Hasso–Plattner–Institut für Softwaresystemtechnik
an der Universität Potsdam

Technischer Bericht

# A Virtual Machine Architecture for Creating IT–Security Laboratories

von
Ji Hu, Dirk Cordel und Christoph Meinel

## Abstract

E-learning is a flexible and personalized alternative to traditional education. Nonetheless, existing e-learning systems for IT security education have difficulties in delivering hands-on experience because of the lack of proximity. Laboratory environments and practical exercises are indispensable instruction tools to IT security education, but security education in conventional computer laboratories poses the problem of immobility as well as high creation and maintenance costs. Hence, there is a need to effectively transform security laboratories and practical exercises into e-learning forms.

This report introduces the Tele-Lab IT-Security architecture that allows students not only to learn IT security principles, but also to gain hands-on security experience by exercises in an online laboratory environment. In this architecture, virtual machines are used to provide safe user work environments instead of real computers. Thus, traditional laboratory environments can be cloned onto the Internet by software, which increases accessibilities to laboratory resources and greatly reduces investment and maintenance costs.

Under the Tele-Lab IT-Security framework, a set of technical solutions is also proposed to provide effective functionalities, reliability, security, and performance. The virtual machines with appropriate resource allocation, software installation, and system configurations are used to build lightweight security laboratories on a hosting computer. Reliability and availability of laboratory platforms are covered by the virtual machine management framework. This management framework provides necessary monitoring and administration services to detect and recover critical failures of virtual machines at run time. Considering the risk that virtual machines can be misused for compromising production networks, we present security management solutions to prevent misuse of laboratory resources by security isolation at the system and network levels.

This work is an attempt to bridge the gap between e-learning/tele-teaching and practical IT security education. It is not to substitute conventional teaching in laboratories but to add practical features to e-learning. This report demonstrates the possibility to implement hands-on security laboratories on the Internet reliably, securely, and economically.

# Contents

# List of Figures

# List of Tables

# 1. Introduction

With increasing emergence of security threats and system vulnerabilities, IT security has more and more impact on our daily work and life. In order to strengthen public awareness of IT security, many universities have integrated security courses into their curricula. Those courses are taught by traditional measures such as textbooks, slides, or papers. In most cases, however, only theoretical aspects of IT security are covered and one essential part of education, i.e. hands-on experience, has been neglected. In very few cases, students have a chance to learn practical skills or to experiment in production environments. In fact, one of the most important purposes of security education is to prepare the skillful workforce in response to the future security challenges [Bishop, 2000]. Practical IT security education must meet the need of training students to apply security technologies for real environments. In this connection, traditional teaching measures have been proved to be insufficient. There is a need to provide hands-on security experiences by exercises in laboratory environments.

## 1.1. Challenges in IT security education

Nowadays, e-learning has been applied in security education and become complementary part to traditional classroom teaching. This can be demonstrated by many tele-lectures, tutoring systems, and demonstration programs which have been used for everyday teaching and learning [Schillings and Meinel, 2002], [Rowe and Schiavo, 1998], [Woo et al., 2002], [Esslinger, 2002], [Spillman, 2002]. Nevertheless, e-learning has difficulties in delivering hands-on security experience. This is particularly because e-learning has failed to deal with laboratory environments and real-life exercises. From a practical point of view, teaching security in conventional computer laboratories is sufficient to help students to gain hands-on experience. But, pitifully, such laboratories often lead to high costs and geographical limits, which degrades its accessibility for the people outside campus. In response to the problems above, we came up with an idea to integrate laboratories environment and practical exercises to e-learning systems. If this idea works it would result in a very helpful solution to offer individuals easy accesses to laboratory resources via the Internet.

## 1.2. The concept of Tele-Lab IT-Security

*Tele-Lab IT-Security* is a novel concept for teaching various subjects of information and network security. Its original objective is to develop a web-based instruction tool which allows students to learn about IT security and also gain experiences by exercises. In order to provide real-life experience, these exercises are planned to run on a real OS. As shown in Figure 1.1, Tele-Lab has a web interface. The tutor is a web server which presents teaching materials from a knowledge repository to students and provides them navigation in learning and exercises.

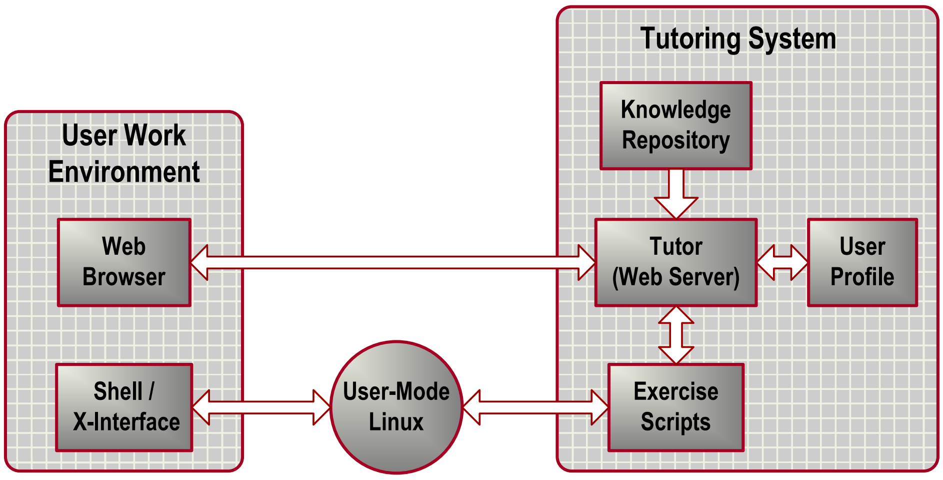


Figure 1.1.: The concept of Tele-Lab IT-Security .

User work environment is supported by a native Linux which integrates necessary security tools which are needed by the exercises. Therefore each Linux machine can be used as a "virtual" laboratory where a student is able to exercise practical security. Those exercises are designed as scripts so that the tutor can prepare tasks by calling them. After students finish their tasks with security tools, their submission will be evaluated also by scripts. In this way, we can realize a real-life learning scenario.

To support teaching security in laboratories, we have developed a standalone Tele-Lab system [Hu et al., 2003] which uses a native Linux as a user work environment. In order to ease administration workload, the entire Tele-Lab system including both the Linux system and teaching materials is integrated into a *Knoppix* live-CD [Hu and Meinel, 2004]. It is a special bootable CD on which we can run a complete Linux OS without the presence of a hard disk. By this means, a portable and reliable training CD has been realized. It can be easily used on any common PC. If any failures or errors take place, the computer can be restarted without damaging hardware or software systems.

The motivation to develop the virtual machine architecture for Tele-Lab is to transfer the native Tele-Lab system into a web-based server from which laboratory resources are available for remote users via the Internet. In this architecture *virtual machines* [Goldberg, 1974] instead of physical PCs are applied for providing separated work environments. Virtual machines (VM) are software for simulating a physical computer. Since they are normally user-level applications and can be connected to a network, we can run a number of VMs on a host and clone an entire laboratory network on a host. On a VM we can give a privilege right to its user. Any crash of the VM caused by the user would not affect the Tele-Lab host and can be easily recovered. Thus laboratory resources can be created and accessed conveniently at a low cost.

## 1.3. Requirements for online security laboratories

Tele-Lab IT-Security provides a virtual machine architecture for online security laboratories. Before it becomes a pragmatic solution, following requirements need to be specially taken into account:

- **Functionality**. In order to effectively clone functions which are normally provided by real laboratories, VMs must be carefully configured with suitable VM software. A user who works on a VM should not notice obvious differences in function from his/her experience on a real machine.
- **Reliability**. Users would feel frustrated if services or VMs are frequently interrupted by failures. Hence, prompt detection of VM errors and fast recovery from failure are necessary to guarantee service availability.
- **Security**. With privileges allowed on VMs and Internet connections, a VM might be converted by its user to an attack workstation and endanger production networks. User activities on the VM must be under control and misuses of laboratory resources should be prevented by security measures.
- **Performance**. In order to run VMs as many as possible, the Tele-Lab server must provide enough system resources. On the other hand, performance is crucial for individual VMs. The Tele-Lab server has to manage simultaneous access and each VM has to possess reasonable resources for its user.

Therefore, besides the architecture, this report also focuses on a set of solutions to meet its functionality, reliability, security, and performance requirements.

## 1.4. Structure

The rest of this report is organized as follows: Section 2 briefly introduces related work in security education. Section 3 presents an overview of the Tele-Lab IT-Security architecture. Then Section 4 explains the design and implementation of the Tele-Lab architecture and also addresses the construction, management, and security issues in detail. Section 5 describes the performance benchmarks of the virtual machines of Tele-Lab. Section 6 introduces the application and evaluation of the Tele-Lab architecture and presents some case studies. Finally Section 7 concludes the report and figures out future work.

# 2. Related work

Related work in security education mainly includes multimedia courseware, demonstration software, simulation systems, and dedicated computer laboratories for security experiments.

## 2.1. Multimedia courseware

Multimedia courseware normally consists of digitized lectures and demonstrations, which features text, images, animation, videos, etc, e.g. tele-Task [Schillings and Meinel, 2002] is a state-of-the-art streaming system to create online lectures and seminars for teaching IT security. Courseware in nature supports e-learning on the Web, but it pitifully offers no hands-on experiences to learners.

## 2.2. Demonstration software

Demonstration software programs such as Cryptool [Esslinger, 2002] and CAP ("Cryptographic Analysis Program") [Spillman, 2002] 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. E.g. [Rowe and Schiavo, 1998] and [Woo et al., 2002] familiarize students with intrusion detection by creating audit files with information on user activities and asking students to detect and resolve an intrusion problem. CyberCIEGE [Irvine and Thompson, 2004] 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. Examples include [Vigna, 2003], [Ragsdale et al., 2003], [Hoffman et al., 2003], [Lindskog et al., 1999]. 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[Vigna, 2003]. 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.

Existing work above indicates that we need a new approach to integrate practical security exercises into e-learning and degrade the costs of laboratory resources.

# 3. Architecture

Figure 3.1 shows an overview of the Tele-Lab architecture. Its main components include: **Tele-Lab portal,** which is the front-end of Tele-Lab to provide a web interface for accessing laboratories via networks; **Virtual lab**, which hosts a number of configured virtual machines (VMs) to be assigned as dedicated laboratory environments; and related **VM management** functions for the assignment, monitoring, and maintenance of the VMs.

## 3.1. The Tele-Lab portal

The portal provides entrances for both learners and administrators. If a learner logs on it will present a VNC applet embedded in the browser, by which the learner can access a VM. The applet is a client of the Virtual Network Computing [Richardson et al., 1998] which is a remote display system to allow a user to view a desktop environment from the Internet. In Tele-Lab, VNC is used to provide an interface to VMs. For an administrator, the portal presents a management front-end for monitoring and managing Tele-Lab and its VMs.

## 3.2. Virtual lab

The virtual lab runs a number of VMs and connects them to a virtual network. In order to prevent them from exhausting the system resources of the host, we have chosen User-Mode Linux (UML) to create lightweight VMs. One advantage of UML is that it can create a small virtual Linux on top of a real Linux operating system. These VMs are carefully configured so that the host can support more VMs and yet maintain a reasonable performance.

Each VM implements a self-contained learning environment that has been shown in Figure 1.1. It consists of an IT security tutoring system and a user work environment. The tutoring

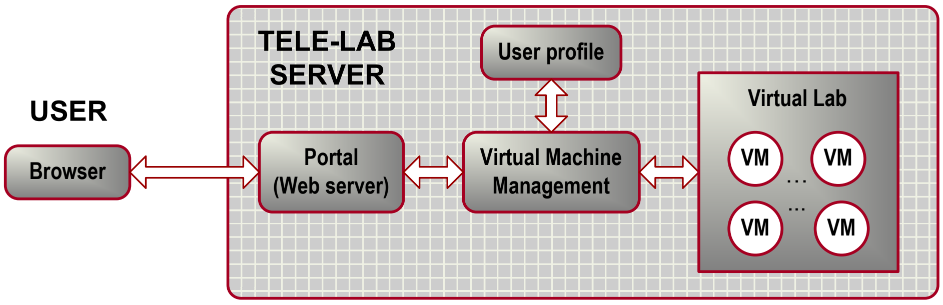


Figure 3.1.: An overview of the Tele-Lab architecture.

Table 3.1.: A chapter on Password-based Authentication

| | |
|---|---|
| Concepts section | Password hashing (DES and MD5)<br>The “passwd” file in Linux<br>Password selection criteria |
| Tools section | The *passwd* command in Linux<br>The *John-the-Ripper* Password Cracker |
| Exercise section | Cracking random-generated passwords with John-the-Ripper |

system is a local web server to present teaching contents and organize security exercises. The user work environment is supported by a UML Linux system. It allows privileged access for finishing exercises.

### 3.2.1. Tutoring system

The security tutoring system consists of a knowledge repository, a tutor, and some exercise scripts.

- **Knowledge Repository** is a content database which has three types of teaching sections: theoretical concepts, tool tutorials, and exercises. *Concept Sections* introduce declarative knowledge of a subject e.g. public-key encryption. They are presented in text, graphics or videos. *Tool Sections* present the tutorials on some security tools. They are closely related to the subject and involve screenshots or animations. In *Exercise Sections* students can perform security tasks. They are designed as scenarios and implemented by scripts. To finish a task, the students must apply some of the tools introduced in the tool sections on the VM and take step-by-step interaction with the tutoring system. Generally a security subject contains sections of those three types. Table 3.1 is an example chapter for a security subject.

- **Tutor** is responsible for content presentation, learning navigation, and exercise management. The tutor organizes the sections in the repository to form a teaching chapter. It also guides a user where to start a subject and where to continue at the end of each learning step. Meanwhile the tutor records user's learning results and shows some statistics about the user. The exercise management is done by calling scripts (e.g. Perl or PHP scripts) to prepare tasks and evaluate results.

- **User profile** is imported to the tutoring system when the user logs in from the portal. It stores a user's personal data and performance data. Personal data include some items like names, accounts and user types. Users can register themselves as one of three types: “general user", “administrator" and “student". For each user type, a different set of the subjects is defined. E.g. “administrator" allows a user to learn about more advanced subjects than general users. Performance data is about user's learning performance, e.g. information about the sections worked through and the time spent on them. Those data are used to create a summary about a user's learning.

### 3.2.2. User work environment

The user work environment is provided by UML. It consists of a virtual Linux OS kernel and a basic file system (a Debian image). Basic system programs, graphical user interfaces, and open-source security tools are installed on it. User configurations such as the accounts and software settings are also predefined. Thus a user has a convenient operating environment and a lot of trivial steps are avoided. The local user interface on a VM is straight and simple: To access the tutoring system, a Mozilla browser is used and for performing exercises security programs or tools can be applied via a shell or an X-window interface. When necessary a user is able to switch to a privileged mode and perform system-level operations.

## 3.3. Virtual machine management

The goal of the VM management is to ensure Tele-Lab to continuously run in a reliable way. Its main management functions include VM administration, VM monitoring, and user monitoring. Section 5 will give more details on management functions.

## 3.4. Security management

Security is an important factor which we must deal with in every aspect of the Tele-Lab architecture. Generally, each user is allowed a privilege right on his/her virtual machine. This situation introduces a serious security problem: users might convert their virtual machine into an attack station and corrupt the Tele-Lab system or compromise production networks. The mission of security management of Tele-Lab is to implement effective security isolation and to prevent misuse of virtual machines.

# 4. Creating virtual machines

VMs should be real, lightweight platforms with reasonable performances. This section describes installation, resource allocation and benchmarks of VMs.

## 4.1. Installation

VM installation includes the work to install and configure a virtual OS, a user work environment, and the tutoring system.

### 4.1.1. Virtual kernel

The virtual kernel is implemented by User-Mode Linux (UML) which can be treated as any other ordinary application on the host and can be easily customized. The Tele-Lab host is a Debian Linux server. The VM created by UML consists of a virtual kernel (version 2.4.26-3um) and at least one virtual disk partition. A VM has a similar boot process to a native Linux except that it starts from a user terminal.

### 4.1.2. Virtual disks

They provide file systems for installing a base system of Linux and other software programs. To run UML, we need at least a root filesystem. File systems are simulated by using large sized files on the host and will be specified when booting a VM. The generated filesystem is empty. We then created a Debian installation on the filesystem so that the UML kernel is able to run a basic Debian Linux with necessary programs (including drivers, shell, essential system and graphical programs). The Debian installation for VMs is special compared with a normal installation because it must match a virtual kernel instead of a real kernel.

### 4.1.3. User work environment

It requires necessary software and appropriate configuration. System software includes shell programs such as editors and administration tools. For web applications, a simple graphical user interface (X-window and the *XFCE* window manager) and a browser (*Mozilla-firefox*) are installed. Other programs are installed in terms of the needs of security exercises. E.g. *GunPG*[1] program for PGP encryption, *John-the-Ripper* for Password-based Authentication, and the *Exim* mail server, *OpenSSL*, and *Mozilla-thunderbird* for Secure Email. Configurations

[1]The GNU Privacy Guard (GunPG) `http://www.gnupg.org/`.

are important to user interfaces. When a VM is assigned, user configurations are automatically applied. E.g. a default account, "*bob*", is used for a user to do most exercises, but if privileged operations are needed, the user can obtain a root account by clicking an icon. As to email configurations, account settings of the local mail server and clients are also prepared in advance. Thus users can avoid irrelevant system setups before they can start exercises.

### 4.1.4. Tutoring server

It is a local web server on a VM, which consists of HTML or PHP pages and CGI programs such as Perl scripts, shell scripts, and binary programs. Web tools such as *Apache* and the *PHP4* and *Perl* interpreters are installed on each VM. Web contents stay on the host. They are shared by all VMs and can be conveniently synchronized.

## 4.2. Resource allocation

Resource allocation of VMs is to isolate resource consumption and performance among them. By allocating processor, memory and disk resources to VMs, we can limit the resource usage of individual VMs and improve overall performances.

### 4.2.1. Processor resource allocation

Isolation of processor resources among VMs is necessary to stop denial-of-service attacks caused by malicious processes in a VM. Processor resource allocation and isolation are supported by the following ways or their combinations:

- Limiting the number of the virtual processors assigned to a VM: i.e. if a VM is given two processors, it has no more than two processes running on the host no matter how many processes it is trying to generate.
- Degrading the priority of VM processes by adjusting their *nice*[2] value: then VM processes get less processor time than other host processes.
- Using high-end hardware for the host machine: fast processors and big memory can effectively relieve performance impacts from VMs.

### 4.2.2. Virtual memory allocation

The size of "physical" memory of a VM is specified before it is booted. The memory is not a real physical space of the host but a virtual memory space emulated by a temporary file. The virtual memory is then mapped into a physical address space and swapped out to the disk when it does not need to access physical space. Thus, we can provide VMs with a big virtual memory space instead of the real size of physical memory. The virtual memory is mounted as a dynamic RAM based filesystem supported by the temporary filesystem (TMPFS). TMPFS

[2]"*nice*" is a Linux command to execute a command with lower priority, i.e. be "nice" to other users.

[Snyder, 1990] can effectively speed up accessing of virtual memory files. E.g. we suppose to run 30 VMs on a host, each of which needs 64 MB “physical” memory. Intuitively we need at least 1.8 GB (30 $\times$ 64 MB) RAM space. With virtual memory (1.8 GB swapped in a temporary directory), only about 809 MB real RAM was indeed used for running those VMs. It is just 42% of the total "physical memory" space for 30 VMs.

### 4.2.3. Virtual disk resource allocation

Each VM needs an installed Linux filesystem on a virtual disk. To avoid assigning each VM a big disk image, we created an original image as a “backing file” and save live changes to a small Copy-On-Write (COW) file. Then, all VMs can read data from the same backing file while storing individual updates into their COW file. Since COW files can be created, copied or deleted as a normal file, VMs can be started, killed, or recovered on demand like any other application. Tele-Lab uses a 300 MB backing file which contains a typical Linux installation. In fact, copy-on-write (COW) files are used by VMs. This saves a large number of disk space. E.g. under general use of a VM, COW files are very small (20 B - 200 KB). To start 30 VMs on the host, about 310 MB real disk space (30 COW files plus the backing file) is indeed used, which is merely 3,4% of the total disk space (30 $\times$ 300 MB) all VM can provide.

# 5. Virtual machine management

VM management has an important role in Tele-Lab: it is responsible for assignment and maintenance of VMs and therefore guarantees them work in a reliable manner. User-Mode Linux has an essential control tool, i.e. *mconsole*. It allows a client to access UML kernel from the host to configure, stop, reboot, and backup VMs. Mconsole also has an event notification interface, through which programs inside a VM can send messages to the host. Based on mconsole, it is possible to implement advanced management functions for Tele-Lab.

## 5.1. Virtual machine management framework

The management frame of Tele-Lab consists of five main components: a VM assignment table and the function modules to administrate and monitor VMs or users (see Figure 5.1). The VM assignment table is a data structure which records actual states of each VM. The VM administration module is responsible for starting, stopping, or recovering VMs in specific circumstances. The VM monitoring module detects and reports critical errors of the VMs. The user monitoring module watches user activities on the VM and detects abnormal events. The user notification module informs a user about events on his/her VM.

## 5.2. The VM assignment table

Information in the VM assignment table indicates which VMs have been assigned and to whom, which are free, and which are found failed and need recovery. Active VMs have an

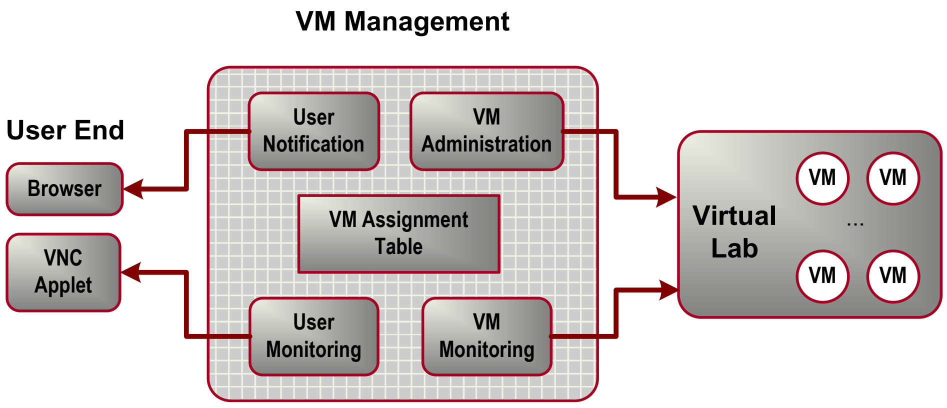
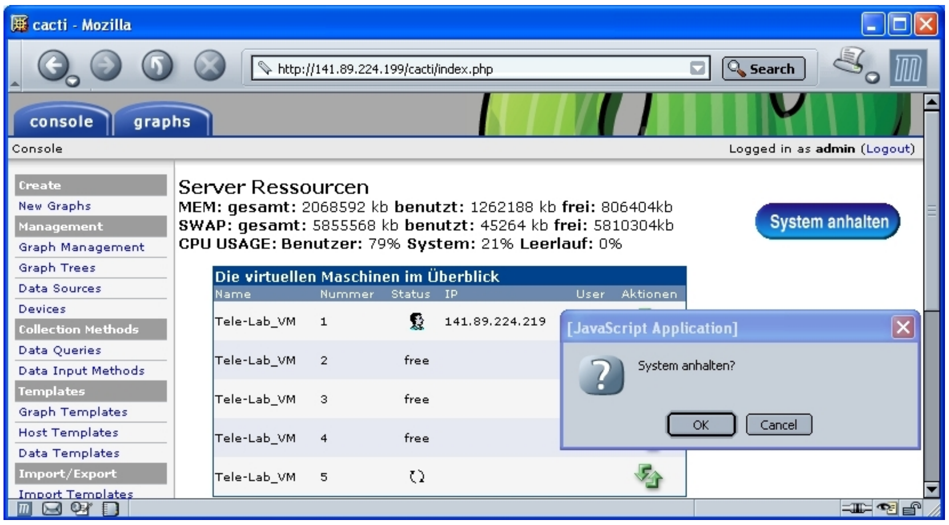


Figure 5.1.: VM management framework.



Server Ressourcen
**MEM: gesamt:** 2068592 kb **benutzt:** 1262188 kb **frei:** 806404kb
**SWAP: gesamt:** 5855568 kb **benutzt:** 45264 kb **frei:** 5810304kb
**CPU USAGE: Benutzer:** 79% **System:** 21% **Leerlauf:** 0%

Die virtuellen Maschinen im Überblick

| Name | Nummer | Status | IP | User | Aktionen |
|---|---|---|---|---|---|
| Tele-Lab_VM | 1 | | 141.89.224.219 | | |
| Tele-Lab_VM | 2 | free | | | |
| Tele-Lab_VM | 3 | free | | | |
| Tele-Lab_VM | 4 | free | | | |
| Tele-Lab_VM | 5 | | | | |

System anhalten?

Figure 5.2.: Web administration console.

entry in the table. Each entry includes fields like VM name and number, current mode, source IP, and current user. The VM name and number are used to identify a VM. Source IP is referred to the address of the user machine. The user name specifies the user who is using the VM. The mode indicates the current state of the VM. A VM runs in one of three possible modes: "*free*", "*assigned*", and "*recovered*". A VM is "free" when it is started and has not yet assigned to any one. A VM is "assigned" to a user when he or she logs in. Then it becomes an exclusive workstation for that user. The "recovered" mode is a transitional state and triggered in two cases: (1) if a VM fails it will be marked as "recovered" and restarted in the background; (2) if a user logs out, his/her VM will be immediately reclaimed and recovered. After recovery, it will change to "free" and a new VM is ready.

## 5.3. VM administration

The VM administration module provides control functions to assign, reclaim, and recover VMs. When a user logs in, the administration module searches the assignment table for a free VM and assigns one to this user if possible. Then the mode of the VM changes to "assigned" and the corresponding entry in the assignment table is updated. When a user logs out, his/her VM is reclaimed and returns to mode "free" after a recovery procedure. A VM is recovered to default settings by killing all processes related to the VM and restarting it with a default COW copy. Recovery is triggered when a user logs out or exceptional events happen (e.g. VM failures).

We also developed a friendly interface for the administrator to control VMs and monitor their system status. This interface includes an administration console and a system status monitor.

- Administration Console provides controls to start/stop the Tele-Lab system, create/recover VMs, and shows the updates of the VM assignment table (see Figure 5.2).

- System Status Monitor provides system status information of the host and each VM. Details about their processor, memory, disks, and network performance are presented by graphs. It periodically collects status data from the host and VMs. Then those status data are processed and performance-related statistics are generated. Finally a set of graphs is created from the performance data.

## 5.4. VM monitoring

This module collects the state of each VM and reports errors to the administration module. There are two approaches to implement monitoring. The first approach is installing a monitoring agent on each VM, which can report precise and detailed state information of the VM in real time. But its problem is that it can be interrupted or disabled by users. Alternatively, we can periodically poll VMs, scan their services, and report errors on the services. Normally, the services to be scanned are necessary. Any failure of them will result in difficulty in working on the VM. E.g. the VNC, Web, and an email server are a prerequisite for learning and exercises. Any failure of them indicates that the VM is defective and needs recovery. This approach is useful to detect critical errors in general situations though some errors might be missed. Tele-Lab implemented monitoring by services scanning because of its simplicity and reliability. Even if any critical errors are missed, the user who notices the problems can request a new VM by clicking a button on his/her web page.

## 5.5. User monitoring

The user monitoring module is used to reduce unnecessary occupation of system resources. It monitors user activities on VMs and measures user's idle time by tracking a user's keyboard and mouse inputs on a VM. Instead of monitoring system event logs which could be altered by users, we developed a software hook at the VNC client and directly monitored input events. We modified part of the original VNC client program and added a detection code to record the time of the user's latest keyboard/mouse events and measure his/her idle time on a the VM. If the idle time exceeds a threshold (e.g. 20 mins), it indicates the user won't continue his/her learning and his/her VM will be reclaimed. Thus the freed resources can be used for other users.

## 5.6. User notification

This function is necessary to inform a user of special events on his/her VM in real time. E.g. if any critical failure is detected and before the VM is recovered, its user should be notified of this event and then he/she can continue exercises on a VM. In order to capture messages, a small frame is embedded in the browser, which refreshes itself each 15 seconds to get message updates on VM events.

# 6. Security

Security is an important consideration to the Tele-Lab architecture. Security requirements of Tele-Lab are special compared with other online learning or tutoring systems. In many security exercises, users are allowed a privilege right on VMs. This introduces a security risk that a user might convert his/her VM to an attack station and corrupt the Tele-Lab system or compromise production networks. Therefore, we had to find a way to balance functionality and security of Tele-Lab. We came up with the idea of security isolation, which allows necessary accessibility to VMs while constraining risks in a safe scope.

## 6.1. Security requirements

In general, corruptions of VMs will not cause serious problems because they can be handled by the VM management. Our concern is about the risk of misusing VMs. Because misuses often take place around the boundary between a VM and the outside, we had to make sure those isolations at the boundaries on the system and virtual network levels.

1. *System level isolation.* The host operating system should be protected from any intrusion from VM processes. Strict access control must be enforced between a VM and the host to isolate their process space from each other.
2. *Network level isolation.* Users should only be able to establish connections which are required for performing exercises. We need effective access control between a VM and its outside nodes such as other VMs and production systems to prevent network attacks.

## 6.2. System level isolation

The basic idea to secure User-Mode VM processes is “root jail” [Dike, 2001], i.e. the privilege right of a VM can be safely assigned to its user by running the VM as a user-level application on the host. Thus, the root user is jailed in the VM and privileged access from a VM to the host is disallowed. Successful root jailing requisites that system calls from the VM on the host must be safe. VMs are emulated by the port of system calls and thus the most possible way to break out of a VM to the host is to exploit system calls. Therefore, we must assure that any kind of VM processes has no ability to execute arbitrary system calls to the host.

User-Mode Linux VM processes by default run in the tracing thread mode (the “*tt*” mode). This mode is problematic because the VM kernel shares the same address space with its user processes and the kernel space is writable. By accessing kernel data, a process could possibly break out to the host by forging system calls. Security isolation requires that the VM kernel memory is protected against modification by user space and any critical information of the kernel must stay inside the kernel. In this way, user processes of a VM have no possibility to forge system calls that they want and to escape from the jail.

System security isolation can be implemented with the *Separate Kernel Address Space* mode (the "*skas*" mode), which is a secure mode to run User-Mode Linux. It requires to apply a patch on the host kernel. This patch implements a separate address space scheme: the VM kernel runs in an entirely different host address space from its user processes. VM kernel binary and data are totally invisible to its processes and to anyone logged in to it. This makes VM kernel data secure from tampering by its processes. Therefore, "root jail" with the support of the "skas" mode can effectively protect the host and implement system security isolation.

## 6.3. Network level isolation

The level of security isolation constrains all user actions in the scope of a VM by applying controls on all connections between the VM and the outside. The access control has the following policies:

- *Local connections and local network services are allowed on a VM.*
- *A VM is not allowed to initialize any network connection to the other VMs on the host.*
- *A VM is allowed to accept or respond to the connections for the VNC service if this VM has been assigned.*
- *Except for those connections mentioned in Policy 3, a VM is not allowed to launch any forms of connections for the Internet.*

Our idea for access control is enforcing the policies above on a so-called "*iptables*" firewall. Iptables sets up a firewall to perform operations on IP packets. Iptables has three kinds of tables to define control rule sets. One table contains a number of chains. Each chain defines a set of rules. Those rules are applied on those packets which traverse the chain. The *nat* table and the *filter* table are the most important tables for defining access control rules.

1. The "*filter*" table is specifically designed to filter packets.
2. The "*nat*" table is designed to perform network address translation for packets, portforwarding, etc.

### 6.3.1. IP-address reuse

Each VM is a node of a private network (a virtual LAN in the host). By default no VMs are accessible from the outside. If any VM is assigned, its VNC service will be accessible over the Internet via the host network interface. This requires multiple VMs to share a public address. Such an address reuse scheme was implemented by port forwarding, which assigns each VNC server on the VMs a unique port number. When a VNC packet comes to the public network interface, iptables checks the packet header and forwards it to the right VM depending on its heading port number. With address reuse, access control can be implemented by applying rules on port forwarding.

E.g. IP address authentication can be done by allowing only a validated user to be able to connect to her/his own VM. E.g. each VM on the virtual LAN is assigned a private address, which is calculated from the number of the VM, "**192.168.0.99 + number**". For instance, "VM1" has "192.168.0.100", "VM2" has "192.168.0.101", and so forth. The VNC server on each VM is locally listening on two ports: "5800" and "5900" (see the 2nd column of Table 6.1). The port forwarding rules translate those internal access points to public access points. Then, those VNC services are mapped to different ports ("**5799 + number**" and "**5899 + number**") on the host network interface ("141.199.55.40"). The translation table is shown in Table 6.1.

Table 6.1.: The translation table of the VNC access points.

| **VM** | **Internal Access Point** | **Public Access Point** |
|---|---|---|
| VM1 | 192.168.0.100 : (5800, 5900) | ⇒**141.199.55.40** : (580**0**, 590**0**) |
| VM2 | 192.168.0.101 : (5800, 5900) | ⇒**141.199.55.40** : (580**1**, 590**1**) |
| ... | ... | ... |
| VM30 | 192.168.0.129 : (5800, 5900) | ⇒**141.199.55.40** : (58**29**, 59**29**) |

Those translation rules in the nat table are dynamically managed depending on the assignment situation of the VMs. At the beginning, iptables disables any port forwarding for VMs. If a VM is assigned, specific rules are inserted to the nat chains. When the user logs out, those rules for his/her VM are immediately abandoned. For example, when a user from "163.158.1.2" logs in to the Tele-Lab host which publicly runs on the "141.199.55.40", a VM must be assigned to this user. Here, we suppose VM3 on "192.168.0.102" is chosen by the VM management system. Then two rules will be inserted into the PREROUTING chain of the nat table:

```
iptables -t nat -A PREROUTING -s 163.158.1.2 -p tcp dport 5802 -j DNAT to 192.168.0.103:5800
```

and

```
iptables -t nat -A PREROUTING -s 163.158.1.2 -p tcp dport 5902 -j DNAT to 192.168.0.103:5900
```

Those rules map the VNC service of VM3 to the public access point: "141.199.55.40 : (5803 and 5903)". The "***-s***", "***-p***" and "*–**dport***" options are used for access control: only VNC connections from the authenticated user (on "163.158.1.2") are allowed to be forwarded to that VM. Finally, this public access point will be passed to the VNC client at the user end, so it knows where to connect the VNC server of the VM. Table 6.2 shows the enforcement of the port forwarding rules on the packets in this example.

Table 6.2.: Enforcement of port forwarding rules.

| VNC Packet | Original | Altered |
|---|---|---|
| Client request | Source: 163.158.1.2/7890* <br> Dest: 141.199.55.40/5802 | Source: 163.158.1.2/7890 <br> Dest: **192.168.0.102/5800** |
| Server reply | Source: 192.168.0.102/5800 <br> Dest: 163.158.1.2/7890 | Source: **141.199.55.40/5802** <br> Dest: 163.158.1.2/7890 |
| Client request | Source: 163.158.1.2/8335* <br> Dest: 141.199.55.40/5902 | Source: 163.158.1.2/8335 <br> Dest: **192.168.0.102/5900** |
| Server reply | Source: 192.168.0.102/5900 <br> Dest: 163.158.1.2/8335 | Source: **141.199.55.40/5902** <br> Dest: 163.158.1.2/8335 |

(* 7890 or 8335 is the local port number of the VNC client.)

### 6.3.2. Packet filtering for access control

The virtual network is different from any conventional LAN. It is created by a virtual switch device. As shown in Figure 6.1, "eth0" is the external interface of the host and "tap0" is the internal interface created by the virtual switch. This virtual network has a very special feature: each VM is attached to a central node (the virtual switch on the host), and any traffic of VMs must be relayed by the host. Therefore, we can apply an iptables firewall to control traffic. In order to satisfy control policies, we have two categories of filtering rules for iptables. All filtering rules refer to internal addresses because the NAT chains are processed before the route decision in the packet filtering chains. That is, source or destination addresses seen by the packet filter are local addresses.

*Packet Filtering between VMs and External Networks*: the following rules can be applied based on the network structure in Figure 6.1.

**Rule 1**: external hosts are only allowed to access the VNC service on the virtual network (on port 5800 and 5900).

```
iptables -A FORWARD -i eth0 -o tap0 -p tcp –dport 5800,5900 -j ACCEPT
```

**Rule 2**: only VNC related packets are allowed to go out of the virtual network to the outside.

```
iptables -A FORWARD -i tap0 -o eth0 -m state –state ESTABLISHED, RELATED -j ACCEPT
```

**Rule 3**: any packet from the Tele-Lab host onto the virtual network is allowed (for the VM management).

```
iptables -A FORWARD -s 141.199.55.40 -o tap0 -j ACCEPT
```

**Rule 4**: only those packets related to the packets in Rule 3 are allowed to reach the Tele-Lab host from the virtual network.

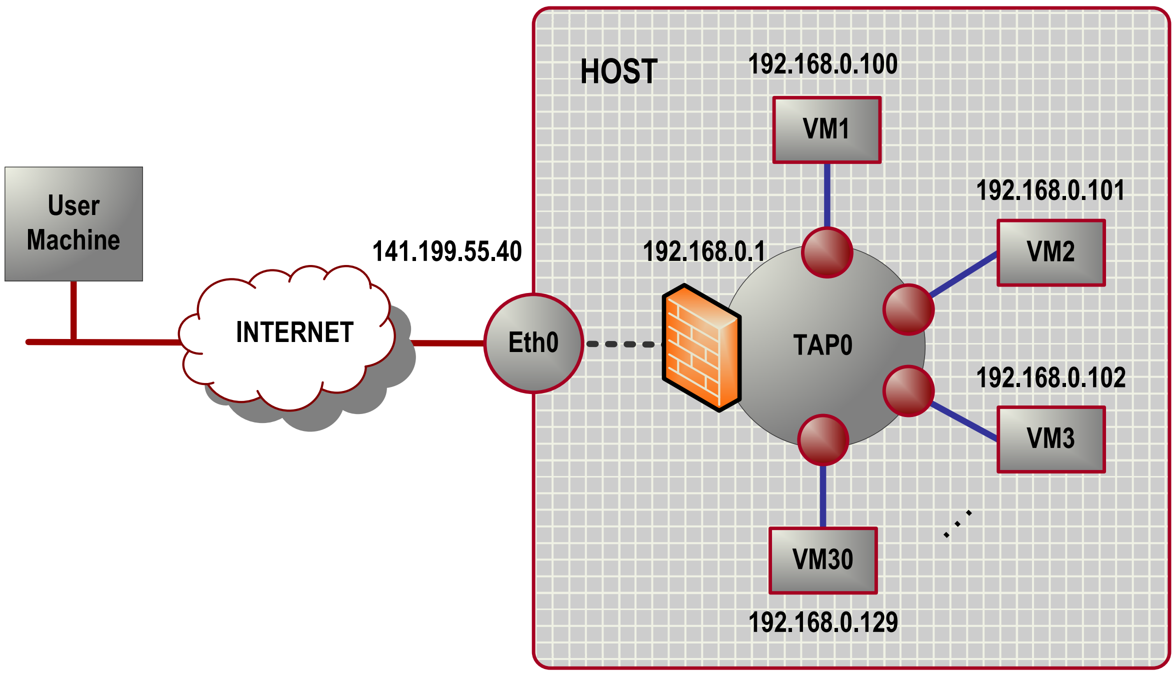


Figure 6.1.: Structure of the virtual network.

```
iptables -A FORWARD -i tap0 -d 141.199.55.40 -m state –state ESTABLISHED, RELATED -j ACCEPT
```

*Packet Filtering between VMs*: Iptables is able to filter any packets between VMs because the Tele-Lab host is a central node of the virtual network. In order to isolate internal connections between VMs, we can apply a simple rule on the INPUT chain.

**Rule 5**: a packet will be dropped if both the source and destination addresses of this packet fall in the same address range of the virtual network.

```
iptables -A INPUT -i tap0 -o tap0 -j DROP
```

# 7. Performance

Performance is an important matter to the usability of VMs. The difference between VMs and physical machine should be as small as possible. The following benchmarks were done to determine the effectiveness of the replacement of real machines with VMs.

## 7.1. Performance benchmarks

*Lmbench* [McVoy and Staelin, 1996] was used for measuring VM performances. Lmbench is an open-source software suite for operating system microbenchmarks. Because performance issues are usually caused by latency problems, bandwidth problems, or their combinations, lmbench runs a set of small microbenchmarks to measure system latency and bandwidth of data movement among the processor and memory, network, file system, and disk.

Machines to be benchmarked include the host, VMs, and other computers for comparison. Their system specifications are summarized in Table 7.1. Tele-Lab server ("P IV 2800") is the host. Its Linux kernel was specially patched to enable the User-Mode Linux "SKAS" mode, which improves VM kernel performance and security. We ran thirty VMs on the host. The 64 MB "physical" memory is simulated by the virtual memory of the host. The virtual disk partition is provided by a COW file which only stores differences from a shared root disk image. "P II 350" is a reference machine, i.e. a native PC with decent installation. Specifications and benchmark data of two native systems ("P Pro 167" and "P Pro 133") came from the lmbench database [McVoy and Staelin, 1996]. Lmbench version 3.0-a3 was applied in the benchmark. It was compiled and installed on each machine. We first started all VMs on the host, and then ran lmbench on the host and on VMs respectively. Depending on hardware conditions, it took lmbench about from 30 minutes to one hour to finish the benchmark on each machine. It then produced a long list of detailed benchmark results.

## 7.2. Benchmark results

Among benchmark output, we were particularly interested in those related to the performance of processes, memory, and filesystems, such as process creation time, memory read/write bandwidth, and file system latency.

Figure 7.1 shows the time used to create processes on each platform. We can find that the process creation time of the VM is very close to the machine of "P Pro 167". I.e. the process performance of VMs is nearly at the level of that of an Intel Pentium Pro 167 computer. This performance looks not so satisfying compared with those of modern computers because the UML Linux has to spend extra time to make system calls for processor simulation. However, UML Linux is resource-friendly, e.g. one host can effectively run more VMs at a relatively low

Table 7.1.: System specification of the performance benchmark.

| Sys. | Hardware | Kernel | Filesys. |
|---|---|---|---|
| P IV 2800 | P IV 2.8 GHz (hyper threading)<br>2 GB RAM<br>50 GB IDE HD | Linux 2.4.26 / i686 - smp | EXT3 |
| VM | Virtual kernel<br>64 MB V-RAM<br>300 MB V-Disk | UML 2.4.26 | EXT2 |
| P II 350 | P II 350 MHz<br>64 MB RAM<br>10 GB Disk | Linux 2.4.18 | EXT3 |
| P Pro 167 | P Pro 167 MHz | Linux1.3 / i686 | EXT2 |
| P Pro 133 | P Pro 133 MHz | SunOS 5.5.1 | UFS |

resource expense. Although part of process performance is sacrificed, we gained efficiency in hand.

Memory read and write rates are shown in Figure 7.2. We can find the memory performance of the VM is extremely impressive. Except for the host, it performs much better than other native machines. The memory bandwidths of VM are about 6.6 times higher than those of "P II 350" and nearly equivalent to those of the host. This is because UML Linux is able to utilize most resources of the host and so inherits a good performance.

Figure 7.3 shows the result of file system latency. Lmbench created 1,000 zero-sized small files in the current working directory and then removed those files. The time of the file creation and removal was measured respectively. Similar to the memory bandwidth, the filesystem performance of the VMs is also very satisfying and its file I/O is one time faster than that of the native computer of "P II 350".

The benchmark results indicate that a high-performance host helps to improve performance of the VMs. With reasonable configurations, a VM can be seen as a machine between a Pentium II 350 computer and a Pentium Pro 167 computer. Intuitive experience also shows that VMs can smoothly run most general applications from common system operations to web browsing. Therefore, VMs have been proved an effective and efficient computing-resource replacement of physical machines for Tele-Lab IT-Security.

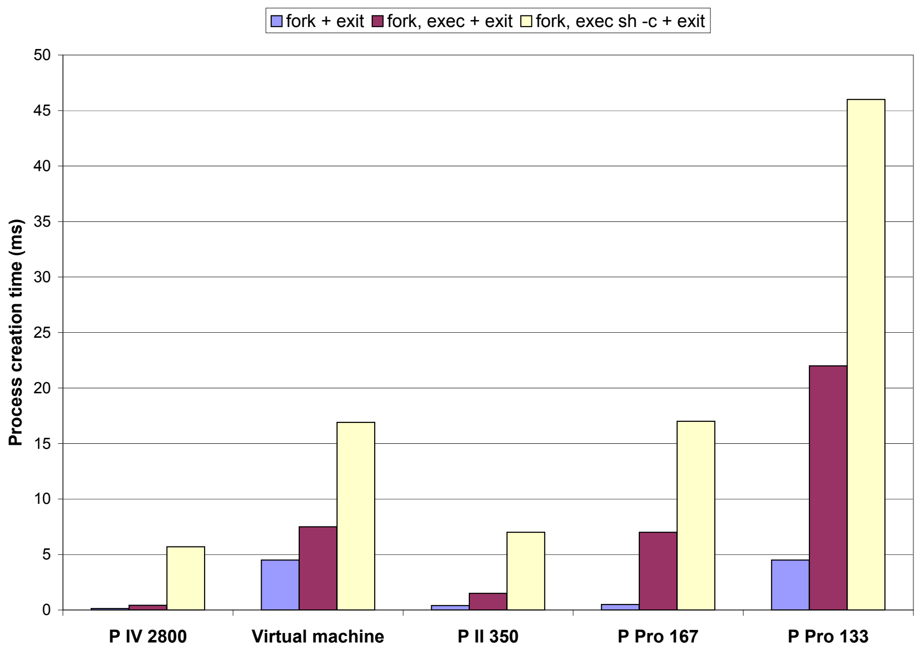


Figure 7.1.: Process creation time of the performance benchmark.

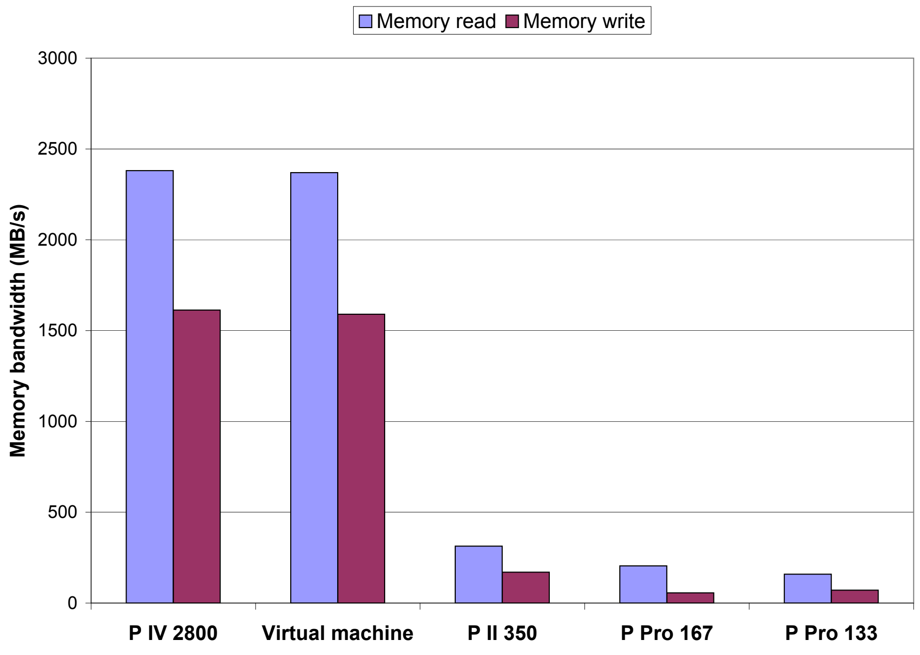


Figure 7.2.: Memory bandwidth of the performance benchmark.

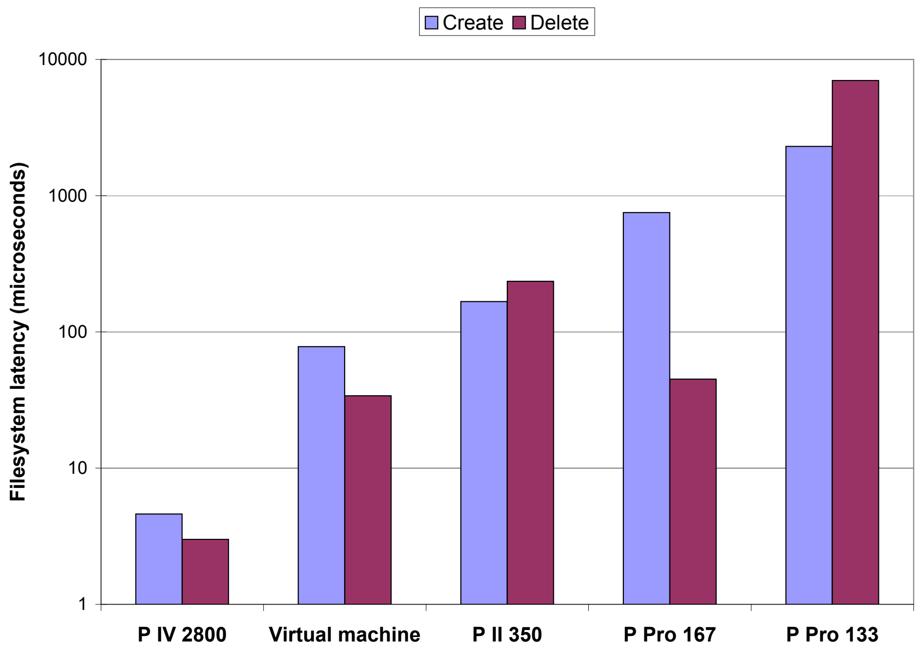


Figure 7.3.: Filesystem latency of the performance benchmark.

# 8. Applications

Tele-Lab IT-Security can be used to teach various subjects in cryptography and network security. Like other online training or tutoring systems, Tele-Lab IT-Security presents theoretical facts about subjects as well as relevant tutorials and demos. The distinction of Tele-Lab from the other systems is that it offers students chances to perform security experiments in a lightweight and real-life laboratory environment. Up to the winter semester 2005/2006, with the help of a student group at the Hasso-Plattner-Institute, we created eleven security chapters featuring practical exercises for Tele-Lab. Those chapters cover a broad range of subjects such as secret-/public-key encryption, secure email, authentication, port scanning, access control, intrusion detection, man-in-the-middle, firewalls, wireless security, etc. Practical features of those chapters are listed in Table 8.1.

With a modularized structure for organizing contents, Tele-Lab can be adapted for specific teaching purposes or user groups. In general, Tele-Lab can be used in the following ways:

- *Supporting security courses.* Tele-Lab supplies laboratory exercises for a specific course. Contents are designed or re-organized according to a course plan. In this circumstance, the Tele-Lab server can be deployed in a laboratory or accessible from campus networks.
- *Online-learning.* Tele-Lab is operated as an e-learning service. It is a complete tutoring system: both theoretical basics and practical exercises are integrated. With the virtual machine architecture, Tele-Lab is a distinct tele-teaching tool for delivering hands-on experience over the Internet.
- *Industrial training.* Tele-Lab is customized for security training in industry. Contents are developed for specific training topics or products which are interesting to companies or theirs customers.

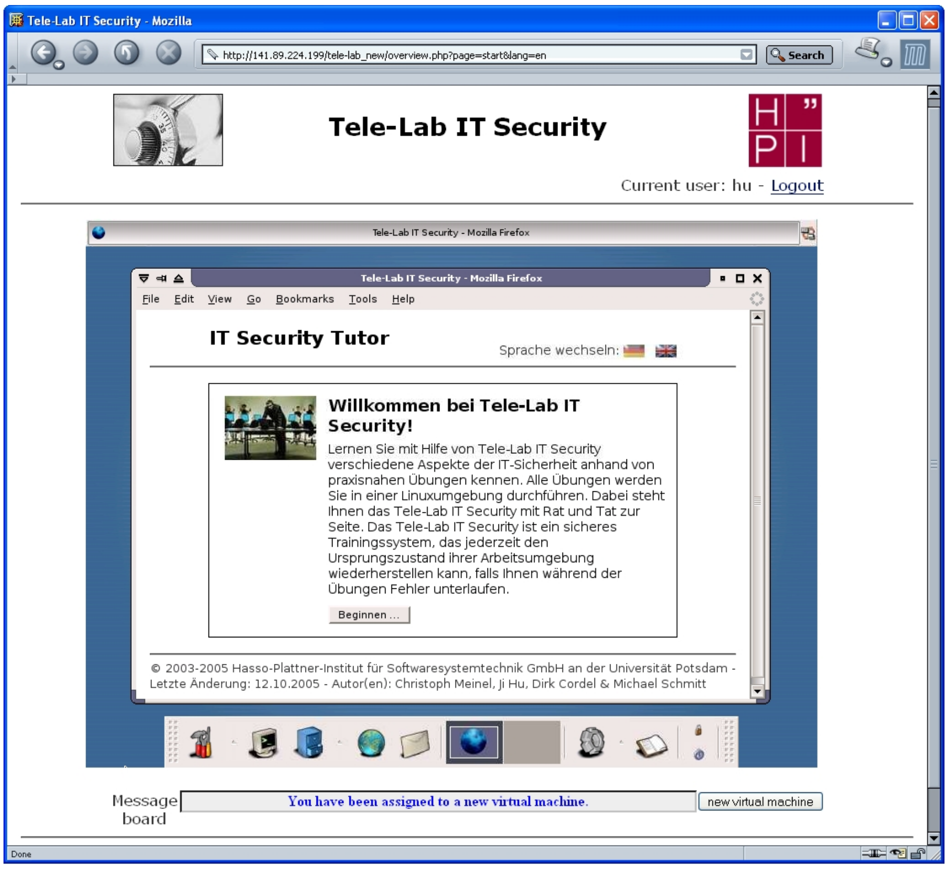
## 8.1. Learning processes

Before a user starts learning, he/she must register an account. On the registration page, the user inputs personal data as well as selects language settings ("English" or "German") and user categories ("admin", "general user", or "IT student"). The data then is processed to create a user profile in a database. The security tutor is by default started when the virtual machine is assigned to the user. Once the user starts the security training, the profile of this user will be imported from the host and applied in the virtual machine (see Figure 8.1). Then the tutor presents a list of available chapters from which the user can choose one. A Tele-Lab chapter is arranged like this: theoretical facts of a chapter are introduced first; afterwards, tutorials of related security tools are presented; finally practical exercises are prepared and assigned to the user.

Normally, an exercise is performed in three steps:

Table 8.1.: Topics being developed in Tele-Lab IT-Security.

| **Topic** | **Practical Features** |
|---|---|
| Symmetric Encryption | Learn about cryptography and symmetric encryption. Exercise message encryption/decryption using GnuPG. |
| Public-Key Encryption | Exercise how to use GunPG and OpenSSL to create key pairs and certificates. Exercise encrypting and signing with both tools. |
| Secure Email | Learn about email security standards, SMIME and OpenPGP. Exercise signing /encrypting messages via the Mozilla Thunderbird client using SMIME and Enigmail (PGP) tools. |
| Password-based Authentication | Learn about password security and exercise decoding passwords with the John-the-Ripple cracker. |
| Access Control | Demonstrate how access control mechanisms in Linux are breached by Buffer Overflow. |
| Port Scanning | Exercise how to find services on the target host with Nmap and close unnecessary services. |
| Firewalls | Exercise configuring an *iptable* packet filter and setting up a firewall in Linux. |
| Intrusion Detection | Exercise setting up the Snort IDS program. Detecting attacks from the Snort log files. |
| Packet Sniffing | Demonstrate how plain FTP sessions can be eaves-dropped. Exercise detecting ARP spoofing attacks with the Arpwatch tool. |
| Man-in-the-Middle | Demonstrate how man-in-the-middle attacks compromise SSL sessions. |
| Wireless security | Learn about Wireless related sniffers and crackers. |


Tele-Lab IT Security - Mozilla
http://141.89.224.199/tele-lab_new/overview.php?page=start&lang=en
Search
Tele-Lab IT Security - Mozilla Firefox
File Edit View Go Bookmarks Tools Help
Sprache wechseln:
Beginnen ...
new virtual machine
Done

Tele-Lab IT Security

Current user: hu - Logout

IT Security Tutor

Willkommen bei Tele-Lab IT Security!

Lernen Sie mit Hilfe von Tele-Lab IT Security verschiedene Aspekte der IT-Sicherheit anhand von praxisnahen Übungen kennen. Alle Übungen werden Sie in einer Linuxumgebung durchführen. Dabei steht Ihnen das Tele-Lab IT Security mit Rat und Tat zur Seite. Das Tele-Lab IT Security ist ein sicheres Trainingssystem, das jederzeit den Ursprungszustand ihrer Arbeitsumgebung wiederherstellen kann, falls Ihnen während der Übungen Fehler unterlaufen.

© 2003-2005 Hasso-Plattner-Institut für Softwaresystemtechnik GmbH an der Universität Potsdam - Letzte Änderung: 12.10.2005 - Autor(en): Christoph Meinel, Ji Hu, Dirk Cordel & Michael Schmitt

Message board: You have been assigned to a new virtual machine.

Figure 8.1.: The start page of the IT security tutor.

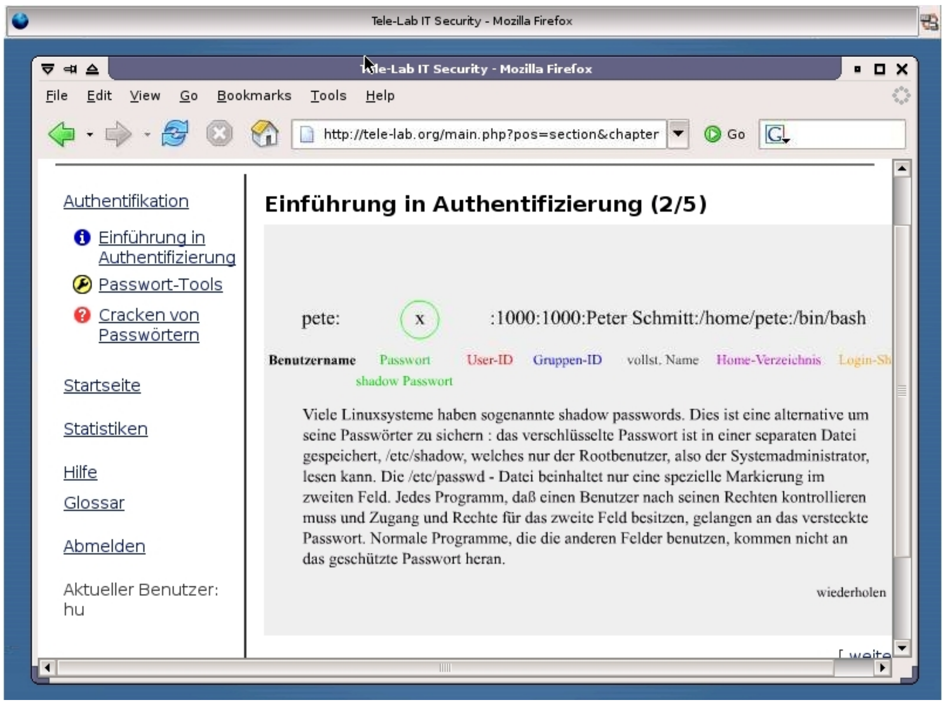
- *Exercise preparation.* To configure necessary system settings, the tutor will invoke particular Perl or shell scripts on the virtual machine. E.g. for the "Secure Email" exercise, a virtual role and related mail settings are created so that email can be exchanged in an effective circumstance.
- *Task generation.* The tutor activates scripts to generate data or materials (e.g. files or messages), which are needed in the exercise. Then questions are shown on the web page. If possible, those questions are dynamically generated with different details each time. E.g. in the "Password Cracking" exercise, the "*passwd*" file to be decrypted is generated at run time.
- *Result evaluation.* The user completes tasks on the virtual machine. After submiting results, the tutor evaluates them by scripts. To simulate a real scenario, evaluation process is often done in an interactive way. For instance, to finish the Secure Email exercise, the user has to interact with the tutor and exchange signed or encrypted messages. In each step, the tutor checks messages and tells user what to do in the next step. In case of fail, the user can repeat the exercise until a correct solution is found.

## 8.2. Case study: password-based authentication

We demonstrate a concrete learning process with an example chapter, Password-Based Authentication. It is about how users are authenticated through passwords and how passwords are protected in the Linux systems. In its exercise, a Linux *passwd* file is generated. The user needs a privilege right to crack it with the*"John-the-Ripper"* cracker.

- In the introduction sections, concepts of password-based authentication are introduced first (see Figure 8.2). Those concepts include password hashing (DES and MD5), the UNIX "passwd" or "shadow" files, and safe password criteria.
- Information about relevant security tools is presented next. Those tools include the *passwd* command[1], the PAM[2], and the John-the-Ripper password cracker. Some of them are introduced by Flash animation clips (see Figure 8.3).
- Tasks such as creating password hashes and cracking passwords are assigned to the user. Here we only demonstrate the completion of the password cracking task. The first step is work environment preparation: password hashing configuration is initialized on the virtual machine and DES is chosen as the default hashing function .
- The next step is to assign the user a task to crack passwords with John-the-Ripper (see Figure 8.4).
- Each time, a group of random passwords is generated and saved in a "passwd" file (see Figure 8.5).
- The user downloads the "passwd" file into a local directory of the virtual machine (see Figure 8.6).

[1]A Linux command which creates passwords by DES hashing.
[2]PAM is "*Pluggable Authentication Modules*" to configure password hashing.



Einführung in Authentifizierung (2/5)

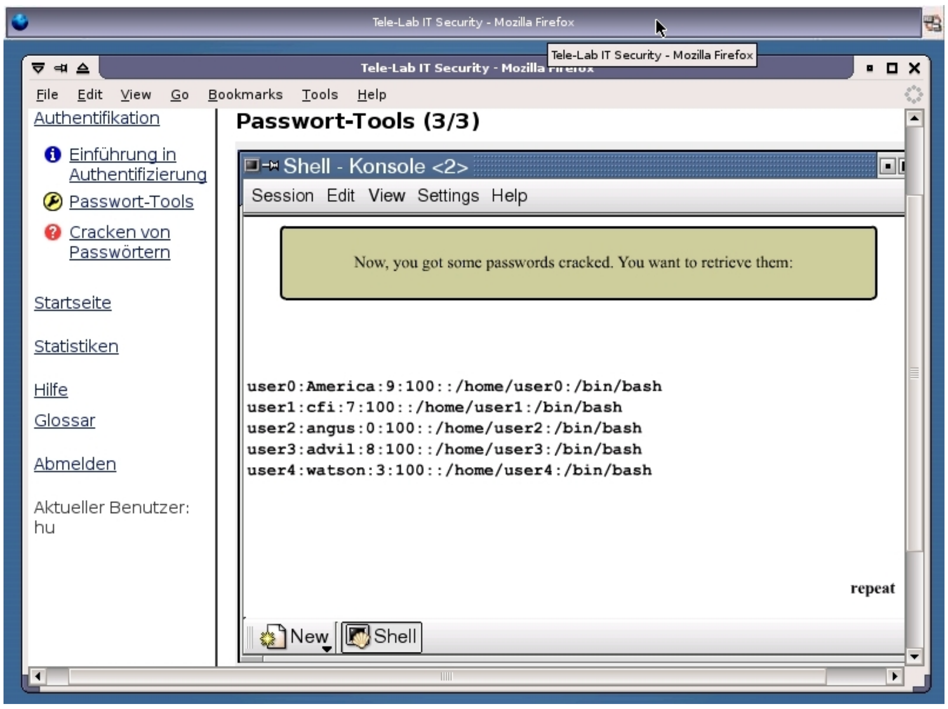
pete: x :1000:1000:Peter Schmitt:/home/pete:/bin/bash

Viele Linuxsysteme haben sogenannte shadow passwords. Dies ist eine alternative um seine Passwörter zu sichern : das verschlüsselte Passwort ist in einer separaten Datei gespeichert, /etc/shadow, welches nur der Rootbenutzer, also der Systemadministrator, lesen kann. Die /etc/passwd - Datei beinhaltet nur eine spezielle Markierung im zweiten Feld. Jedes Programm, daß einen Benutzer nach seinen Rechten kontrollieren muss und Zugang und Rechte für das zweite Feld besitzen, gelangen an das versteckte Passwort. Normale Programme, die die anderen Felder benutzen, kommen nicht an das geschützte Passwort heran.

Figure 8.2.: Introduction to password-based authentication.

- Evaluating results. In order to apply John-the-Ripper, the user must have a privilege right. Tele-Lab provides a "Root Terminal" button on the virtual machine. From that, the user can switch to a root mode and execute John-the-Ripper to crack passwords (shown in Figure 8.7).
- After passwords are found and answers are submitted, the tutor examines the submission (see Figure 8.8). Then the evaluation result is shown to the user and recorded in the user profile.

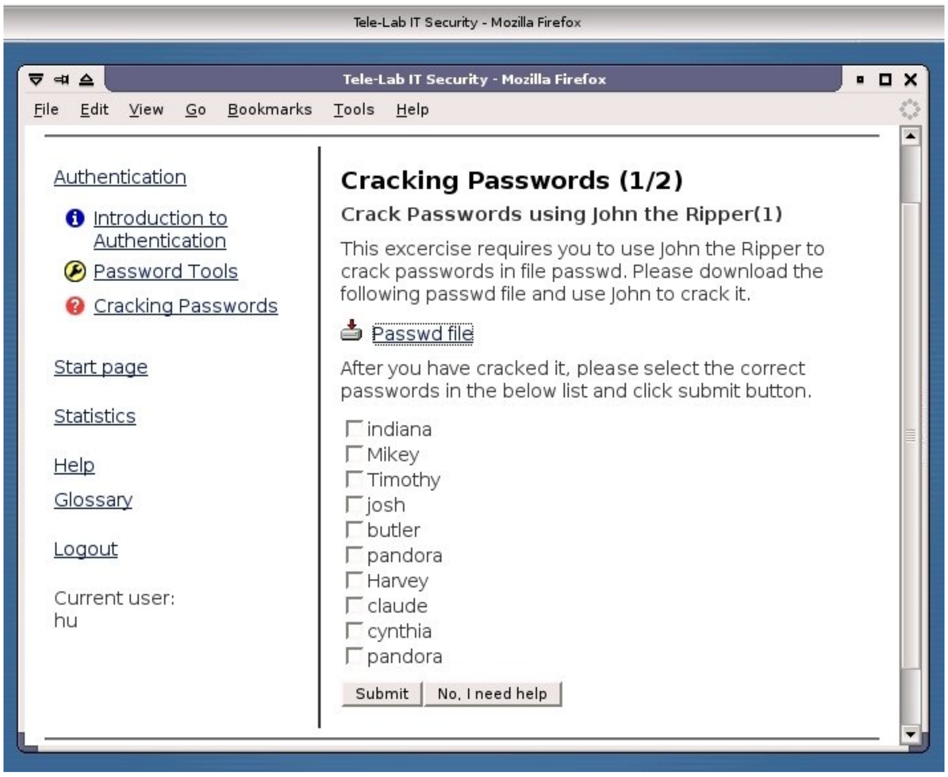
More examples are given in Appendix A and Appendix B.



Now, you got some passwords cracked. You want to retrieve them:

```
user0:America:9:100::/home/user0:/bin/bash
user1:cfi:7:100::/home/user1:/bin/bash
user2:angus:0:100::/home/user2:/bin/bash
user3:advil:8:100::/home/user3:/bin/bash
user4:watson:3:100::/home/user4:/bin/bash
```

Figure 8.3.: The password cracker tutorial created by Flash.



Cracking Passwords (1/2)

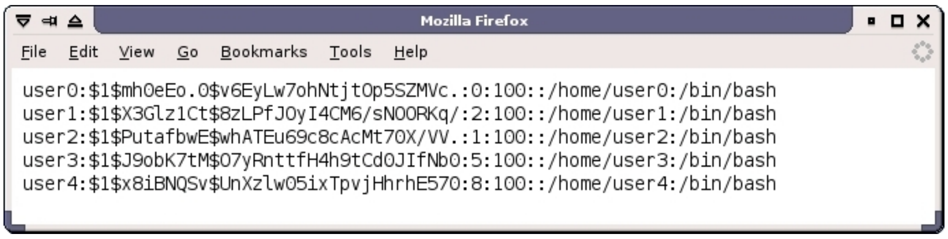
Crack Passwords using John the Ripper(1)

This excercise requires you to use John the Ripper to crack passwords in file passwd. Please download the following passwd file and use John to crack it.

After you have cracked it, please select the correct passwords in the below list and click submit button.

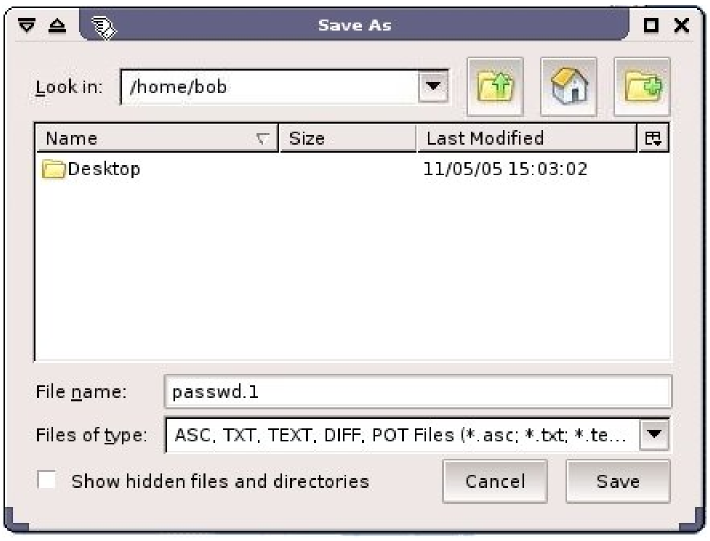
- [ ] indiana
- [ ] Mikey
- [ ] Timothy
- [ ] josh
- [ ] butler
- [ ] pandora
- [ ] Harvey
- [ ] claude
- [ ] cynthia
- [ ] pandora

Figure 8.4.: The password cracking exercise.



```
user0:$1$mh0eEo.0$v6EyLw7ohNtjtOp5SZMVc.:0:100::/home/user0:/bin/bash
user1:$1$X3Glz1Ct$8zLPfJOyI4CM6/sNOORKq/:2:100::/home/user1:/bin/bash
user2:$1$PutafbwE$whATEu69c8cAcMt70X/VV.:1:100::/home/user2:/bin/bash
user3:$1$J9obK7tM$O7yRnttfH4h9tCd0JIfNb0:5:100::/home/user3:/bin/bash
user4:$1$x8iBNQSv$UnXzlw05ixTpvjHhrhE570:8:100::/home/user4:/bin/bash
```

Figure 8.5.: Content of a Linux "passwd" file.



Look in: /home/bob

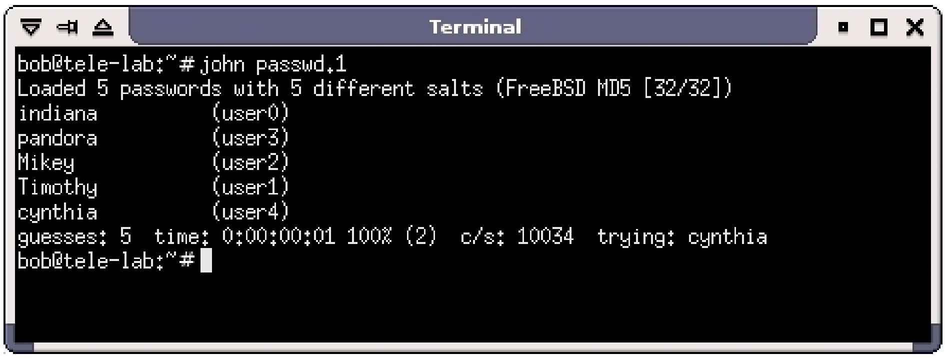
| Name | Size | Last Modified |
|---|---|---|
| Desktop | | 11/05/05 15:03:02 |

File name: passwd.1

Files of type: ASC, TXT, TEXT, DIFF, POT Files (*.asc; *.txt; *.te...

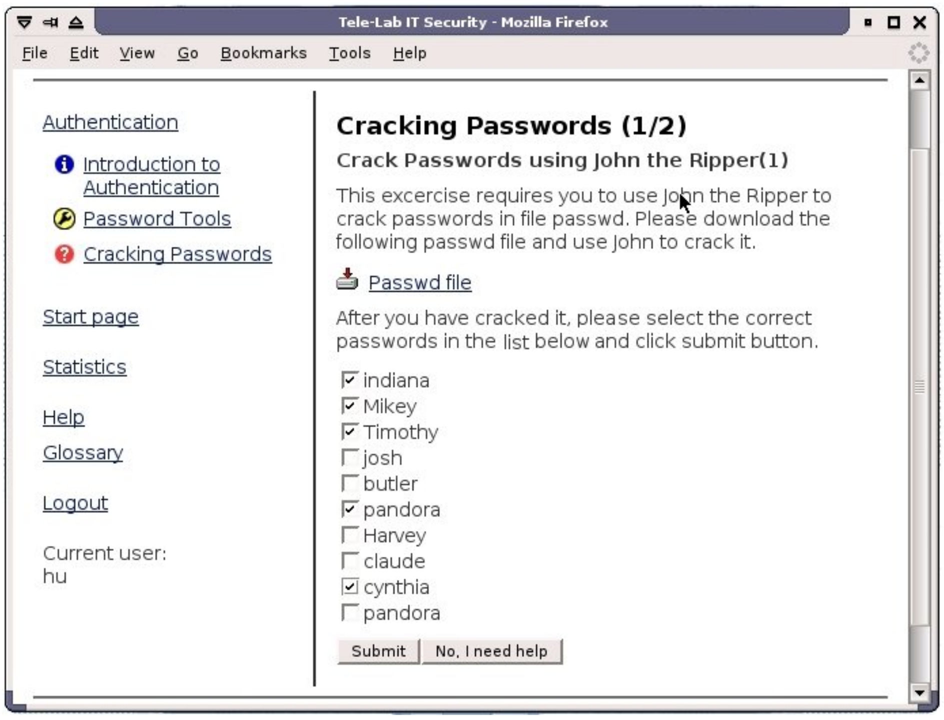
☐ Show hidden files and directories

Figure 8.6.: Downloading the "passwd" file.



```
bob@tele-lab:~#john passwd.1
Loaded 5 passwords with 5 different salts (FreeBSD MD5 [32/32])
indiana          (user0)
pandora          (user3)
Mikey            (user2)
Timothy          (user1)
cynthia          (user4)
guesses: 5  time: 0:00:00:01 100% (2)  c/s: 10034  trying: cynthia
bob@tele-lab:~#
```

Figure 8.7.: Cracking the "passwd" file in a root shell.


Tele-Lab IT Security - Mozilla Firefox
File Edit View Go Bookmarks Tools Help
Authentication
Introduction to Authentication
Password Tools
Cracking Passwords
Start page
Statistics
Help
Glossary
Logout
Submit
No, I need help

Current user:
hu

Cracking Passwords (1/2)

Crack Passwords using John the Ripper(1)

This excercise requires you to use John the Ripper to crack passwords in file passwd. Please download the following passwd file and use John to crack it.

Passwd file

After you have cracked it, please select the correct passwords in the list below and click submit button.

- [✓] indiana
- [✓] Mikey
- [✓] Timothy
- [ ] josh
- [ ] butler
- [✓] pandora
- [ ] Harvey
- [ ] claude
- [✓] cynthia
- [ ] pandora

Figure 8.8.: Submitting the cracking result.

# 9. Conclusions

This report has presented Tele-Lab IT-Security which is a novel virtual machine architecture for creating online IT security laboratories. It allows students not only to learn security concepts and principles, but also to experiment security and gain hands-on experiences in a lightweight and safe laboratory environment. In addition to the architecture itself, a set of technical solutions has been proposed to address the requirements of functionality, reliability, security and performance. Tele-Lab IT-Security has following advantages which distinguish it from other security teaching/learning approaches:

- Besides a tutoring system for learning theoretical principles, Tele-Lab offers hands-on experiences.
- Tele-Lab provides a real-life laboratory environment instead of limited simulation. It mobilizes the access to learning as well as reduces the costs for creation and maintenance of laboratory platforms.
- Tele-Lab has a thin web user interface through which security tools and programs are accessible via a VNC applet without additional client software.
- Although risky security experiments are allowed in Tele-Lab, it is managed in a reliable and available manner. The misuses of laboratory resources are prevented by security isolation.

Future work will focus on extending interfaces for other virtual machines software such as Xen, VMWare, etc. By doing so, we can offer users more security experiences on different operating systems like Windows platforms.

In summary, Tele-Lab IT-Security is a research work attempting to bridge the gap between e-learning and practical IT security education. Tele-Lab is not going to completely substitute the conventional laboratory teaching but add practical features to e-learning. Our work has demonstrated the possibility to implement hands-on security laboratories on the Internet reliably, securely, and economically.

# Bibliography

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

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

[Esslinger, 2002] Esslinger, B. (2002). Cryptool - spielerischer einstieg in klassische und moderne kryptographie: Neue version - fundierte awareness in deutsch und englisch. *Datenschutz und Datensicherheit*, 26(10).

[Goldberg, 1974] Goldberg, R. P. (1974). Survey of virtual machine research. *IEEE Computer*, pages 34–45.

[Hoffman et al., 2003] Hoffman, L., Dodge, R., Rosenberg, T., and Ragsdale, D. (2003). Information assurance laboratory innovations. In *Proceedings of the 7th Colloquium for Information Systems Security Education*, Washington, DC, USA.

[Hu and Meinel, 2004] Hu, J. and Meinel, C. (2004). Tele-Lab "IT-Security"on CD: Portable, reliable and safe IT Security training. *Computers & Security, Elsevier*, 23(4):282–289.

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

[Irvine and Thompson, 2004] Irvine, C. E. and Thompson, M. F. (2004). Expressing an information security policy within a security simulation game. In *Proceedings of the Sixth Workshop on Education in Computer Security (WECS6)*, pages 43–49, Monterey, USA.

[Lindskog et al., 1999] Lindskog, S., Lindqvist, U., and Jonsson, E. (1999). IT security research and education in synergy. In *Proceedings of the 1st World Conference on Information Security Education*, Stockholm, Sweden.

[McVoy and Staelin, 1996] McVoy, L. and Staelin, C. (1996). Lmbench: Portable tools for performance analysis. In *Proceedings of the 1996 USENIX Annual Technical Conference*, pages 279–294, Berkeley.

[Ragsdale et al., 2003] Ragsdale, D., Lathrop, S., and Dodge, R. (2003). Enhancing information warfare education through the use of virtual and isolated networks. *Journal of Information Warfare*, 2(3):53–65.

[Richardson et al., 1998] Richardson, T., Stafford-Fraser, Q., Wood, K. R., and Hopper, A. (1998). Virtual network computing. *IEEE Internet Computing*, 2(1):33–38.

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

[Schillings and Meinel, 2002] Schillings, V. and Meinel, C. (2002). Tele-TASK - tele-teaching anywhere solution kit. In *Proceedings of ACM SIGUCCS 2002*, Providence, USA.

[Snyder, 1990] Snyder, P. (1990). tmpfs: A virtual memory file system. In *Proceedings of the European UNIX Users Group Conference*, pages 241–248.

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

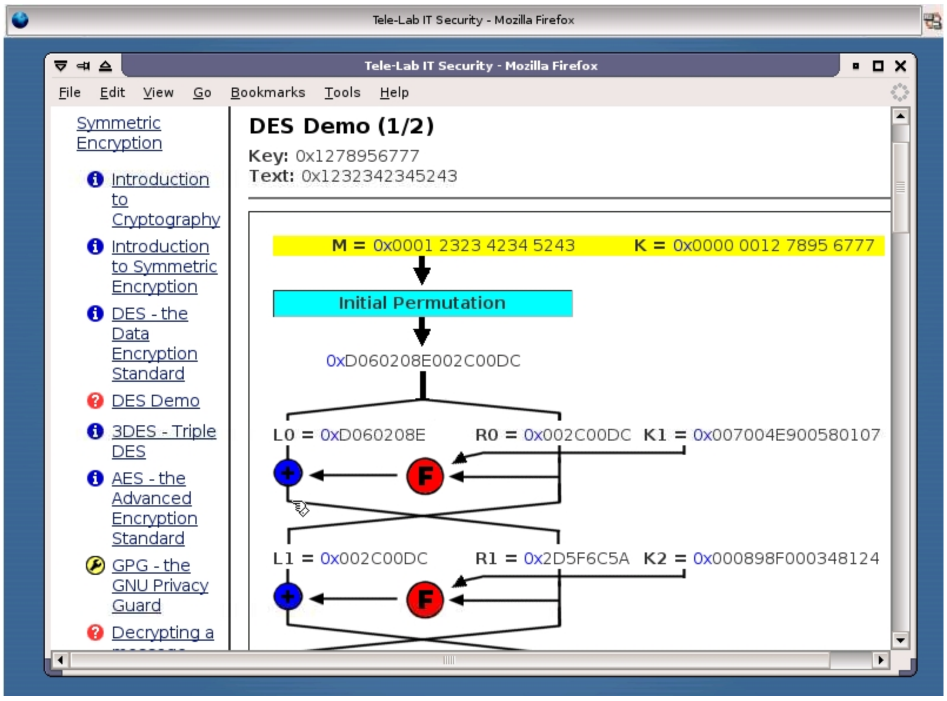
[Vigna, 2003] Vigna, G. (2003). Teaching hands-on network security: Testbeds and live exercises. *Journal of Information Warfare*, 2(3):8–24.

[Woo et al., 2002] Woo, C., Choi, J., and Evens, M. (2002). Web-based ITS for training system managers on the computer intrusion. In *Proceedings of the 6th International conference on Intelligent Tutoring Systems*, pages 311–319, Biarritz, France and San Sebastian, Spain.

# A. Symmetric encryption demonstration

The Tele-Lab chapter, “Symmetric Encryption”, introduces important concepts on symmetric encryption, illustrates related algorithms, and provides practical exercises. In the exercises, a user can apply a open-source encryption tool, GnuPG, to encrypt or decrypt messages. Following contents and exercises are provided in this chapter:

- Essential cryptographic concepts and typical cipher algorithms are introduced in early sections (see Figure A.1). E.g. in order to explain the DES algorithm[1], we integrated a visual demonstration from which a user can run explore the algorithm and watch en-/decryption details .



DES Demo (1/2)

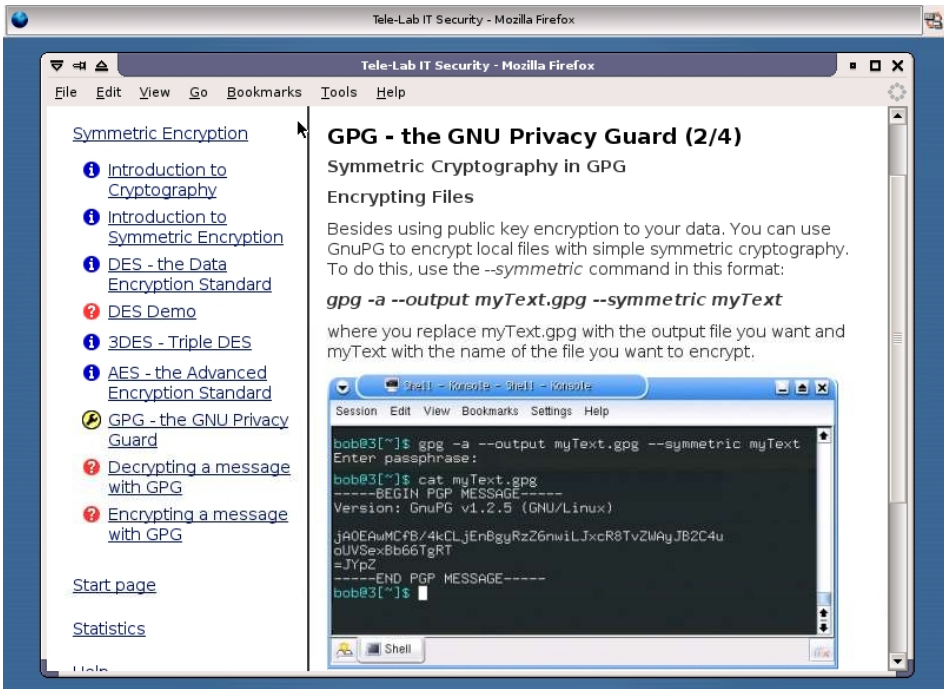
**Key:** 0x1278956777
**Text:** 0x1232342345243

Tele-Lab IT Security - Mozilla Firefox
File Edit View Go Bookmarks Tools Help
Symmetric Encryption
Introduction to Cryptography
Introduction to Symmetric Encryption
DES - the Data Encryption Standard
DES Demo
3DES - Triple DES
AES - the Advanced Encryption Standard
GPG - the GNU Privacy Guard
Decrypting a
M = 0x0001 2323 4234 5243
K = 0x0000 0012 7895 6777
Initial Permutation
0xD060208E002C00DC
L0 = 0xD060208E
R0 = 0x002C00DC
K1 = 0x007004E900580107
F
L1 = 0x002C00DC
R1 = 0x2D5F6C5A
K2 = 0x000898F000348124
F

Figure A.1.: DES demonstration.

- The tutorial section introduces an encryption tool, GPG (“*GNU Privacy Guard*”) which is an opensource PGP (“*Pretty Good Privacy*”) program. From this tutorial, the user learns how to apply encryption technology and tools to meet everyday needs (see Figure A.2).

[1]DES is a significant symmetric algorithm which includes 16 encryption rounds.


Tele-Lab IT Security - Mozilla Firefox
Tele-Lab IT Security - Mozilla Firefox
File Edit View Go Bookmarks Tools Help
Symmetric Encryption
Introduction to Cryptography
Introduction to Symmetric Encryption
DES - the Data Encryption Standard
DES Demo
3DES - Triple DES
AES - the Advanced Encryption Standard
GPG - the GNU Privacy Guard
Decrypting a message with GPG
Encrypting a message with GPG
Start page
Statistics
Shell - Konsole - Shell - Konsole
Session Edit View Bookmarks Settings Help
Shell

**GPG - the GNU Privacy Guard (2/4)**

**Symmetric Cryptography in GPG**

**Encrypting Files**

Besides using public key encryption to your data. You can use GnuPG to encrypt local files with simple symmetric cryptography. To do this, use the *--symmetric* command in this format:

***gpg -a --output myText.gpg --symmetric myText***

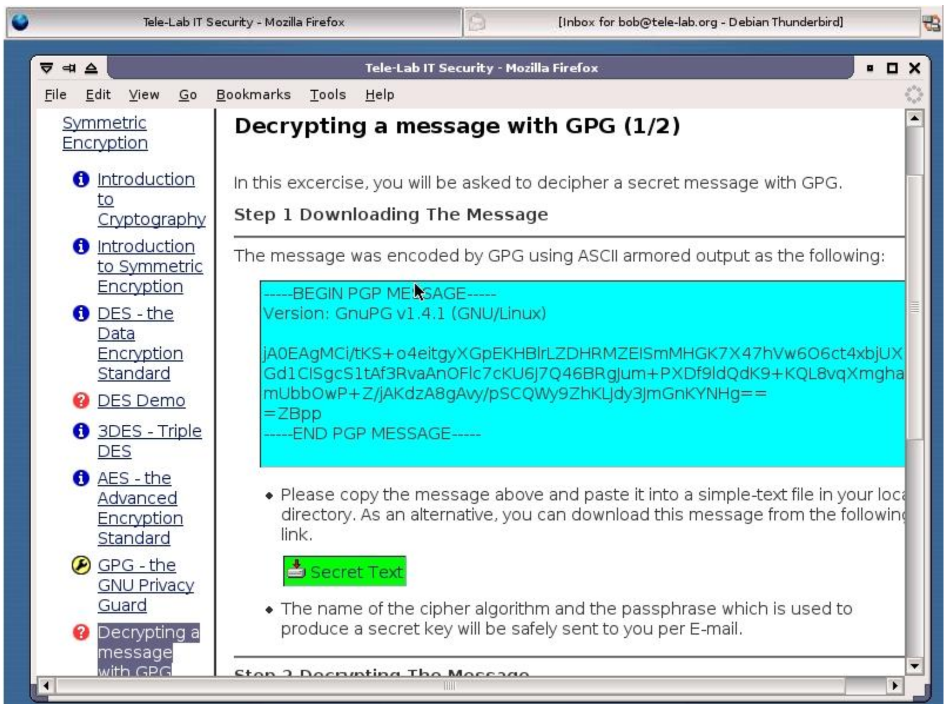
where you replace myText.gpg with the output file you want and myText with the name of the file you want to encrypt.

```
bob@3[~]$ gpg -a --output myText.gpg --symmetric myText
Enter passphrase:
bob@3[~]$ cat myText.gpg
-----BEGIN PGP MESSAGE-----
Version: GnuPG v1.2.5 (GNU/Linux)

jA0EAwMCfB/4kCLjEnBgyRzZ6nwiLJxcR8TvZWAyJB2C4u
oUVSexBb66TgRT
=JYpZ
-----END PGP MESSAGE-----
bob@3[~]$
```

Figure A.2.: The GnuPG tutorial.

- In the exercise sections, encryption and decryption exercises are assigned to the user. The process of the decryption exercise includes the following steps: First, the tutor creates four texts and one of them is randomly chosen. The chosen text is then encrypted by GPG in the background and the corresponding secret message is produced (see Figure A.3).
- The task for the user is to select the right plaintext of a list of four possible plaintexts. In order to finish this task, the user must download the secret message, find the key from email, and decrypt it with GPG. Then the user has to submit the plaintext he/she have found (see Figure A.4).
- After submission of the result, the tutor checks it and finds out whether it is the matched answer to the question. The evaluation result will be shown to the user and recorded into his/her user profile. In case of fail, the tutor will ask him or her to have another try.
- The encryption process is relatively simple. The tutor asks the user to encrypt a text. The user has to do encryption and submit the secret text with a key which is used for encryption. Then the tutor tries decrypting the submission. If it can successfully decrypt the text with that key, this means the user has done a right job, otherwise he/she is asked to try again.


Tele-Lab IT Security - Mozilla Firefox
[Inbox for bob@tele-lab.org - Debian Thunderbird]
File Edit View Go Bookmarks Tools Help
Symmetric Encryption
Introduction to Cryptography
Introduction to Symmetric Encryption
DES - the Data Encryption Standard
DES Demo
3DES - Triple DES
AES - the Advanced Encryption Standard
GPG - the GNU Privacy Guard
Decrypting a message with GPG
Secret Text

**Decrypting a message with GPG (1/2)**

In this excercise, you will be asked to decipher a secret message with GPG.

**Step 1 Downloading The Message**

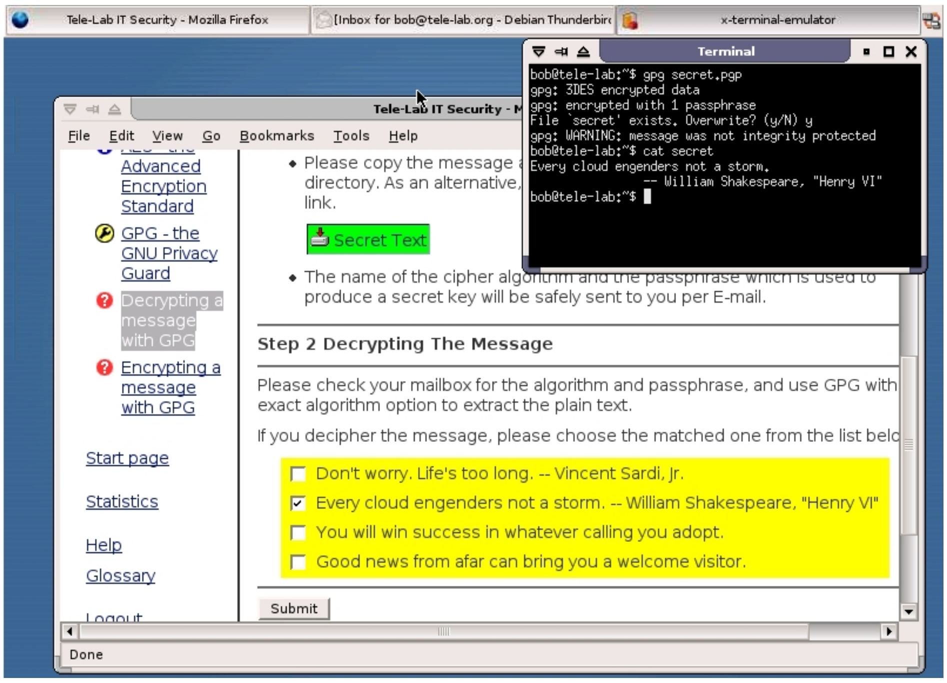
The message was encoded by GPG using ASCII armored output as the following:

```
-----BEGIN PGP MESSAGE-----
Version: GnuPG v1.4.1 (GNU/Linux)

jA0EAgMCi/tKS+o4eitgyXGpEKHBlrLZDHRMZEISmMHGK7X47hVw6O6ct4xbjUX
Gd1ClSgcS1tAf3RvaAnOFlc7cKU6J7Q46BRgJum+PXDf9ldQdK9+KQL8vqXmgha
mUbbOwP+Z/jAKdzA8gAvy/pSCQWy9ZhKLJdy3JmGnKYNHg==
=ZBpp
-----END PGP MESSAGE-----
```

- Please copy the message above and paste it into a simple-text file in your loca directory. As an alternative, you can download this message from the followin link.
- The name of the cipher algorithm and the passphrase which is used to produce a secret key will be safely sent to you per E-mail.

Figure A.3.: The task of the GPG decryption exercise.


Tele-Lab IT Security - Mozilla Firefox
[Inbox for bob@tele-lab.org - Debian Thunderbird
x-terminal-emulator
Terminal
Tele-Lab IT Security
File Edit View Go Bookmarks Tools Help
Advanced Encryption Standard
GPG - the GNU Privacy Guard
Decrypting a message with GPG
Encrypting a message with GPG
Start page
Statistics
Help
Glossary
Logout
Secret Text
Submit
Done

```
bob@tele-lab:~$ gpg secret.pgp
gpg: 3DES encrypted data
gpg: encrypted with 1 passphrase
File `secret' exists. Overwrite? (y/N) y
gpg: WARNING: message was not integrity protected
bob@tele-lab:~$ cat secret
Every cloud engenders not a storm.
                -- William Shakespeare, "Henry VI"
bob@tele-lab:~$
```

- Please copy the message directory. As an alternative, link.
- The name of the cipher algorithm and the passphrase which is used to produce a secret key will be safely sent to you per E-mail.

**Step 2 Decrypting The Message**

Please check your mailbox for the algorithm and passphrase, and use GPG with exact algorithm option to extract the plain text.

If you decipher the message, please choose the matched one from the list belo

- ☐ Don't worry. Life's too long. -- Vincent Sardi, Jr.
- ☑ Every cloud engenders not a storm. -- William Shakespeare, "Henry VI"
- ☐ You will win success in whatever calling you adopt.
- ☐ Good news from afar can bring you a welcome visitor.

Figure A.4.: Completion of the GPG decryption exercise.

# B. Secure email demonstration

This chapter is intended to familiarize a user with digital certificates so that he/she can sign and encrypt email. Exercises in this chapter simulate everyday email communications which need a complicated and interactive user work environment. The learning process is described as follows:

- The tutor presents concepts of secure email, which include security requirements for email, measures to protect email such as encryption and digital signatures, and secure email standards like *OpenPGP*[1] and *SMIME*[2].
- The *Thunderbird* email client is introduced. Thunderbird is a popular email tool used on Linux and Windows. It integrates a SMIME module and an OpenPGP module called *Enigmail*. The SMIME module is the default security feature of Thunderbird and Enigmail is a plug-in which encrypts or digitally signs email by GnuPG. The tutorials for both email security tools are presented to illustrate how to apply them to secure email.
- Two sets of tasks are designed for SMIME and Enigmail respectively. In both tasks, the user is required to securely exchange email with a virtual partner.

## B.1. The SMIME exercise

As shown in Figure B.1, the SMIME exercise requires an interactive laboratory environment. In this environment, OpenSSL[3], a certificate authority (CA), related mail services and accounts have to be provided. For effective interaction, a virtual email partner named "*Alice*" is also created. This partner account is necessary because, without it, a user does not know with whom to exchange messages. In this exercise, the user is asked to manage digital certificates, create/verify signature of messages, and encrypt/decrypt messages. The procedure that a user creates a signature and sends a signed mail to Alice is described below:

1. Work Environment Preparation.
    - The tutor generates a virtual user, Alice, i.e. creates and configures her Linux account and email settings.

[1]"OpenPGP" is an encryption scheme based on PGP for interpreting and sending digitally signed and encrypted messages.
[2]"SMIME" (Secure Multipurpose Internet Mail Extensions) is the Internet standard for secure e-mail attachments based on RSA and MIME (Multipurpose Internet Mail Extensions).
[3]OpenSSL is a cryptography toolkit implementing the Secure Sockets Layer (SSL v2/v3) and Transport Layer Security (TLS v1) network protocols and related cryptography standards (`http://www.openssl.org/`).

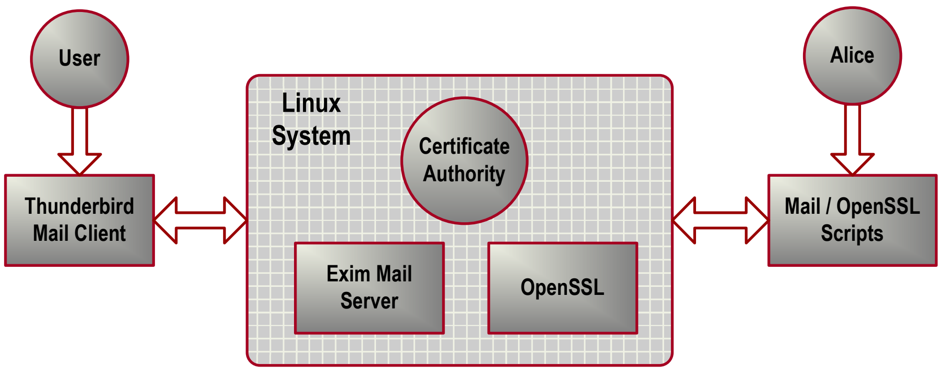


Figure B.1.: Working environment of the SMIME exercise

- The tutor creates a certificate authority by generating a root certificate and setting up a certificate service.
- The tutor issues a certificate for Alice and configures her mail account settings with this certificate.

2. Exercise Execution.

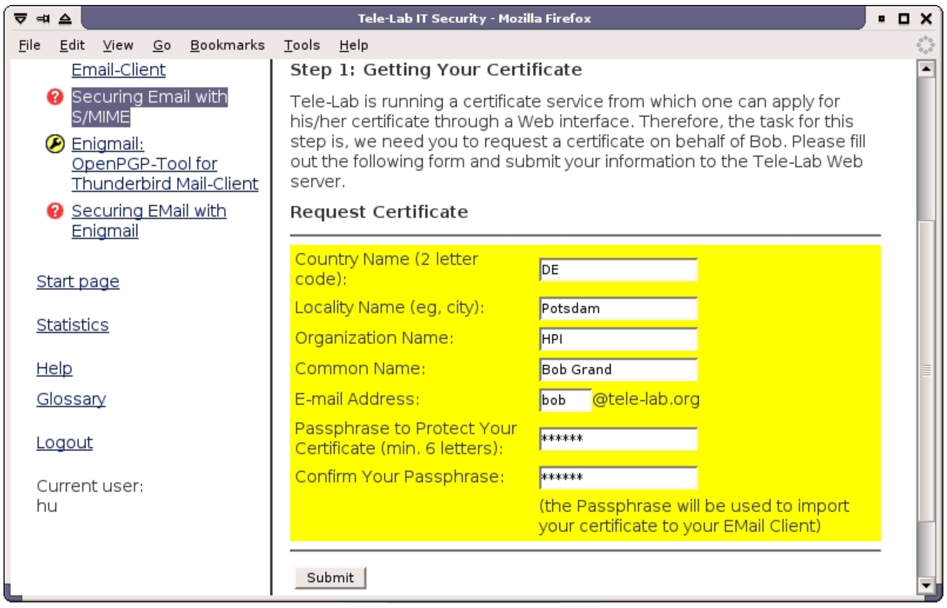
    - Importing the certificate. *(1)* A certificate request page is shown to the user. He/she can fill out the certificate request form and apply for a certificate from the CA (see Figure B.2). *(2)* The user's private key is created and a corresponding certificate file (in *PKCS12*[4] format) is generated. The user is asked to download his/her certificate file and import it to his/her mail client. *(3)* The user opens the Thunderbird's SMIME certificate manager, imports the certificate, and trusts it for identifying email users (see Figure B.3). *(4)* The user also needs to configure his/her mail account to use the imported certificate.
    - Signing messages. The user is asked to write to Alice with a digital signature (shown in Figure B.4). The user must compose a message, sign it with his/her private key, and send it to Alice (see Figure B.5).

3. Result Evaluation

    - To evaluate the result, the tutor behaviors as Alice in the background. It first checks Alice's inbox and fetches the message sent by the user.
    - The tutor verifies its signature by OpenSSL scripts and confirms whether it has been sent from the student and matches with the origin (see Figure B.6).
    - If verification succeeds, the user will receive a confirmation message and the completion of the task will be recorded into the user profile.

The message encryption/decryption task is arranged after the message signing task and follows a similar procedure:

[4]PKCS 12 is the RSA personal information exchange syntax standard which describes a portable format for storage and transportation of user private keys, certificates etc.



Tele-Lab IT Security - Mozilla Firefox
File Edit View Go Bookmarks Tools Help
Email-Client
Securing Email with S/MIME
Enigmail: OpenPGP-Tool for Thunderbird Mail-Client
Securing EMail with Enigmail
Start page
Statistics
Help
Glossary
Logout

Current user: hu

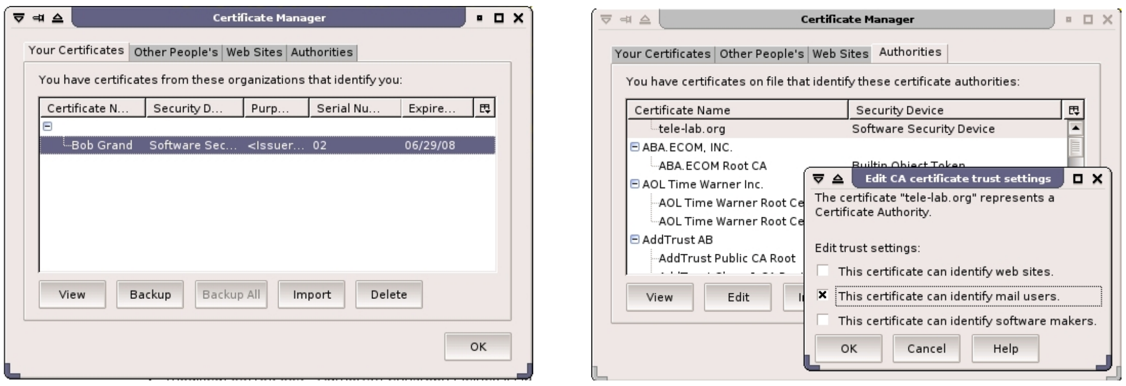
**Step 1: Getting Your Certificate**

Tele-Lab is running a certificate service from which one can apply for his/her certificate through a Web interface. Therefore, the task for this step is, we need you to request a certificate on behalf of Bob. Please fill out the following form and submit your information to the Tele-Lab Web server.

**Request Certificate**

| | |
|---|---|
| Country Name (2 letter code): | DE |
| Locality Name (eg, city): | Potsdam |
| Organization Name: | HPI |
| Common Name: | Bob Grand |
| E-mail Address: | bob @tele-lab.org |
| Passphrase to Protect Your Certificate (min. 6 letters): | ****** |
| Confirm Your Passphrase: | ****** |
| | (the Passphrase will be used to import your certificate to your EMail Client) |

Figure B.2.: Requesting a personal certificate.



You have certificates from these organizations that identify you:

| Certificate N... | Security D... | Purp... | Serial Nu... | Expire... |
|---|---|---|---|---|
| Bob Grand | Software Sec... | <Issuer... | 02 | 06/29/08 |

You have certificates on file that identify these certificate authorities:

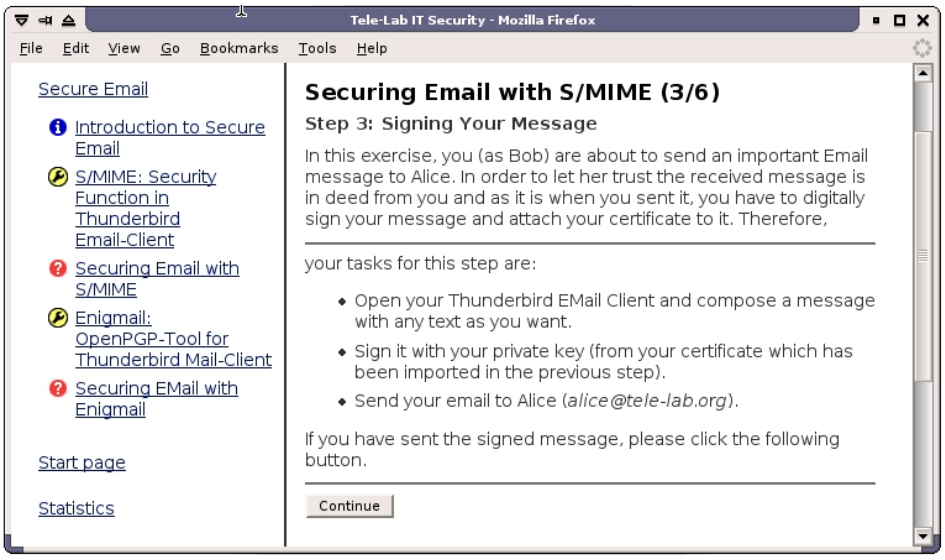
| Certificate Name | Security Device |
|---|---|
| tele-lab.org | Software Security Device |
| ABA.ECOM, INC. | |
| ABA.ECOM Root CA | Builtin Object Token |
| AOL Time Warner Inc. | |
| AOL Time Warner Root Ce | |
| AOL Time Warner Root Ce | |
| AddTrust AB | |
| AddTrust Public CA Root | |

The certificate "tele-lab.org" represents a Certificate Authority.

Edit trust settings:

☐ This certificate can identify web sites.
☒ This certificate can identify mail users.
☐ This certificate can identify software makers.

Figure B.3.: Import and configuration of a personal certificate.



Tele-Lab IT Security - Mozilla Firefox
File Edit View Go Bookmarks Tools Help
Secure Email
Introduction to Secure Email
S/MIME: Security Function in Thunderbird Email-Client
Securing Email with S/MIME
Enigmail: OpenPGP-Tool for Thunderbird Mail-Client
Securing EMail with Enigmail
Start page
Statistics
Continue

**Securing Email with S/MIME (3/6)**

**Step 3: Signing Your Message**

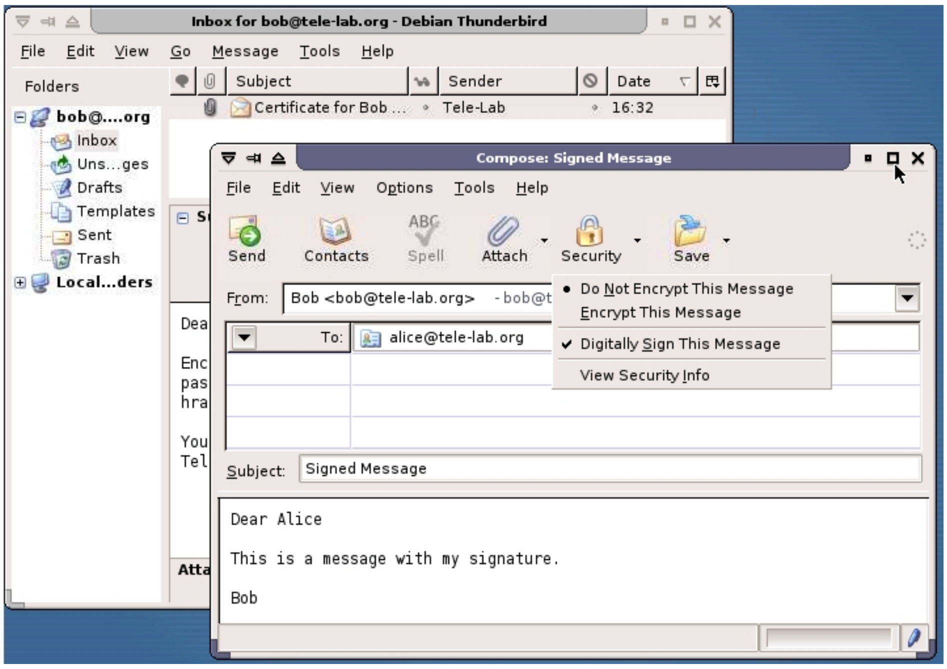
In this exercise, you (as Bob) are about to send an important Email message to Alice. In order to let her trust the received message is in deed from you and as it is when you sent it, you have to digitally sign your message and attach your certificate to it. Therefore,

your tasks for this step are:

- Open your Thunderbird EMail Client and compose a message with any text as you want.
- Sign it with your private key (from your certificate which has been imported in the previous step).
- Send your email to Alice (*alice@tele-lab.org*).

If you have sent the signed message, please click the following button.

Figure B.4.: The task to digitally sign a message.



From: Bob <bob@tele-lab.org> - bob@t

To: alice@tele-lab.org

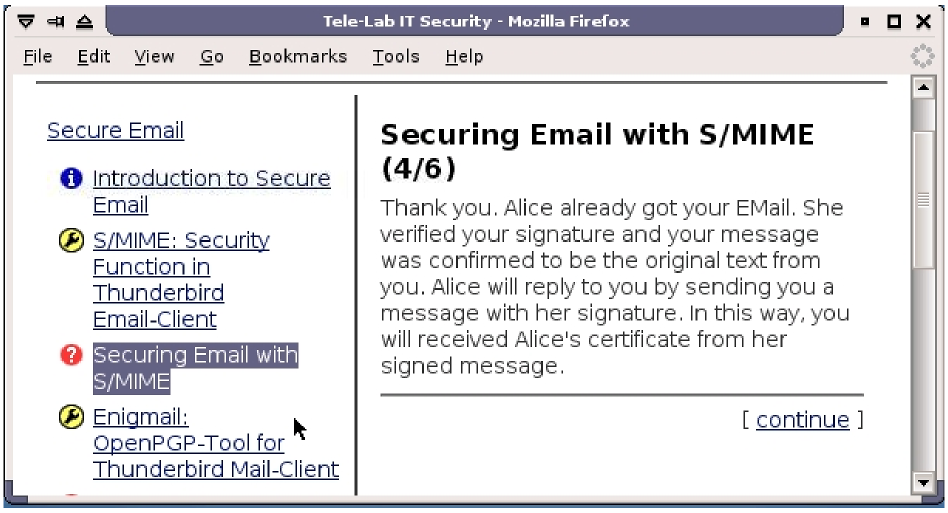
Subject: Signed Message

```
Dear Alice

This is a message with my signature.

Bob
```

Figure B.5.: Signing a message.


Tele-Lab IT Security - Mozilla Firefox
File Edit View Go Bookmarks Tools Help
Secure Email
Introduction to Secure Email
S/MIME: Security Function in Thunderbird Email-Client
Securing Email with S/MIME
Enigmail: OpenPGP-Tool for Thunderbird Mail-Client
[ continue ]

**Securing Email with S/MIME (4/6)**

Thank you. Alice already got your EMail. She verified your signature and your message was confirmed to be the original text from you. Alice will reply to you by sending you a message with her signature. In this way, you will received Alice's certificate from her signed message.

Figure B.6.: Verifying the signature for evaluation.

- After the user receives the signed message of Alice and imports her certificate, he/she is asked to decode a message from Alice. The message is encrypted by Alice with the user's own public key.
- The user then encrypts a message with Alice's public key and sends it to Alice. The tutor will try to decrypt the message with Alice's private key corresponding to her certificate and record the (un-)successful completion of the task in the user profile.

## B.2. The Enigmail exercise

The Enigmail exercise includes the tasks to manipulate public-private key pairs, create/verify signature of messages, and encrypt/decrypt messages. In fact, the principle of the Enigmail exercise is very similar to that of SMIME (see Figure B.7). It also needs a virtual partner, Alice. The only deference between both is that Enigmail is based on open PGP which uses a trust web model and has no central certificate authority. Certificates (PGP key pairs) and trust have to be managed manually. The steps to prepare key pairs include:

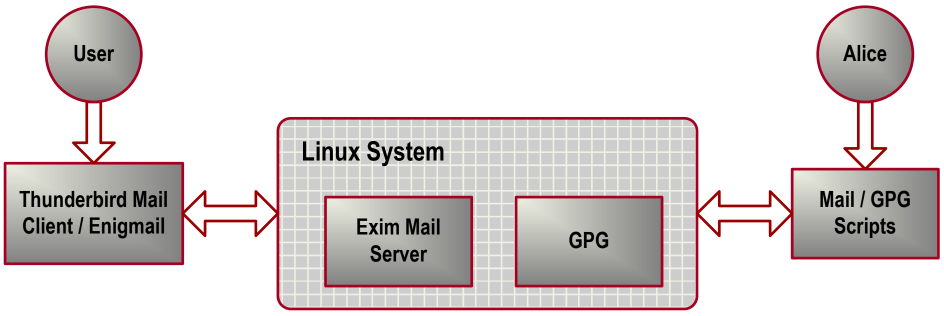
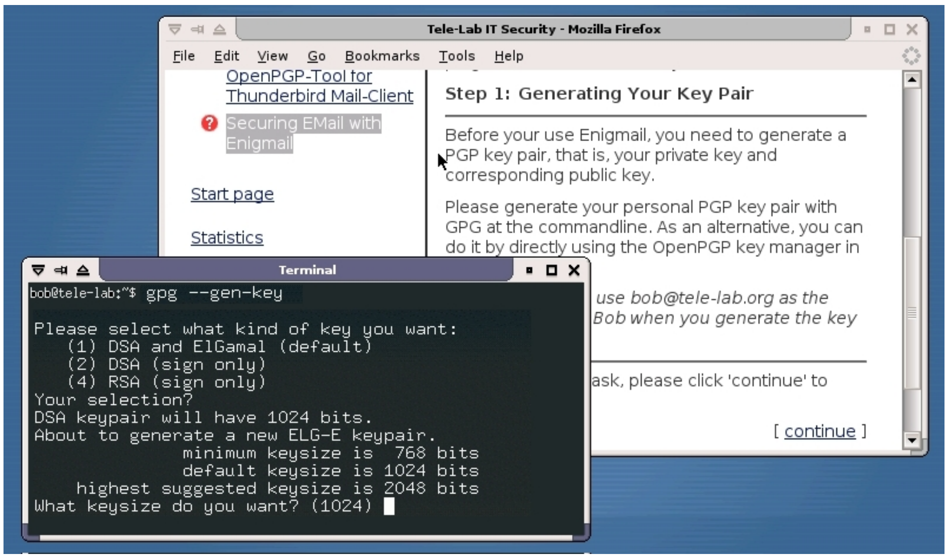


Figure B.7.: The work environment of the Enigmail exercise.

- First, the user must create a key pair of his/her own by using the GPG tool and then configure his/her Enigmail settings of Thunderbird with the created key pair (see Figure B.8).
- The user publishes his/her public key and obtains those of others. The user uploads the key to the tutor and the tutor will configure it in Alice's mail account. The user can download Alice's public key from the tutor. Thus, the user and Alice can exchange public keys with each other (see Figure B.9).

With key pairs available and public key exchanged, the rest tasks of email signing and encryption follow similar processes to those of the SMIME exercise. The difference is that the user will use OpenPGP key pairs instead of certificates, and the tutor will use GPG scripts in the background to run interactive tasks.


Tele-Lab IT Security - Mozilla Firefox
File Edit View Go Bookmarks Tools Help
OpenPGP-Tool for Thunderbird Mail-Client
Securing EMail with Enigmail
Start page
Statistics
Terminal
[ continue ]

Step 1: Generating Your Key Pair

Before your use Enigmail, you need to generate a PGP key pair, that is, your private key and corresponding public key.

Please generate your personal PGP key pair with GPG at the commandline. As an alternative, you can do it by directly using the OpenPGP key manager in

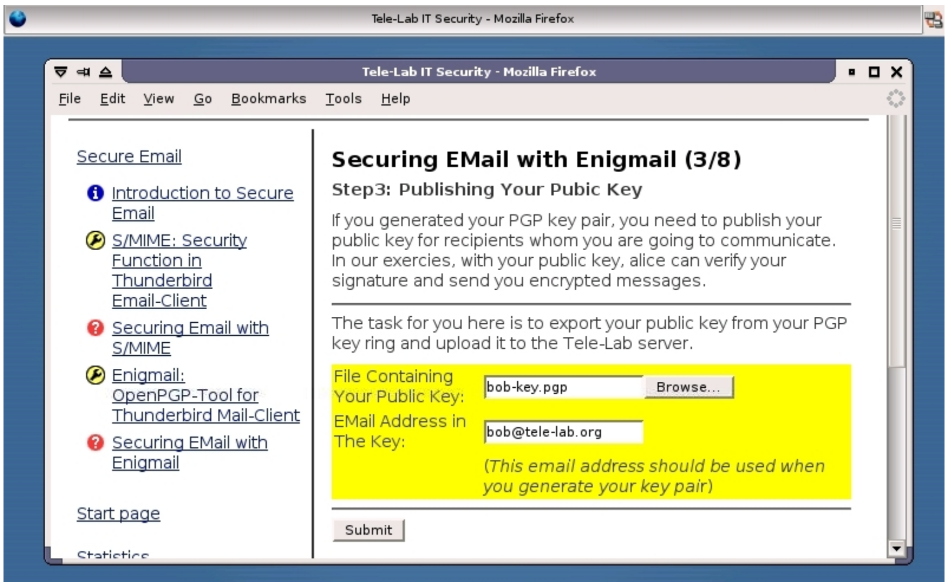
*use bob@tele-lab.org as the Bob when you generate the key*

ask, please click 'continue' to

```
bob@tele-lab:~$ gpg --gen-key

Please select what kind of key you want:
   (1) DSA and ElGamal (default)
   (2) DSA (sign only)
   (4) RSA (sign only)
Your selection?
DSA keypair will have 1024 bits.
About to generate a new ELG-E keypair.
              minimum keysize is  768 bits
              default keysize is 1024 bits
    highest suggested keysize is 2048 bits
What keysize do you want? (1024)
```

Figure B.8.: Creating a PGP keypair.


Tele-Lab IT Security - Mozilla Firefox
Tele-Lab IT Security - Mozilla Firefox
File Edit View Go Bookmarks Tools Help
Secure Email
Introduction to Secure Email
S/MIME: Security Function in Thunderbird Email-Client
Securing Email with S/MIME
Enigmail: OpenPGP-Tool for Thunderbird Mail-Client
Securing EMail with Enigmail
Start page
Statistics
Browse...
Submit

Securing EMail with Enigmail (3/8)

Step3: Publishing Your Pubic Key

If you generated your PGP key pair, you need to publish your public key for recipients whom you are going to communicate. In our exercies, with your public key, alice can verify your signature and send you encrypted messages.

The task for you here is to export your public key from your PGP key ring and upload it to the Tele-Lab server.

| | |
|---|---|
| File Containing Your Public Key: | bob-key.pgp |
| EMail Address in The Key: | bob@tele-lab.org |
| | *(This email address should be used when you generate your key pair)* |

Figure B.9.: Publishing the public key.