# Tele-Lab IT-Security: an architecture for an online virtual IT security lab

Christian Willems
Hasso-Plattner-Institut für Softwaresystemtechnik
Prof.-Dr.-Helmert-Straße 2–3
Potsdam, Germany
christian.willems@hpi.uni-potsdam.de

## Abstract

*Recently, Awareness Creation in terms of IT security has become a big thing – not only for enterprises. Campaigns for pupils try to highlight the importance of IT security even in the user's early years. Common practices in security education – as seen in computer science courses at universities – mainly consist of literature and lecturing. In the best case, the teaching facility offers practical courses in a dedicated isolated computer lab. Additionally, there are some more or less interactive e-learning applications around.*

*Existing offers can do nothing more than impart theoretical knowledge or basic information. They all lack of possibilities to provide practical experience with security software or even hacker tools in a realistic environment. The only exceptions are the expensive and hard-to-maintain dedicated computer security labs. Those can only be provided by very few organisations.*

*Tele-Lab IT-Security was designed to offer hands-on experience exercises in IT security without the need of additional hardware or maintenance expenses. The existing implementation of Tele-Lab even provides access to the learning environment over the Internet – and thus can be used anytime and anywhere. The present paper describes the extended architecure on which the current version of the Tele-Lab server is built.*

## 1. Introduction

The increasing propagation of complex IT systems and rapid growth of the internet more and more attracts notice to the importance of IT security issues. The limits of technical security solutions are fixed by the lacking awareness of computer users, caused by laziness, inattentiveness and missing education. In the context of awareness creation IT security training becomes a topic of strong interest – as well as for companies as for indiviuals.

Traditional techniques of teaching (i.e. lectures or literature) have turned out to be not suitable for security training, because the trainee cannot apply the principles from the academic approach to a realistic environment within the class. In security training, gaining practical experience through exercises is indispensable for consolidating the knowledge [2].

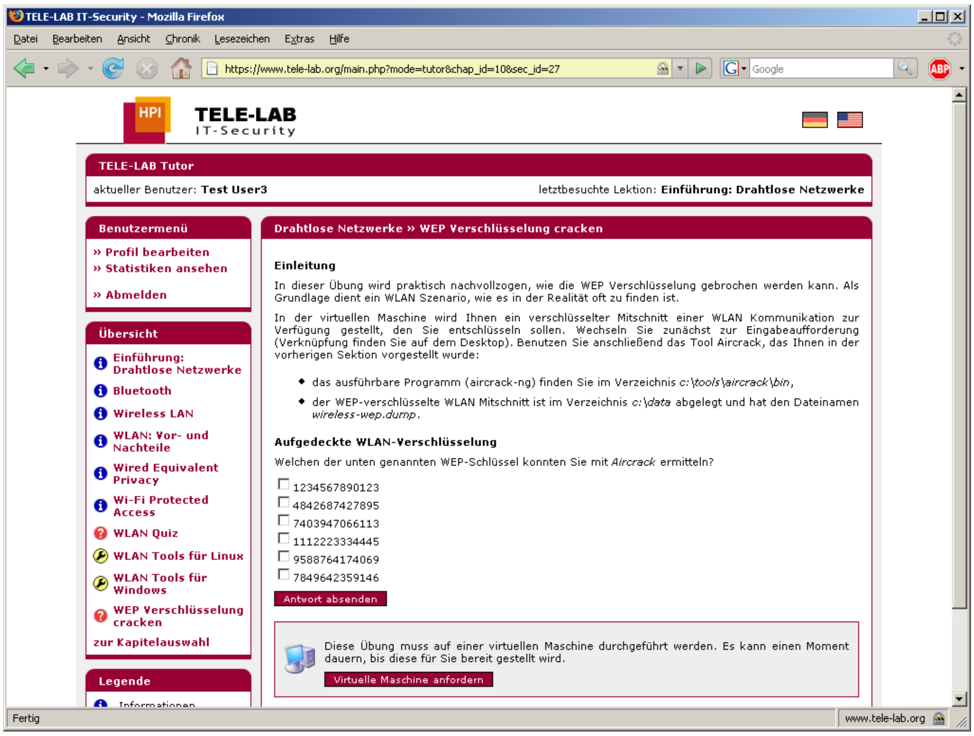
Precisely the allocation of an environment for these practical exercises pose a challenge for research and development. That is, because students need privileged access rights (root/administrator-account) on the training system to perform most of the imaginable security exercises. With these privileges, students can easily destroy a training system or even use it to attack other computers.

The classical approach is to provide a dedicated computer lab for security training. Such labs bare different backdraws: they are immobile, expensive to purchase and maintain and must be isolated from all networks.

Teleteaching approaches for security education are multimedia courseware or demonstration software, which do not offer practical exercises. In simulation systems users have kind of hands on experience, but a simulator doesn't behave like a realistic environment and simulation of complex systems is very difficult. Those approaches to security education will be presented more precise in section 2.

The Tele-Lab project (at first proposed in [10]) provides a novel e-learning system for practical security training in the WWW and inherits all characteristics from offline security labs. The Tele-Lab server basically consists of a web based tutoring system and a training environment built of virtual machines. The tutoring system offers three kinds of content: information chapters, introductions to security- and hacker-tools and finally definitions of practical exercises. Students perform those exercises on virtual machines (vm) on the server, which they use via remote desktop access. A virtual machine is a software system, that provides a runtime environment for operating systems. Such software-emulated computer systems allow easy deployment and recovery in case of failure.

Christian Willems:
"Tele-Lab IT-Security: an Architecture for an Online Virtual IT Security Lab"
in Proceedings of 2nd International Workshop on E-Learning and Virtual and Remote Laboratories 2008 (VIRTUAL-LAB),
Universitätsverlag Potsdam, Potsdam, Germany, ISBN 978-3-940793-17-1.



aktueller Benutzer: **Test User3** letztbesuchte Lektion: **Einführung: Drahtlose Netzwerke**

**Drahtlose Netzwerke » WEP Verschlüsselung cracken**

**Einleitung**

In dieser Übung wird praktisch nachvollzogen, wie die WEP Verschlüsselung gebrochen werden kann. Als Grundlage dient ein WLAN Szenario, wie es in der Realität oft zu finden ist.

In der virtuellen Maschine wird Ihnen ein verschlüsselter Mitschnitt einer WLAN Kommunikation zur Verfügung gestellt, den Sie entschlüsseln sollen. Wechseln Sie zunächst zur Eingabeaufforderung (Verknüpfung finden Sie auf dem Desktop). Benutzen Sie anschließend das Tool Aircrack, das Ihnen in der vorherigen Sektion vorgestellt wurde:

- das ausführbare Programm (aircrack-ng) finden Sie im Verzeichnis *c:\tools\aircrack\bin*,
- der WEP-verschlüsselte WLAN Mitschnitt ist im Verzeichnis *c:\data* abgelegt und hat den Dateinamen *wireless-wep.dump*.

**Aufgedeckte WLAN-Verschlüsselung**

Welchen der unten genannten WEP-Schlüssel konnten Sie mit *Aircrack* ermitteln?

☐ 1234567890123
☐ 4842687427895
☐ 7403947066113
☐ 1112223334445
☐ 9588764174069
☐ 7849642359146

Diese Übung muss auf einer virtuellen Maschine durchgeführt werden. Es kann einen Moment dauern, bis diese für Sie bereit gestellt wird.

**Figure 1. Screenshot: exercise on Wireless LAN in the Tele-Lab tutor**

A learning unit on e.g. "wireless networks" introduces to different WiFi technologies like Wireless LAN or Bluetooth, explains the functionality of mechanisms and protocols for wireless security and highlights weaknesses which lead to security problems. Thereafter, the tutoring system presents wireless tools for Windows and Linux like "Kismet" or the "Aircrack Suite". The chapter concludes with an exercise, where the student is asked to reveal a WEP encryption key from a wireless traffic dump file using "aircrack".

For that exercise, the student requests a virtual machine (here: Linux or Windows). If there is a free vm on the server, the student will be assigned to that vm and a remote desktop session will be started in an applet window. After performing the exercise (cracking the wireless dump), the student must enter the revealed WEP key in the tutor and prove, that he solved the exercise (see figure 1). The vm will be reclaimed and restored to its original state automatically.

Currently, there are also chapters on access control, authentication, encryption, intrusion detection, malicious software, man-in-the-middle-attacks, packet sniffing, port scanning and secure email. Additional chapters can be authored and integrated easily.

The architecture for providing this learning environment is described in section 3. The paper will be concluded with a summary of the results and an outlook on future work in section 4.

## 2 Related work

Related work in security education mainly includes Web-based training and multimedia courseware, demonstration software, simulation systems, and dedicated computer laboratories for security experiments. Recently, effords in compiling practical computer science courses using virtual machine technology have turned out some similarities to the Tele-Lab approach.

### 2.1 Web-based training and multimedia e-lecturing

Web-based training (WBT) means e-learning application that present learning units via WWW in the user's browser. Those learning units usually consist of text, images and custom made animations. Extended to multimedia courseware, WBT also offers *real* multimedia content, i.e. audio and video. WBT courses usually are digitized lectures and demonstrations. A modern e-lecturing system is e.g. *tele-TASK* [15]. It is a state-of-the-art streaming system to create online lectures and seminars for teaching IT security. WBT and e-lectures in nature support e-learning on the Web, but they offers no hands-on experiences to learners at all.

### 2.2 Demonstration software

Demonstration software programs such as *Cryptool* [7] and *CAP ("Cryptographic Analysis Programs")*

[16] are educational suits for learning about cryptography and cryptanalysis. They provide more interactivity for students to play with algorithms and therefore have more practical features than multimedia courseware. However, because demonstration software is mainly used to learn academic cryptographic algorithms, practical network security tools or everyday environments are seldom involved in learning of this kind.

### 2.3 Simulation systems

Simulation systems are normally used to train students in specific IT security subjects. In [14] and [20], the authors describe, how to familiarize students with intrusion detection by creating audit files with information on user activities. They ask students to detect and resolve an intrusion problem. *CyberCIEGE* [11] is a simulation game in which players construct computer networks and make decisions to protect valuable assets from attacks. Simulation systems offer students chances to perform operations for accomplishing "real" tasks such as identifying intrusion and recovering or cleaning systems. Nonetheless, those security operations are not really performed but simulated in an abstract environment. In fact, real computer systems can be modeled by such simulations only to a very limited degree. Simulation systems increase interactivity to some degree but still offer students no chance to apply real world tools and see what's going on in practice.

### 2.4 Dedicated computer laboratories

Dedicated computer laboratories for IT security have been created in many universities, e.g. described in [18] or [13]. Security experiments or exercises are usually arranged on the dedicated computer networks. Compared to other approaches, dedicated computer laboratories are ideal environments for practical security teaching because security exercises are performed by application of production software in real systems. However, practical education by laboratory measures normally results in high costs. Dedicated networks require expensive hardware/software investments and intensive efforts to create, configure, and maintain laboratory environments as well as to prepare, supervise, and evaluate exercises. On the other hand, most security exercises require system level access to the operating system. This introduces the risk of misuse and inconvenience of administration. For security reason, dedicated networks are normally operated on isolated networks, which implies that such security laboratories pitifully fail to benefit a wider range of learners outside campus.

### 2.5 Virtual machine based computer courses

Instead of dedicated laboratories, at some universities virtual machines have been established for courses with special requirements to hard- or software configuration. In [4], the author describes the utilization of UML[1]-based virtual machines for a course in operation systems maintenance, [3] proposes the use of VMware Workstation[2] for courses on network administration, security and database maintenance. The virtual machines are mainly used for providing an environment, where students are allowed to arbitrary configure an operation system and different applications. The virual machines must be compiled customly for every course and somehow distributed to the students. Both authors confirm the opinion of the Tele-Lab research group refering to the virtual machine technology as a cost efficient replacement for physical dedicated labs.
Existing work above indicates the value of a project like Tele-Lab, integrate practical security exercises into e-learning and degrade the costs of laboratory resources.

## 3 Architecture for the Tele-Lab server

The architecture presented in the paper at hand mainly consists of four components (see figure 2):

1. *Portal* and *tutoring environment* consist of dynamic webpages delivered by an Apache webserver. This is what was described as a WBT application above. Additionally, the tutoring environment provides user controls to require a virtual machine for exercising and initiates remote desktop sessions.

2. The *Virtual machine pool* is a collection of prepared virtual machines primarily managed by an arbitrary virtual machine monitor (see section 3.1 for details).

3. The *database* contains all user profile data, content data and persistent information on virtual machine states.

4. For proxying the remote desktop connections, the server is equipped with a free implementation of the *NX server* (see section 3.2).

5. The central *Tele-Lab control server* (also described in section 3.1) handles control of the virtual machines' state and the remote desktop connections.

[1] User-mode Linux is a Linux kernel for the user-space, allows the operation of virtual machines

[2] A host-based virtual machine monitor application, see `http://www.vmware.com/products/ws`

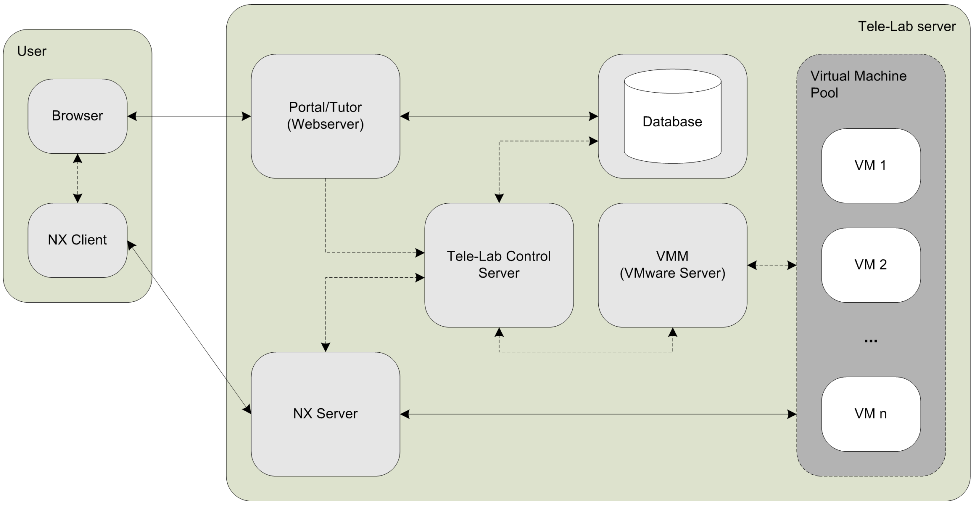


**Figure 2. Overview: architecture of the Tele-Lab server**

Before describing the important components in detail, the paper will motivate the objectives of the design process for this architecture. It was generally driven by the following requirements that arose from the analysis of previous versions of Tele-Lab such as *Tele-Lab on CD* [9] or the existing prototype of the Tele-Lab server [8].

*Handy system usage* showed up to be important because all former implementations of Tele-Lab was packed with the (partly deprecated) remainings from the very first prototypes user interface – where Tele-Lab was ment to be an offline/single-user system. Another significant backdraw of the existing UI was the implementation of the Remote Desktop Access as a VNC-applet *inside* the tutoring system. This resulted in a very small remote desktop and wired up situations like *"a browser an a desktop in a browser"*. Design and implementation of the dynamic webpages will not be within the scope of this paper.

Also important was the objective to gain *support for different operating systems inside the virtual machines*. The prototype of the server uses User-mode Linux [5] for the virtual machine pool for reasons of performance, security and availability of alternatives. UML can naturally only support Linux as guest-os inside its virtual machines. But Tele-Lab also wants to address people as users that are bound to Windows systems. A comparison of the available virtualization applications showed, that the VMware products and *Xen* [1] in combination with *Intel's VT* (hardware-aided virtualization [17]) can host both Linux and Windows having Linux as os on the physical host. As it is to remark that Windows guest support is still only experimental in Xen [6], the free *VMware Server*[3] is chosen for the virtual machine pool in this architecture.

Comparing the different virtualization solutions on the market also turned out that vendors and open source projects continously enhance their software suites: more features, more guest os support, better performance. So *being as independent as possible from a concrete virtual machine monitor* should be the next objective. Therefore this paper defines possible states of the virtual machines and proposes an interface with functions to trigger the appropriate transitions for vm control.

Of course, *security and reliability* are important issues for an e-learning system on IT security. Facing the facts: the Tele-Lab server provides remote administrator access to virtual machines on the Internet. This concept rises lots of – partly vertical – security objectives:

- protect the Tele-Lab server from attacks from the internet,
- protect the Tele-Lab server from attacks from malicious vm users,
- protect the internet world and own network against malicious vm users and
- protect vm users against other malicious vm users.

The implementation of the described architecture must prevent attacks from the internet on the web interface (server, scripting interpreter and scripts) and the remote desktop access service. Attacks from inside a virtual machine mainly object the virtual machine monitor: attacks on the hypervisor itself or the attempt to execute own code on the physical machine – bypassing the vmm. Different solutions for some concrete security issues will be described in sections 3.1 and 3.2. Please note that the protection against cross site scripting, code insertion or SQL injection through the user interface and security critical system updates have been considered to be an issue in implementation and administration, but will not be covered in detail here.

[3] see `http://www.vmware.com/products/server`

### 3.1 Virtual machine management and the generic vmm interface

The virtual machine pool consists of a number of virtual machines provided by the VMware server. Those virtual machines are run from different cloneable os images, each image configured as user vm or victim vm for the one or more exercises. The vm pool is maintained by the Tele-Lab Control Server, a service on the physical host.

**Virtual machine states** Within Tele-Lab, virtual machines can have a set of different states. Those trigger associated actions in the control monitor – which makes up the virtual machine management. The defined states are:

- *down* – the vm is not running, virtual machine image files exist, the vm is registered with the vmm and could be booted
- *ready* – the vm was started and is running, vm has networking up and a working ip
- *assigned* – the vm has been requested for an exercise and is assigned to a user
- *crashed* – the vm has crashed due to misuse or other guest system errors, transitions to *recovering*
- *recovering* – the vm is being restored to original state and rebooted

**Virtual machine control** The control server for virtual machine management has been implemented as a java application, that provides a webservice running in an *Apache Axis2* webservice engine. Currently, the services are only accessed by scripts running on the Tele-Lab's webserver. For security reasons, the Axis2 only accepts connection requests from localhost via the loopback interface. At the moment, the control server provides three webservices:

- *VM request* – when a user requires a vm using the HTML control in the tutoring interface, the webserver calls a service to get a vm *assigned*. If there is a free (*ready*) machine, the control server assigns that vm to the user and prepares the remote desktop connection (see section 3.2).
- *Change virtual machine pool* – two more webservices enable Tele-Lab administrators to *start* additional or *stop* unused machines from a web-based admin interface. This changes the state of virtual machines between *down* and *ready*.

More webservices may come, when clustering and communication between nodes will be implemented (see section 4).

The other purpose of the control server is to periodically check all virtual machines' states and eventually trigger actions:

- If a vm's state is *ready*, everything should be in place. In a longer interval, the control server checks, if the vm is still answering (heartbeat ping) and has networking up. If not, the vm is recovered.
- Should a vm be right *assigned* to a user, the control server checks, if the remote desktop connection is still alive. If not, the vm is recovered, the connection is being cleaned up.
- A *recovering* vm is being restored to its original state. In this implementation, the VMware Server make recovery an easy task. The virtual machines have been configured to have *non-persistent virtual hardisks*. This means, changes to a vm during a session are cleaned up whenever the machine stops. Recovery becomes "powering off" and rebooting a vm. A *recovering* machine becomes *ready*, when the boot sequence is finished – which is periodically checked by the control monitor.

**VMM interface** Speaking of starting, stopping or powering off virtual machines, sending heartbeat pings or checking vm's networking means using any of the virtual machine monitor's interfaces. For automatic control purposes, this can be either a command line interface or some API. All necessary calls to the vmm are encapsuled within a class, that implements the interface `VMControl`, which defines the methods `startVM()`, `stopVM()`, `recoverVM()`, `assignVM()`, `isAlive()` and `isConnected()`.
Currently, the interface is implemented in the `VMwareControl` class, which interacts with the CLI of the VMware Server. For example, a call to `startVM()` causes the `VMwareControl` object to execute `vmware-cmd /path/to/vm/config.vmx start`.
Like this, every method defined in the interface is mapped on database manipulation and CLI execution. Exceptions are `assignVM()` and `isConnected()`, which both also call functions for RDA connection control (see section 3.2).
When setting up the Tele-Lab server with a new virtualization software suite, a new class implementing the `VMControl` interface must be implemented.

### 3.2 Remote Desktop Access

Classic remote desktop connection environments are based on VNC (Virtual Network Computing), Windows RDP (Remote Desktop Protocol), X11 or terminal server solutions. The prototype of Tele-Lab server used VNC because of its adaptive bandwidth and the integrated Java-applet client. Though, the choice of VNC has different backdraws. First of all, there is no free available proxy solution for VNC, which causes the need for dynamic reconfiguration of the firewall in the existing prototype. Second is the weak authentication in the VNC protocol which causes possible unauthorized access or connection hijacking. Third is the rather poor performance of VNC connections. Remote desktops provided via VNC usually "feel" slow and sluggish.

The descriped architecture proposes NX[4] for the remote desktop connections. NX infrastructure and protocol offer features and performance that overcome the above described problems of VNC.

**NX features [12]** NX is a client/proxy remote desktop suite which implements a custom NX protocol and a set of very efficient compression algorithms. A NX remote desktop connection consumes about 40 kbit/s of bandwidth. The NX server is implemented as a proxy and can forward local and remote X sessions as well as remote RDP or VNC sessions. The NX protocol uses SSH for authentication and SSL for optional connection encryption.

The listed features bring everything needed for the Tele-Lab server's architecture. An RDA connection to a Linux machines tunnels the X server of a vm to the user, a remote desktop session on a Windows XP machine uses RDP via NX. The integration of SSH and SSL guarantees enhanced security.

Besides the official implementation of the NX server from NoMachine, there are two free variants of the server: the *freeNX*[5] project and the *2X terminalserver*[6]. The implementation of this architecture uses the 2X server.

**NX server integration** As already mentioned when describing the Tele-Lab control server in section 3.1, the remote desktop connection management is strongly connected to the virtual machine control. NX allows the configuration of remote desktop connections using `nxs` files (NX session). Those XML session files contain all necessary connection information as key-value pairs: server host, user credentials, connection parameters (screen size etc.), connection forwarding parameters and the SSH key.

NX session files are created by the vm control server, when a virtual machine are assigned to users. The files are stored into the document root of the Tele-Lab webserver. For security reasons, file names of the session files are scrambled via *MD5*. Both, filename and the session file itself are only transferred over HTTPS. When the session file has been established sucessfully, the assigned user must be enabled for NX. Therefore, the command line interface of the NX server is used. The control server executes `nxserver --userenable the_username`.

As this code example exposes, new users of the Tele-Lab server must also be registered to the NX server. This and other administration funtionality is also implemented in the Tele-Lab control server. Further details and features of the admin interface of the Tele-Lab server cannot be covered by this paper for reasons of extensiveness.

For assigned virtual machines, the control server checks, if the connection is still alive. Execution of `nxserver --list` returns a list of active sessions. The detection of an assigned vm with inactive connection triggers the immediate deletion of the corresponding generated NX session file.

**NX client integration** There is a reliable working beta-version of the NX client as Java-applet available from NoMachine. With the so called `NX WebCompanion` the remote desktop can be integrated into the web-based tutor environment seamlessly. As described above, connection data is stored in an NX session file on the webserver, when a vm is assigned. The tutor script opens a popup browser window with the NX client applet and the URL of the session file as applet parameter. The applet establishs the connection (performs SSH authentications, SSL handshake and negotiation of connection parameters) and starts the remote desktop session.

For security reasions, users are not allowed to copy and paste between the remote desktop and their local desktop.

When the NX connection is quit by the user or interrupted otherwise, or if the virtual machine crashes during a session, vm and NX connection are rolled back to the original states.

## 3.3 More implementation details

The realisation of the proposed architecture consists of many more components than the above described. Design and implementation of the virtual machine management including the encapsulation of the virtualization solution and the integration of the tutoring environment with the virtual laboratory using the NX technology are just pointed out as the major issues.

Other important implementation aspects have been:

- design and realisation of portal and tutor user interfaces
- database design and optimization
- implementation of administration interface and functionality
- host system setup and configuration

[4] Nomachine Inc. provides commercial NX server but free libraries and client, see `http://www.nomachine.com`

[5] see `http://freenx.berlios.de`

[6] see `http://www.2x.com/terminalserver`

- application and service configuration
- host networking configuration (physical and virtual network interfaces)
- complex host firewall configuration

Every single of the listed tasks poses little challenges described in detail by [19].

## 4 Conclusions and future work

The present paper describes an architecture for a comprehensive online e-learning solution for IT security. The proposal integrates a web-based tutoring application with a secure virtual laboratory environment. The virtual computer lab consists of various virtual machines. It is accessed via the internet using remote desktop connections. The architetcure has been implemented successfully, content from previous Tele-Lab revisions is already integrated for several chapters.

Future work will focus on different aspects: concerning technology, a second implementation of the vmm interface (e.g. `XenControl`) will be implemented. For better performance (providing a larger number of virtual machines simultanously) implementing clustering Tele-Lab servers could be a needful enhancement for the architecture. Other possible technology upgrades focus on collaborative learning: community tools, messenging and chat or even remote desktop assistance are considered to be integrated.

The second part of future work is the generation of content: implementation of learning units and exercise scenarios, compilation of courses for different learning levels and the identification of future topics.

Third and last aspect for the future is the evaluation of the system as e-learning tool: is learning (are awareness creation campaings) with Tele-Lab IT-Security more efficient than with classical teaching methods? Currently, different high schools over germany test and evaluate a Tele-Lab course with lectures for scholars of different age.

## References

[1] P. Barham, B. Dragovic, K. Fraser, S. Hand, T. Harris, A. Ho, R. Neugebauer, I. Pratt, and A. Warfield. Xen and the art of virtualization. In *SOSP 03: Proceedings of the 19th ACM symposium on Operation system principles*, pages 164–177, New York, USA, 2003. ACM Press.

[2] M. Bishop. Education in information security. *IEEE Concurrency*, 8(4):4–8, 2000.

[3] W. I. Bullers, S. Burd, and A. F. Seazzu. Virtual machines – an idea whose time has returned: application to network, security, and database courses. In *SIGCSE '06: Proceedings of the 37th SIGCSE technical symposium on Computer science education*, pages 102–106, New York, NY, USA, 2006. ACM.

[4] R. Davoli. Teaching operating systems administration with user mode linux. In *ITiCSE '04: Proceedings of the 9th annual SIGCSE conference on Innovation and technology in computer science education*, pages 112–116, New York, NY, USA, 2004. ACM.

[5] J. Dike. User-mode linux. In *Proceedings of the 5th Annual Linux Showcase & Conference*, Oakland, California, USA, 2001.

[6] Y. Dong, S. Li, A. Mallick, J. Nakajima, K. Tian, X. Xu, F. Yang, and W. Yu. Extending Xen with Intel Virtualization Technology. *Intel Technology Journal*, 10:193–204, 2006.

[7] B. Esslinger. Cryptool – spielerischer Einstieg in klassische und moderne Kryptographie: neue Version – fundierte Awareness in Deutsch und Englisch. *Datenschutz und Datensicherheit*, 26(10), 2002.

[8] J. Hu, D. Cordel, and C. Meinel. A Virtual Machine Architecture for Creating IT-Security Laboratories. Technical report, Hasso-Plattner-Insitut, 2006.

[9] J. Hu and C. Meinel. Tele-Lab IT-Security on CD: Portable, reliable and safe IT security training. *Computers & Security*, 23:282–289, 2004.

[10] J. Hu, M. Schmitt, C. Willems, and C. Meinel. A tutoring system for IT-Security. In *Proceedings of the 3rd World Conference in Information Security Education*, pages 51–60, Monterey, USA, 2003.

[11] C. E. Irvine and M. F. Thompson. Expressing an information security policy within a security simulation game. In *Proceedings of the Sixth Workshop on Education in Computer Security*, pages 43–49, Monterey, USA, 2004.

[12] G. Pinzari. Introduction to NX Technology. Technical report, Nomachine Inc., 2003.

[13] D. Ragsdale, S. Lathorp, and R. Dodge. Enhancing information warfare education through the use of virtual and isolated networks. *Journal of Information Warfare*, 2:53–65, 2003.

[14] N. Rowe and S. Schiavo. An intelligent tutor for intrusion detection on computer systems. *Computers and Education*, 31:395–404, 1998.

[15] V. Schillings and C. Meinel. Tele-TASK – teleteaching anywhere solution kit. In *Proceedings of ACM SIGUCCS*, Providence, USA, 2002.

[16] R. Spillman. CAP: A software tool for teaching classical cryptology. In *Proceedings of the 6th National Colloquium on Information System Security Education*, Redmond, Washington, USA, 2002.

[17] R. Uhlig, G. Neiger, D. Rodgers, A. L. Santoni, F. C. Martins, A. V. Anderson, S. M. Bennett, A. Kagi, F. H. Leung, and L. Smith. Intel Virtualization Technology. *IEEE Computer*, 5:48–56, 2005.

[18] G. Vigna. Teaching hands-on network security: Testbeds and live exercises. *Journal of Information Warfare*, 2:8–24, 2003.

[19] C. Willems. Eine erweiterte Architektur für den Tele-Lab Server: Implementierung eines generischen Interfaces für Virtual Machine Monitors. Master's thesis, Universität Trier, 2006.

[20] C. Woo, J. Choi, and M. Evens. Web-based ITS for training system managers on the computer intrusion. In *Proceedings of the 6th International conference on Intelligent Tutoring Systems*, Biarritz, France and San Sebastian, Spain, 2002.