published as: J. Hu, D. Cordel, Christoph Meinel: Virtual Machine Management for Tele-Lab "IT-Security" Server;
Proceedings of the 10th IEEE Symposium on Computers and Communications (ISCC 2005);
La Manga del Mar Menor, (Cartagena, SPain), 2005, pp. 448- 453, ISBN: 0-7695-2373-0.

Virtual Machine Management for Tele-Lab "IT-Security" Server

Ji Hu, Dirk Cordel and Christoph Meinel
The Hasso-Plattner Institute, University of Potsdam, Germany
{ji.hu, cordel, meinel}@hpi.uni-potsdam.de

Abstract

Tele-Lab "IT-Security" Server is a new concept for practical security education. The Tele-lab server builds a virtual security laboratory using lightweight virtual machines and features a pure web user interface. Thus, students can easily exercise security over the Internet. Running such a security laboratory on the Internet is difficult. Its infrastructure is subject to misuse and crash due to the nature of security tasks. The management of virtual machines is a crucial component which enables running the Tele-Lab server on the Internet. It effectively saves resources and improves security and reliability of the virtual laboratory. This paper briefly reviews the architecture of the Tele-Lab "IT-Security" server, describes its virtual machine management in detail, and presents some experiential and experimental results.

1. Introduction

Passing hands-on experience plays an important role in today's security education because students need learn to apply security concepts in a real environment. It has been recognized that laboratory exercises must be included into our teaching syllabus. To this end, many security tutoring and training systems, which concern practical experience, are developed in academia and industry.

Due to the problematic nature of security tasks, there are several challenges in design and implementation of security exercises. Those challenges are respectively described as following.

- Running practical exercises implies an **insecure** system environment. For finishing tasks, students have to be given super-user rights of systems in most cases. This introduces a risk of misuse of system by a malicious user.
- With privileged operations, user activities are hard to expect and to control. The laboratory environments become **unreliable**. Failures or errors highly possibly lead to corruption of systems.
- Preparation and maintenance of laboratory systems are **effort-intensive**, which involve a lot of manual administration and recovery work.

Aiming at the challenges above, Tele-Lab "IT-Security" tries to develop a computer-aided tutoring system which helps students gain security experience. The training content covers various aspects of IT security, including cryptography, digital certificates, secure Email, authentication and security scanning. The basic idea is that we integrate practical, interactive security exercises into the tutoring system and automatically manage them in a Linux system.

Tele-Lab has a web-based structure. The tutor is a web server which presents teaching contents, guides user in learning and manages exercises. The Linux system is prepared as a small "virtual" laboratory with various open-source security tools. Scripts are used to implement interactive exercises. By invoking such scripts, the tutor prepares tasks. After a student completes tasks in the Linux system, the tutor evaluates results. In this way, real-life and interactive scenarios are realized. The first Tele-Lab system we developed is a standalone system [6], which puts both the tutoring server and the Linux system on a computer. To eliminate inconvenient recovery from failures happened in learning, we developed the Tele-Lab "IT-Security" CD [5]. It integrates the entire system to a Knoppix live-CD which has ability to run a complete Linux system on the CD instead of a hard disk. Thus, any failures and errors can be easily recovered by rebooting system. A detailed description is found in [5].

The motivation of the Tele-Lab "IT-Security" Server is to move the system on the Internet. Its concept and framework can be found in [4]. The distinct feature of the server is that laboratory environments are provided with virtual machines [3] instead of native Linux systems. The virtual machine (VM) is a software application to simulate a real machine. It is possible to run multiple VMs on a host and connect them to a network. A user can be assigned a privilege right on a VM and perform exercises on it. These VMs can be easily managed, e.g. fast recovery from failure. Thus, labora-

tory resources can be shared by remote users in a very economic and automated way.

Running Tele-Lab Server on the Internet raises some new requirements, e.g. providing enough VM resources, higher reliability and security, and good quality of service. This paper will discuss the management of Tele-Lab Server. The management functions include virtual machine status monitoring, user activity monitoring, virtual machine resource management, and network and security management. The management is a very crucial part which satisfies the requirements above and makes applying the Tele-Lab server in practice possible. Besides, in order to evaluate the system performance of the Tele-Lab server, we integrated a web based performance monitor. It can collect system status information and performance data of the host server and virtual machines. These statistics data can be shown in graphs and used for further analysis.

The rest of the paper is structured as follows: Section 2 reviews the architecture of the Tele-Lab "IT-Security". Then, Section 3 introduces the laboratory management of the Tele-lab server. Experience and some experimental results are presented in Section 4. In the end, Section 5 concludes the paper.

2. The Tele-Lab Server Architecture

We briefly review the architecture of the Tele-Lab "IT-Security" server. More detailed descriptions can be found in [4]. As shown in Figure 1, the Tele-Lab server consists of five parts: a tutor, a knowledge repository, user data, the user interface and a virtual lab.

The functions of the tutor are similar to those of the previous systems except that exercise scripts are moved to the VM. The tutor has to manage them via a SSH connection to that VM. Teaching materials in the knowledge repository include tele-lectures [11], lecture notes about security concepts, multimedia demos about security tools' usage. Learning performance is stored in the user data, which is useful to build a student model.

The user interface to a VM is implemented using a remote access solution from the Virtual Network Computing (VNC) [10]. The VNC developed a remote frame buffer (RFB) protocol which collects inputs from a remote client, encodes desktop images on the server, and sends them to the client. VNC also provides a Java-applet client which embeds a desktop viewer in the browser. For finishing exercises, a user only needs a standard browser instead of dedicated client software.

If a student logs in, the virtual lab will provide him/her a laboratory environment. This environment is built with the User-Mode Linux virtual machines [1]. User-Mode Linux is special virtual machine software which runs virtual operating systems (Linux) on a native Linux system. These VMs run as user-level application on the host as any regular computer program. Therefore, their destruction does not result in negative effects on the host. Each VM is set up to a baseline configuration and prepared with open-source security tools, such as OpenSSL, the Nmap port scanner, the Nessus security scanner, John-the-Ripper (a password cracker), etc. These tools are needed by a user to finish his/her tasks.

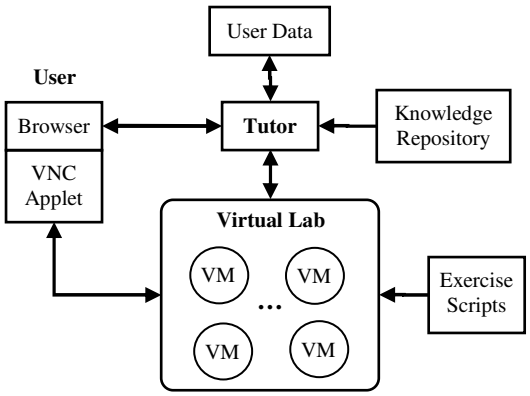

Figure 1. System architecture

3. Virtual machine management

As mentioned in the introduction section, the purpose of the management is to improve security and reliability of the Tele-Lab infrastructure and to tune its performance for best utilization of resources. The virtual machine management is involved in three aspects: the virtual machine resource management, security management and the virtual lab management.

3.1 Virtual machine resource management

Most important factors which affect the performance of a virtual machine include processor, memory and disk resources. Compared to others VM variants, User-Mode Linux is a resource-friendly and manageable virtual machine implementation. It has been chosen for building virtual workstations in the Tele-Lab. The resource management focuses on detailed configurations of the User-Mode Linux VMs and improving performances of both VMs and the host.

It is necessary to isolate the performance of a VM from those of other VMs and the host system. We do not want to see a greedy application on a VM swallow most host resource and spoil its performance. First of all, we degrade the priority of the VM processes by adjusting their *nice* values higher. Thus, VM processes are allocated less CPU time than other host processes. Secondly, the host machine is installed with a SMP Linux kernel which is running on an Intel Pentium 4 Processor with hyper-threading which in effect adds another logical CPU that shares the same resources

with the physical CPU. An application might get a 30 percent performance improvement in the test [7]. In this way, the performance impact of running many VMs is degraded.

When we start a VM, memory resource can be allocated to it by specifying its physical memory size. Instead of using the same size of the memory chunk, the VM gets memory from the virtual memory files in the directory */tmp*. This directory is mounted as a dynamic RAM based file system ("tmpfs"). The "tmpfs" only uses the physical memory it needs. If it's unused it will be swapped out to disk. In this way, we can provide VMs plenty of memory by creating a large /tmp file system using "tmpfs". E.g. we have 2GB RAM on our host, we can create a 6GB /tmp file system as virtual memory for VMs by swapping out the extra 4 GB memory to a swap partition. In the Tele-Lab, we have 30 VMs. Each of them is assigned 64MB memory. With this scheme, only 1078 MB RAM space is in deed used, which is 56% of total space (30 x 64MB = 1920 MB) for VMs.

User-Mode Linux provides only a user mode OS kernel. In order to run a VM, we need an image in which a standard Linux filesystem is installed. UML saves this image in an ordinary file on the host's hard disk. Therefore, the image file can be created, copied or removed. By doing so, it is possible to manage virtual machines as normal applications, i.e. they can be easily started, killed or recovered on demand. We hope to build a lightweight VM. This means the VM image file should be small enough. Thus, more processor and memory resources can be saved. More VMs can run on the host. Also, a small image file helps to make the VM management procedure faster. For instance, a VM can be recovered faster if it fails. Therefore, we allocate each virtual machine a small hard disk space (300 MB) which is enough to contain basic OS system files.

User-Mode Linux VMs have a feature that multiple VMs can share an image using a so-called copy-on-write (COW) layering device. Based on this device, only a complete root image file is needed. This image file is protected by setting it read-only. Each VM is started from its COW file. A COW file only records changes to the original image during runtime. Since these COW files are small (about 20b-200kb), a lot of disk space is saved. For example, in our implementation, the space needed for running 30 VMs is no more than 306 MB which is merely 3.4% of the total disk space (30 x 300MB = 9000MB) we get.

Furthermore, considering a VM might need more space for installing more applications in future. We divide its file system into two parts: a native part and an extended part. The native one is a local file system in the image file mentioned above. Only the Linux kernel and system files are included. Big applications, e.g. graphics applications, are installed in the extended file system which is imported from a directory on the host. The extended file system is mounted each time a VM is booted. Hence, the actual size of a VM which we manage becomes small.

3.2 Security management

The security of the Tele-Lab server concerns two points: first, whether a VM is harmful to the security of the host; second, whether it compromises production networks.

Inside a VM, a user can have a super-user privilege, but this VM is running as the application of an ordinary user on host. It is hard for a VM user to gain privileged access to the host system by exploiting his/her right in the VM. Besides that, a VM can be run in the SKAS (Separate Kernel Address Space) mode [2]. In this mode, the VM kernel space is protected and the host space is invisible to the VM processes, those processes are forbidden breaking out of the VM to the host space. Therefore, a VM would not affect security of the host if we run it in a safe mode.

The Tele-Lab server connects VMs to the Internet which raises a big security concern. In this context, the user activities on a VM must be restricted within the scope of the Tele-Lab server. I.e. a user on a VM is not allowed to launch any connections to external networks or other VMs. First, we build a virtual network with a UML virtual switch. This switch is very special. It connects each VM to a central node (i.e. the host). Any traffic from/to a VM must be relayed by this node. Thus, there is a chance to install a firewall (the Netfilter [9]) on this node to filter and to control all traffic in the Tele-Lab server. The firewall functions include the following aspects. First, it guarantees that a VM is exclusively assigned to a user. This can be done by a dynamic firewall configuration. If a user logs in, a new rule is inserted to the firewall's rule set. It defines that only those connections from current user's IP address can be forwarded to a VM. After the VM is reclaimed or recovered, this rule will be discarded. Second, irrelevant connections to the Internet, except those VNC connections, are disabled. Finally, traffic between both VMs is stopped.

3.3 Lab management

The lab management is responsible for monitoring the Tele-Lab system and improving its reliability. Also, users are monitored. Their inappropriate use of the VM resource is detected and prevented. Five main components are included in the management system (see Figure 2). The VM assignment table is a data structure which records actual states of each running VM. The VM detection module detects and reports errors of the VMs in time. The VM administration module is

responsible for starting, stopping or recovering VMs in specific circumstance. The user detection module monitors user activities on a VM and identifies abnormal events. The user notification module informs a user about events on his/her VM.

Information in the **VM assignment table** indicates which VMs have been assigned and to whom, which are free and which are failed and need recovery. The VM administration module updates this table at each time when a VM is assigned or reclaimed or some errors are found. The information in the table is very important for assigning and managing laboratory resources. Thus, it is the basis of the lab management system. The structure of the table is simple. Each active VM has an entry in the table. The entry includes the data of a VM's number, user name and his/her IP address, and mode.

A running VM is in one of three possible modes: "free", "assigned" and "recovered". A VM is "free" when it is ready to be assigned to a new user. If none of free VMs is available, a user can still login and have chances to apply for one when he wants to work on an exercise. A VM is "assigned" means it becomes a dedicated workstation for its user and his/her name will be written to its record. The "recovered" mode is a transitional status of a VM. If a failed machine is detected, it will be marked as "recovered". Then, its mode will be changed into "free" after it has been started again.

Basic tasks of the **VM administration module** include assigning, reclaiming and recovering VMs. It selects a free VM from the VM assignment table and assigns it to a new user. Afterwards, this VM's mode is changed to "assigned" and its entry in the table is updated. If failures on a VM are detected or a user logs out, the VM is reclaimed and a recovery procedure is launched. The VM will be restored to a default setting and restarted (the VM's processes are killed and its image file is replaced with a default copy). Recovery of VMs prevents a user from working on a defected machine. In cases of failure, a new VM is immediately assigned to the user. The user can continue exercises on a new VM in short time. Besides that, the administration module gives a user a chance to request a new VM in case that he/she finds his/her one crashed.

The VM administration module integrates a VM control tool, named *uml_mconsole*, from the User-Mode Linux utilities package. With this console, it is possible to manipulate VMs directly from the host. For example, command "reboot" will restart a VM; command "halt" will kill a VM immediately. If a VM crashed, the original file system is still working, and the management module only needs to replace its COW-file with a new copy and start this VM again. After that, the VM will notify the administration module and its running mode will be set as free.

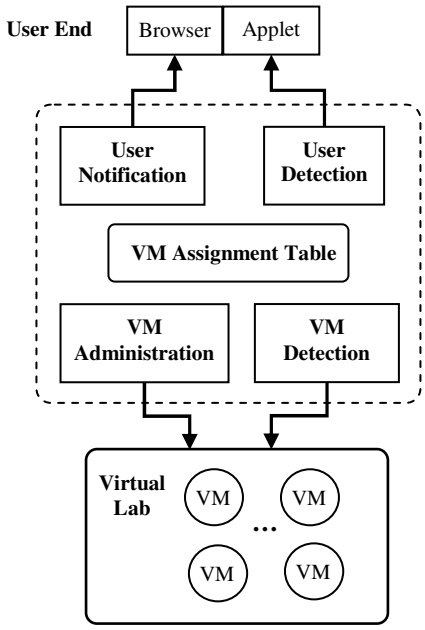

Figure 2: The lab management

The **VM detection module** monitors the status of each VM and reports errors of the VM to the administration module in time. We have two strategies to implement monitoring. One way is to install a real-time monitor on the VM, which sends status information to the host. The local monitor can report precise and detailed status information, but it can be easily interrupted by the VM user. The other way is to periodically (e.g. every three minutes) scan services on the VM. These services are necessary for a user to work on a VM (e.g. VNC services) and for the tutor to contact a VM (e.g. SSH and email services). If connections to a VM or those services fail, the VM is proved to be failed. It is an effective way to find out most errors in a general situation though some errors might be missed. This method is simpler and more reliable than the first strategy. Therefore, service scan is applied in the Tele-Lab server. In addition, in case that some critical errors are missed, a user has a chance to report errors and move onto a new VM by clicking a button on his/her web page.

A host can run only certain number of VMs because system resources are limited. The **user detection module** helps to prevent unnecessary occupation of a VM. User activities are monitored on the VM by tracking user's keyboard and mouse inputs. This function can be realized by a monitor program either at the desktop access client or on its server. In implementation, we modified the VNC applet code and inserted a hook which detects keyboard and mouse actions in the viewer window. In this way, the user's idle time on the VM can be measured. Then, we can count real idle time. If it exceeds a predefined threshold (e.g. half an

hour), we can conclude that the user won't continue learning. The administration module will therefore reclaim her/her VM and let him/her log out.

The **user notification module** is necessary to inform a user about special events in time. It reports a user the latest status of his/her station. E.g. if the detection module finds a failure, the notification module will inform the user that his/her VM needs a recovery; if a free VM is available, it will notify the user to continue exercises on the new one. A web server only passively responds to user requests because the HTTP is not a connect-orient protocol. The notification would not reach a browser if the user does not raise any request. Therefore, we have a small frame in the browser window. It will refresh each 15 seconds, to get the message about the events of the VM.

4. Application and Experience

This section will discuss some educational and technical experience of the Tele-Lab server, which has not been addressed in [4]. Up to now, the Tele-Lab server has integrated two pilot chapters which are ported from the Tele-Lab CD version. Those chapters include a topic about password security and a topic concerning secure email with digital certificates. In the former one, a user will get to know about authentication concepts and how passwords in Linux are protected by hashing and encryption. Then, a typical relevant tool is introduced, i.e. *John the Ripper* (a password cracker). Next, an exercise is prepared. The tutor generates a UNIX *passwd* file which contains random passwords. This file will be passed to the user's VM via SSH. The user must decode passwords in the file by applying the John-the-Ripper cracker on the VM. The user has to submit correct answers to the tutor through the web interface. The tutor evaluates answers and saves the result to his/her user data. The second topic has a similar structure. After explaining some essential digital certificate concepts and usages, some interactive tasks are prepared on the VM. A user should practice signing and encrypting messages with digital certificates using the *Mozilla* e-mail client. We tested the Tele-Lab server with some students in our university, who did not take security courses before. The feedback from the test shows that they can understand theoretical concepts and can complete exercises by their own. If students are unfamiliar with Linux, necessary help information is available in exercises. Situation becomes easier if they already know Linux. Therefore, the use of Linux as a working environment might bring some difficulties, but it is not a big barrier in learning.

Evaluation work has been done for measuring performance of virtual machines. In the evaluation, we install the whole Tele-Lab server on a modern PC. This host PC is equipped with a 2.8GHz Intel Pentium IV CPU which simulates virtual SMP (Symmetric Multi-Processors) by hyperthreading, 3GB RAM, an Intel Pro 1GBit Ethernet interface and a Samsung SV1203N 5400 RPM IDE harddisk. The host operating system is Linux 2.4.26 for i686 with SMP support. We patched the Linux kernel so that the UML "SKAS" mode is enabled. The operating system of the virtual machine is Linux 2.4.26, which is equipped with 64M RAM and 300MB disk space. There were thirty virtual machines running on the host. We measured system performance with a benchmark tool, the *lmbench* program [8]. Its latest version 3.0-a3 was applied. Here, we mainly focus on the benchmark of processes, memory and file system and compare them with some native systems. The results are shown in the following tables (reference data of Linux1.3/i686 and Solaris/i686 come from [8]).

Table 1 System specifications

Name	OS	CPU (MHz)
Linux 2.4	Linux 2.4.26	Intel Pentium II 350
Linux 1.3/i686	Linux 1.3.37	Intel Pentium Pro 167
Solaris/i686	SunOS 5.5.1	Intel Pentium Pro 133
Tele-Lab VM	**UML 2.4.26**	**Virtual processor**

Table 2 Process creation time (milliseconds)
(smaller is better)

System	fork & exit	fork, exec & exit	fork, exec sh -c & exit
Linux 2.4	0.4	1.5	7
Linux 1.3/i686	0.5	7	17
Tele-Lab VM	**4.5**	**7.5**	**16.9**
Solaris/i686	4.5	22	46

Table 3 Memory bandwidth (MB/s)
(bigger is better)

System	Memory read (8M)	Memory write (8M)
Tele-Lab VM	**2602**	**1596**
Linux 2.4	313	170
Linux 1.3/i686	205	56
Solaris/i686	159	71

Table 4 File system latency (microseconds)
(smaller is better)

System	FS	Create	Delete
Tele-Lab VM	**EXT2**	**78**	**34**
Linux 2.4	EXT3	167	235

Linux 1.3/i686	EXT2	751	45
Solaris/i686	UFS	23k	7k

Compared to other native Linux systems, our VM's performance is much better except for the process ability. The reason of the slow process ability is that the tracing thread in the User-mode Linux brings a performance bottleneck because more context switches are needed for each system call. However, virtual machines utilize most resources of the host and inherit better performances. Therefore, the performance of a virtual machine is enough for most common Linux applications. E.g. some graphics programs in the virtual machine, like the Mozilla web browser, are running smoothly and fast in the test.

In normal use, students would seldom corrupt their VMs, but if a user misuses his/her VM, some resources cannot be available for other users. E.g. if a user logs in or logs out frequently, his/her VM is always in the "recovered" mode, which causes a denial of service attack. In addition, virtual security laboratories cannot completely replace the role of conventional security laboratories. E.g. security on Windows and other systems is not addressed. Therefore, integration of Windows security topics will be a part of our future work.

5. Conclusions

In this paper, we have reviewed the architecture of the Tele-Lab "IT-Security" server, described in detail its virtual machine management and presented some experience and results. The Tele-Lab server is a novel concept for providing practical security experience on the Internet. It is built with lightweight virtual machines and features a pure web user interface. The access to the laboratory and use of security tools are available via a browser. By this means, students can exercise security in a real laboratory environment instead of a limited simulation system.

The management of virtual machines enables running Tele-Lab on the Internet. The resource management of virtual machines optimizes the utilization of limited host resources so that more virtual machines with a good performance can be provided. The lab management provides high reliability. By that, critical errors can be detected and automatically recovered in time. Moreover, by network and security configurations, user activities are tightly limited in a secure scope and misuses are prevented from compromising production networks.

Therefore, the Tele-Lab "IT-Security" server and its virtual machine management prove the possibility that expensive security laboratory exercises can be moved to the Internet by an economic, safe and reliable means.

6. References

[1] J. Dike, "A User-mode Port of the Linux Kernel", in *Proceedings of the 4th Annual Linux Showcase & Conference*, Usenix, Atlanta, USA, 2000.

[2] J. Dike, "Reducing SKAS Mode", available from http://user-mode-linux.sourceforge.net/skas.html, 2004.

[3] R. P. Goldberg, "Survey of Virtual Machine Research", *IEEE Computer*, June 1974, pp. 34-45.

[4] J. Hu and Ch. Meinel, "Tele-Lab IT-Security: A Means to Build Security Laboratories on the Web", in *Proceedings of IEEE AINA* 2004 Fukuoka (Japan), 2004.

[5] J. Hu and Ch. Meinel, "Tele-Lab "IT-Security" on CD: Portable, Reliable and Safe IT Security Training", *Computers & Security*, Volume 23, Issue 4, Elsevier, June 2004, pp. 282-289.

[6] J. Hu, M. Schmitt, Ch. Willems and Ch. Meinel, "A Tutoring System for IT Security", in *Proceedings of the 3rd World Conference in Information Security Education*, Monterey, USA, 2003, pp. 51-60.

[7] "Hyper-Threading Technology", available from http://www.intel.com/techtrends/technologies/hyperthreading.htm, 2004.

[8] L. McVoy and C. Staelin, "Lmbench: Portable Tools for Performance Analysis", in Proceedings of the USENIX Annual Technical Conference, Berkeley, 1996, pp.279-294.

[9] Netfilter, "Netfilter and Iptables", available from http://www.netfilter.org/, 2004.

[10] T. Richardson, Q. Stafford-Fraser, K.R. Wood and A. Hopper, "Virtual Network Computing", *IEEE Internet Computing*, Vol.2 No.1, Jan/Feb 1998, pp. 33-38.

[11] V. Schillings and Ch. Meinel, "Tele-TASK – Tele-teaching Anywhere Solution Kit", in *Proceedings of ACM SIGUCCS 2002*, Providence, USA, 2002, pp. 130-133.