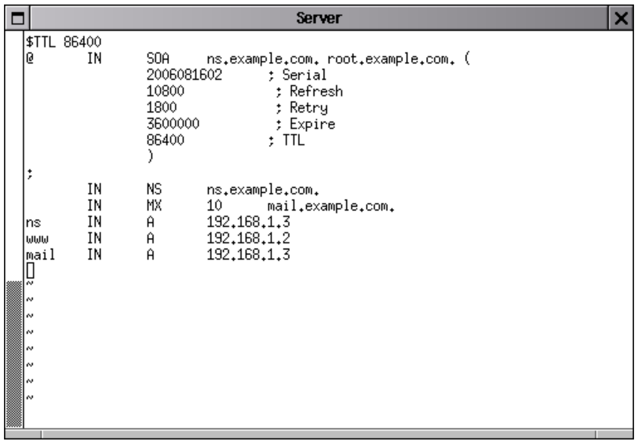


Fig. 3 Network component installation



```
(none):~# apt-get install apache
Reading Package Lists... Done
Building Dependency Tree... Done
The following extra packages will be installed:
  apache-common apache2-utils bsdmainutils debconf debconf-i18n libapr0
  libbz2-1.0 libc6 libdb1-compat libdb4.2 libexpat1 libgcrypt11 libgdbm3
  libgnutls11 libgpg-error0 libldap2 liblocale-gettext-perl liblzo1 libmagic1
  libncurses5 libncursesw5 libnewt0.51 libopencdk8 libpcre3 libpopt0 libsasl2
  libssl0.9.7 libtasn1-2 libtext-charwidth-perl libtext-iconv-perl
  libtext-wrapi18n-perl lynx mime-support perl perl-base perl-modules sed
  slang1a-utf8 ucf whiptail zlib1g
The following NEW packages will be installed:
  apache apache-common apache2-utils debconf-i18n libapr0 libbz2-1.0
  libdb1-compat libdb4.2 libexpat1 libgcrypt11 libgdbm3 libgnutls11
  libgpg-error0 liblocale-gettext-perl liblzo1 libmagic1 libncursesw5
  libnewt0.51 libopencdk8 libsasl2 libssl0.9.7 libtasn1-2
  libtext-charwidth-perl libtext-iconv-perl libtext-wrapi18n-perl lynx
  mime-support perl perl-modules slang1a-utf8 ucf zlib1g
10 packages upgraded, 32 newly installed, 0 to remove and 0  not upgraded.
Need to get 20.3MB of archives. After unpacking 49.6MB will be used.
Do you want to continue? [Y/n] 
```

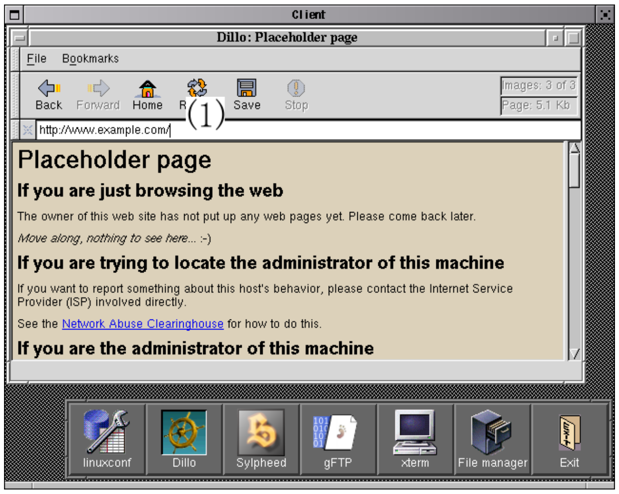
Fig. 4 Installing Apache



```
$TTL 86400
@       IN      SOA     ns.example.com. root.example.com. (
                2006081602      ; Serial
                10800           ; Refresh
                1800            ; Retry
                3600000         ; Expire
                86400           ; TTL
                )
;
        IN      NS      ns.example.com.
        IN      MX      10      mail.example.com.
ns      IN      A       192.168.1.3
www     IN      A       192.168.1.2
mail    IN      A       192.168.1.3

~
~
~
~
~
~
~
~
```

Fig. 5 Setting Bind



Placeholder page

If you are just browsing the web

The owner of this web site has not put up any web pages yet. Please come back later.

Move along, nothing to see here... :-)

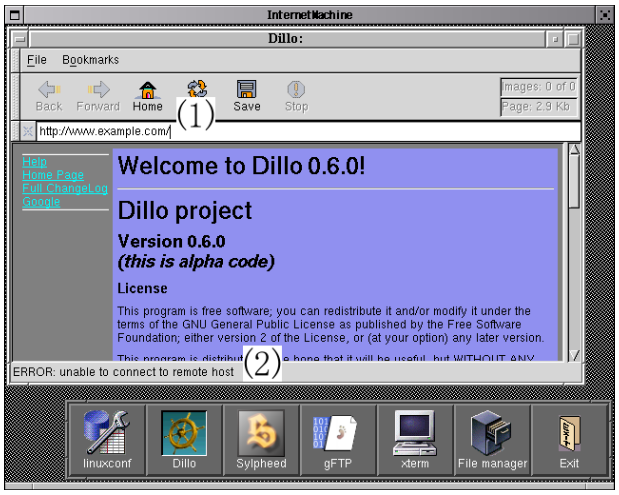
If you are trying to locate the administrator of this machine

If you want to report something about this host's behavior, please contact the Internet Service Provider (ISP) involved directly.

See the Network Abuse Clearinghouse for how to do this.

If you are the administrator of this machine

Fig. 6 Web browsing from a machine placed in the LAN



This program is free software; you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation; either version 2 of the License, or (at your option) any later version.

ERROR: unable to connect to remote host

Fig. 7 Web browsing from pseudo-Internet

5. Evaluation experiences

The aim of this system is to provide learners with an environment for learning and practicing Linux network construction in a safe, easy, and relaxed manner. We performed a questionnaire-based survey of the subjects and measured the boot-time of a network component and CPU use rate to confirm the system's effectiveness. We used a Pentium M 1.6-GHz PC with 256 MB of memory. If this system works on this hardware, it can be used for educational purposes.

5.1 Questionnaire and results

We distributed the eight-point questionnaire shown in Table 3, with subjects using the following scale for evaluation: 5 - greatly; 4 - a lot; 3 - somewhat; 2 - not much; 1 - not at all. Also included was space for free comments. Questionnaire results are shown in Table 3, which also sets out the gist of comments written in the open comment section. Subjects were 11 students, who have experience with Linux network construction. The reason why we selected subjects who not beginners in Linux network construction is that this evaluation does not aim to measure learning effects, but to evaluate the effectiveness for Linux network construction training based on their previous experience. Subjects set up a domain and installed Apache to the debian server with apt commands, and opened Web pages of the debian server to the pseudo-Internet.

Table 3: Question items, average scores and comments

Question	Score
Question 1: Usually Linux network construction training is performed with real networks. How significant is this method that training is performed with virtual networks?	5.00
Question 2: How much did this system help you train in TCP/IP settings?	4.73
Question 3: How much did this system help you train in domain management?	4.64
Question 4: How much did this system help you train in Apache settings?	4.64
Question 5: How much did this system help you train in debian server usage, such as package management?	4.55
Question 6: How much did the client function of the pseudo-Internet help your tasks?	4.82
Question 7: How much did this system help you train in Linux network construction?	4.82
Question 8: How good is system's operability?	3.64
Gist of comments	
Comment 1: I think this system is superior in regards to space to install components, expense to introduce components, and the trouble taken to prepare components.	
Comment 2: I think this system is superior in the point that trainees can train without negative influence from actual networks.	
Comment 3: I could train easily because this system showed the whole structure of a network to me.	
Comment 4: I think that straight cables and cross cables are required in the training of this system.	
Comment 5: I think that popup indications, such as names and explanations of components, are required.	
Comment 6: I could not easily recognize the correspondence between a network component and its control window easily.	
Comment 7: I think this system should have an "undo/redo" function.	
Comment 8: I think that this system should have a function to save networks built by a trainee.	
Comment 9: I think this system should have a function to display guides or hints when trainees are at a loss for what to do.	
Comment 10: I think this system should have a function to display indications for pointing out setting mistakes	

5.2 Consideration of questionnaire

Because this system received many affirmative results, with exception to its operability, we confirmed that it is effective for training in Linux network construction.

The score for Q. 1 proved that this study was significant for Linux network construction training. Comments 1 and 2 mention that low costs (space, money, trouble) and safe training are advantages for trainers, and Comment 3 mentioned that the indication of the whole network structure helped trainees' understanding.

Scores for Qs. 2-6 are evaluations of system setting training, which is the most important purpose of this study. That these scores are high proves that this system is indeed very beneficial for training. The score for Qs. 3, 5, and 6 are evaluations of the training contents, the procedures of which are the same as traditional ones but whose structures are different from the traditional ones. These evaluations are also high, proving the utility of the pseudo-Internet.

The score for Q. 7 proves that a series of works -- network component installation, system settings, and network status checking in this system -- are useful for Linux network construction training. The results proved that this system has strong potential for becoming one of the best choices for a training method in Linux network construction training. Furthermore, Comment 4 mentioned that a shortcoming of this system is that there were not concepts of straight cables and cross cables. Because this system established an important point for system setting functions, network component installation function provided a supporting role for that function. Therefore, we will reinforce such a function in future.

On the other hand, we confirmed that this system has problems with operability (score for Q. 8). From Comments 5 and 6, we understood that it is necessary to improve the system's design to enhance its operability. Therefore, we added the above and development of the functions mentioned in Comments 7 and 8 as future works.

Comment 9 mentioned that functions which display hints or guidance for assisting training are important. Because this study's purpose is to provide a training environment, we have not developed such a function. However, we do consider such a function important for more effective training. Therefore, we would like to add it to our future works.

5.3 Measuring execution time and system use rate

We measured the boot time of network components and CPU use rate to confirm the system's effectiveness.

As for average times over 10 trials, a client spent 15.7 seconds, a router spent 9.7 seconds, a server spent 7.9 seconds, a switching hub spent 0.2 seconds, and a debian server spent 12.8 seconds.

We measured the CPU use rate when performing the same tasks as the evaluation experience in the outline shown by Table 4. Figure 8 shows that result. The CPU use rate rose temporarily when network machinery was booted and Apache was installed; however, it did not sustain this performance in most cases. That is, performance of this CPU is enough in the case that response speed strongly influences training comfort, such as when editing setting files and checking network status.

We consider that this system is practical because it is superior to an actual machine in boot time, and the CPU use rate is not so high that trainees are not irritated by delays when performing their tasks.

Table 4: Time table

Time	Works
02:37	Enabling the pseudo-Internet
02:39	Booting a router
02:41	Booting a debian server and installing Apache
02:49	Booting a client A and performing Web telecommunication to the debian server with an IP address
02:51	Booting a server and constructing a domain
02:54	Performing Web telecommunication with the debian server from client A with FQDN
02:55	Booting client B and performing Web telecommunication with the debian server from client B with FQDN
02:57	Performing Web telecommunication with the debian server from pseudo-Internet

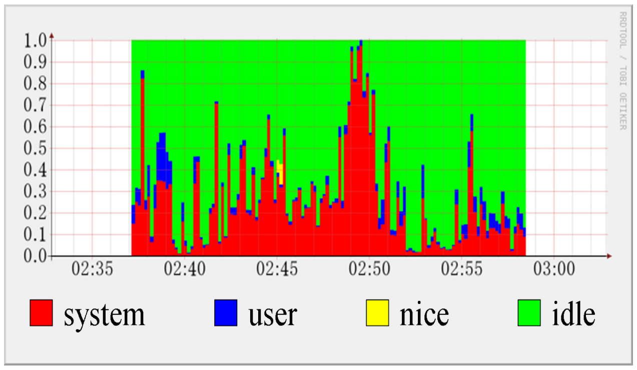


Fig. 8 CPU use rate

6. Conclusion

In this study, we created a system that offers a superior training environment for Linux network construction by using the virtual environment software User-mode Linux. In evaluation experiments with eleven students, we confirmed the effectiveness of this system for training in Linux network construction. Some problems remained, however, with the user interfaces. We think that this system has strong potential for becoming a new choice for a training environment in Linux network construction training, because the system's features are greatly different from those of traditional training environments. Furthermore, in comparison with a traditional method using real networks, this system has features whereby actual Linux installation is not performed and the physical tasks of network component installation are not executed, making it safer, easier, and less stressful for trainees. We think that this kind of training environment is useful for the case that students study network construction alone or where lectures are given under certain limitations such as training time and expense.

Acknowledgments

This research was supported in part by the 21st century COE program "IMI for social information infrastructure" by MEXT and a Grant-in-Aid from the Scientific Research Foundation.

References

[1] VMware - Virtualization Software: http://www.vmware.com/.
[2] The User-mode Linux Kernel Home Page: http://user-mode-linux.sourcefor_ge.net/.
[3] Tateiwa, Y., et al., "Development of a system for education by visualizing network processing based on virtual environment software," IPSJ SIG Technical Reports 2006 - CE - 85 (2005) (in Japanese), 7-14.
[4] Toguro, M. and Kimura, M., "Development of Education-Oriented Network Simulator," The 65th National Convention of IPSJ (2003) (in Japanese), 4-273-274.
[5] Sasaki, K., et al., "Development of HTTP-FUSE-KNOPPIX Base for Server Application Learn Environmental System," The 68th National Convention of IPSJ (2006) (in Japanese), 1-35-36.
[6] The Network Simulator-ns2: http://www.isi.edu/nsnam/ns/.
[7] OPNET: http://www.opnet.com/.
[8] OMNET++: http://www.omnetpp.org/.

Yuichiro TATEIWA received the B.E. degree in Intelligence and Computer Science from Nagoya Institute of Technology and M.S. degree in Human Informatics from Nagoya University in 2002 and 2004. He is interested in education support, network system, and open-source.

Takami YASUDA is a professor in the Graduate School of Information Science at Nagoya University. Yasuda received his BE and ME in electronic engineering form Mie University in 1982 and 1984. He received his Ph.D. in information engineering from Nagoya University in 1989.

Shigeki YOKOI became a Professor of School of Informatics and Sciences in Nagoya University in 1995. In 2003, he became a Professor of Graduate School of Human Informatics in Nagoya University. Then he became a Professor of Graduate School of Information Science in Nagoya University in 2004.

He is engaged in the research on the relationship between technology and real society within our electronic society. He is interested in researching the processes inherent in the creation of new our society, making use of Internet and image media. To this end, we must understand technological innovation precisely, and investigate in detail its applications to actual society.
His current main research topics are as follows.
(1) Development of PC software for senior citizens:
(2) Development of education support software for network engineers:
(3) Development of basic and allocation technologies of Web-based 3D CG: