OpenLabs Security Laboratory – The Online Security Experiment Platform

J. Zackrisson, C. Svahnberg
Blekinge Institute of Technology/School of engineering, Ronneby, Sweden

***Abstract*—For experiments to be reproducible, it is important to have a known and controlled environment. This requires isolation from the surroundings. For security experiments, e.g. with hostile software, this is even more important as the experiment can affect the environment in adverse ways. In a normal campus laboratory, isolation can be achieved by network separation. For an online environment, where remote access is essential, full separation is not possible, and therefore the security implications must be carefully considered.**

In this paper, a way to enable remote experiments is described, where users are given full control over the computer installation. Automating the install procedure and dynamically creating isolated experiment networks provide remote users with the tools needed to do experiments in a reproducible and secure environment. The installation is done by automatically installing prepared clone images, which contains a hard drive representation, to the machines own hard drive. When an experiment starts, the machines in the experiment are automatically booted and placed on an isolated (switched) network. This way experiments are isolated from each other. By using firewall techniques, connections from outside the laboratory can only be made from the user's machine into his/her experiment. When an experiment is done, all data are erased from the machine's hard drive.

***Index Terms*—Computer security, e-Learning, Remote handling, Student experiments, Training.**

I. INTRODUCTION

In 2006, Blekinge Institute of Technology (BTH) decided to create a remote laboratory for advanced security exercises. In the laboratory, students should be able to experiment with insecure protocols, software vulnerabilities and other harmful software, e.g. viruses, in a safe and isolated environment.

The remote security laboratory was the next logical step from the campus security laboratory [1] that has been used for advanced security experiments since 2002. The remote security laboratory introduced new challenges in respect of network security and remote control.

To be able to provide remote controlled experiments in security, isolation of the experiments must be guaranteed. No unwanted information should be able to reach the experiment from the outside world and, more importantly, no information should ever be able to escape the experiment environment. Providing isolation is crucial when experimenting with self-replicating code, e.g. viruses, which must not be allowed to spread outside the laboratory environment. Good experiments should also be reproducible, i.e. it should be possible to restart the experiment from a known state.

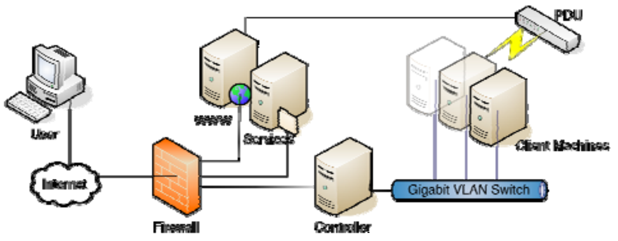


Figure 1. Overview of the remote security laboratory

A user can access the OpenLabs security laboratory at BTH over Internet (See Fig. 1), and make experiment reservations through a web-interface. The same interface will also be used to manage the experiment, e.g. reboot a client machine. When a reservation starts, the client machines will fetch and restore the desired installation from the controller. The controller will also configure the local network, isolating the client machines in one experiment from other concurrent experiments. The firewall rules will be updated, limiting access to the experiment from the users machine. The firewall will also effectively prevent experiments from escaping the laboratory environment. As a consequence, service machines within the laboratory must provide services needed by experiments.

This solution provides a way to isolate experiments from each other and from the outside world. Experiments are also reproducible, since the client machines can restore exactly the same installation any number of times.

This paper will focus on the technical aspect and the construction of the laboratory. In section II, a brief usage overview is given, before giving an in-depth technical description in section III. The chosen solution is discussed and related to in section IV. In Section V other used of the remote security laboratory is explored. Similar laboratories and related work is described in section VI. Finally, the conclusion and future of the project is presented in section VII.

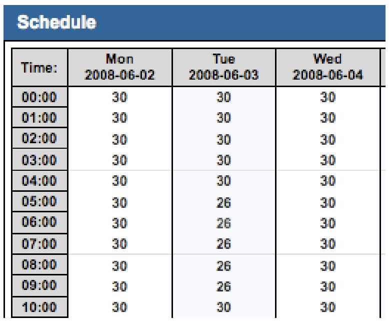
II. USAGE

To give a feeling for how the laboratory is used, we first give a short introduction of the basic usage.

A. Web interface

Through the laboratory's web interface, students can allocate resources and manage their experiments. To get access to the laboratory, the student must first authenticate him/herself. This is done through a login page, where the student enters his/her email address and password. The first time the student enters the page the account has to be activated. The activation process will then generate a password and send it to the user's email address.

After being logged-in, the student can begin making reservations for experiments. The date and time for the experiment can be picked from a schedule display (Fig. 2). The number denotes the number of free machines.

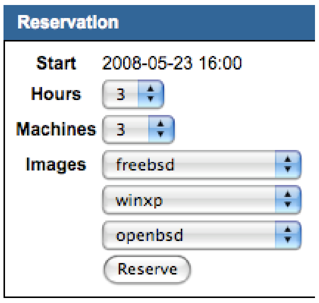


Schedule

Time:	Mon 2008-06-02	Tue 2008-06-03	Wed 2008-06-04
00:00	30	30	30
01:00	30	30	30
02:00	30	30	30
03:00	30	30	30
04:00	30	30	30
05:00	30	26	30
06:00	30	26	30
07:00	30	26	30
08:00	30	26	30
09:00	30	26	30
10:00	30	30	30

Figure 2. Reservation schedule

After the date and time is chosen, the student will have to decide for how long the experiment should last (Fig. 3). Depending on the experiment the student can choose how many and what to install on the machines. In figure 3, the student is about to make a three-hour reservation, with three machines, running FreeBSD, Windows XP and OpenBSD.



Reservation

Start	2008-05-23 16:00
Hours	3
Machines	3
Images	freebsd
	winxp
	openbsd

Figure 3. Machine reservation

B. Clone images

What to install on a machine is described by a clone image, which is a file representing the contents of a hard drive. The administrator or the teacher prepares the clone images in advance, by installing what is needed for the experiment on a machine and then making a clone image from it. Basically any operating system running on x86/x86-64 hardware should work. For example Windows XP, FreeBSD, OpenBSD and Ubuntu have been used successfully.

C. Experimenting

When it is time to experiment, the student will receive an email reminder with all the information needed to begin experimenting. Before being able to remote control the machines of the experiment, the student must first log in and choose to start the experiment from the web page.

When the experiment is started, the machines of the experiment can be remote controlled by common remote control means, such as VNC, RDP (Remote Desktop) or SSH, depending on what the experiment is configured for.

If more than one machine is used in the experiment, they can communicate with each other on an isolated network.

During the experiment, the student has the option to reboot or reinstall the machines of the experiment. That way the student has the option to start the experiment over, or change their mind and install another clone image.

D. Example of an experiment

In a course on malicious software, an installation has been prepared with rootkits, adware and spyware. In this exercise the student has to identify and eliminate the different kinds of malicious software and write a report on the findings. Because of the obtrusive nature of this exercise, it would not be a good idea to do anything similar on the student's own machine or in a normal campus laboratory, without proper isolation. No matter how badly the malicious software or student mess up the machine, the student always has the option of reinstalling and starting over.

E. Isolation

The laboratory is built so that no unwanted traffic can ever leave the laboratory. Remote control is still possible by only letting traffic out as a response to a user-initiated connection. This means that connections from the student's machine to the laboratory are allowed and can be responded upon.

After an experiment is over, all contents of the hard drives of the experiment machines are overwritten with zeroes; this guarantees that information will not accidentally leak to successive experiments. The result is that each experiment is started in a fresh environment.

III. TECHNOLOGY DESCRIPTION

A. Web interface

1) Login

To access the web interface, the users have to be authenticated. The user's email address is used as a unique user identifier. The first time a user wants to access the laboratory, the account has to be activated. This will send a machine-generated password to the user's email address. Using email addresses provide a more flexible solution than, for instance, LDAP authentication, since it puts minimal requirements on the booking infrastructure.

After a successful login, the user is presented with different options depending on which user class he/she belongs to: administrator, teacher or student.

2) Occasions

To manage different usages and user groups the concept of occasions is used. Like an ordinary campus course, occasions have a start and end date. The occasion also describes limits on the maximum number of concurrent machines that can be used by its students and how many students that are allowed. The occasion will also describe which of the clone images the students can use.

The administrators are responsible for creating and managing the occasions. They also appoint one or more teachers, which become the responsible for the occasion. It is then up to the responsible teacher to do further administration, such as adding student users or deciding how many machines and for how long, the students can reserve the machines. (Up to the limit set by the administrator).

3) Pre-reservations

Teachers can create pre-reservations, which is a way of allocating machines in advance, e.g. for teacher-led experiments. The students can then sign up to participate, and a reservation is created from the pre-reserved machines, in the same way as an ordinary reservation. The teacher also has the option of doing forced reservations, where reservations are created automatically for a group of students. This can be useful when teaching inexperienced users (whom you can not rely on making reservations of their own).

4) Long term reservations

To provide a way of doing longer experiments (like simulations), the teacher also has the option of making long-term reservations, which can last longer than the ordinary limits.

5) Reservations

Students can make experiment reservations, by picking a start time and for how long he/she wants to experiment. He/she can then pick how many machines to use, and what to be installed on them, choosing from the list of allowed clone images. Because of the time it takes to install a clone image and to wipe the hard drive, an extra hour before and after the experiment is allocated.

6) Experiments

When the time of the reservation has come, the student must first choose to start the experiment from the web interface, before being granted access. This will lock access to the experiment environment to the specific IP of the machine of the student. If the student's IP changes during the experiment, this lock can be updated (See F for security implications).

While the experiment lasts, the student can view all the needed information about the experiment, e.g. how to access the machines and what is currently installed. He/she also has the option to reboot or reinstall a machine with a new clone image.

7) Maintenance

Because of the risk of machine failure, there has to be some way of disabling the machines from use. An administrator can do this by placing the machine in maintenance mode. This means the machine will not be considered as a candidate when allocating machines for new experiments. The administrator also has the ability to supervise the machines and view their current state, e.g., which machines are in use and by whom.

8) Implementation details

The web interface is implemented in PHP, with MySQL as a database back-end. To reduce the html clutter, Smarty [2] templates are used. The source code for the web interface is open source and available [3] under a BSD-like license.

B. *Laboratory equipment*

The lab consists of 32 reservable machines. Keeping these machines homogenous is not a requirement, but it helps with manageability. The homogeneity also provides redundancy, so even if one machine fails, the lab can remain operational, but with reduced capacity.

The automation of the machine preparation and cleaning requires a way of reliably rebooting the machines; therefore managed power distribution units (PDUs) are used. This means the power to the machine can be controlled and therefore the machines can be forced to reboot, without being dependent on operating system integration or solutions by a specific computer manufacturer.

All client machines are connected to a managed gigabit switch with VLAN (IEEE 802.1Q [4]) support, which can be dynamically reprogrammed. This makes it possible to create a VLAN dedicated to each experiment. In that way we can isolate the experiments from each other, though connected to the same physical network switch. To the experiments it will still appear as they are connected to a simple switched LAN.

C. *The EXP experiment system*

To create and manage the experiments, the EXP experiment system [5] is used. It is a package containing both proprietary software and modified open source software. Its basic purpose is to provide the services and tools needed to create an experiment environment in a scriptable and reproducible way.

1) Controller

At the heart of the EXP system lies the controller, a server with the responsibility to run and configure all the settings and services needed to create a experiment environment, e.g. DHCP, DNS, TFTP and FTP. The controller is a FreeBSD system, with EXP installed on top. It hooks into the operating system startup procedure, where it has the ability to set up network interfaces and generate configuration for the basic services before they start, all depending on the configuration of the EXP system. All configurations are stored in a centralized configuration database on the controller, called ConfUSE.

2) ConfUSE

ConfUSE is a flexible settings database, containing a hierarchy of classes, also called roles, each having a number of settings. The value of these settings can either be plain text, binary or generated from LUA [6] scripts. The hierarchy makes it possible to overload settings from parent roles (Fig. 4), adding the potential of sharing common settings from parent roles. The roles are mapped onto the IP address of the machine making requests to the database; this way a machine is said to have a specific role.

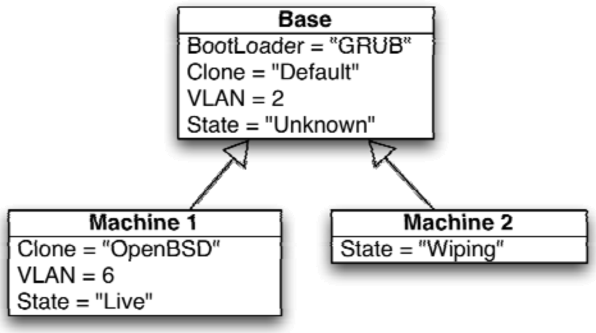


Figure 4. Inheritance between roles

3) *Booting*

To automate the machine cloning process, machines are booted over the network [7]. The boot loader will be loaded over the network from a TFTP-server. As the TFTP-server can read settings from ConfUSE, a suitable boot loader can be provided, depending on the machines role. This makes it possible to control how and what to boot on each machine. For instance, it can boot directly from its hard drive or boot into Cloning mode.

4) *Cloning and wiping*

A minimal FreeBSD kernel with a small memory file system, containing the cloning tools, is used for the cloning process. The cloning tool is controllable from ConfUSE, which makes it possible to automatically install or erase the hard drive.

Clone images are raw copies of the contents of the hard drive, stored in a single file. To reduce their sizes, either Run Length Encoding (RLE) or bzip2 compression can be used. Most experiments will not need an image of the whole disk. For instance, a Windows XP image can be installed on a 10GB partition and the cloning tool can take advantage of this by only copying the first 10GB of the disk drive. The whole disk will still be available to the experiment and most operating systems can grow the file system [8][9].

Clones are stored on the FTP-server and are compressed/decompressed on the fly when needed. The FTP-server can be controlled by ConfUSE, which makes it possible to limit access to clones depending on the role of the machine.

The cloning tool can also be used to overwrite the data on the hard drive. Different patterns are available, e.g. writing zeros, random data or by using the Gutmann algorithm [10]. In the remote lab we only overwrite with zeroes. Using the Gutmann algorithm would be overkill since no user has physical access to remove and analyze the hard drive.

D. *Time controlled events*

The experiments lifetime is controlled by a time scheduled (cron [11]) script. One hour before the experiment reservation begins resources for the experiment are allocated (Fig. 5). A VLAN is assigned and free machines are allocated for the experiment. The VLAN assigned to the experiment is not going to be used until the machine is going into the “live” state. Until then, the setting is stored in ConfUSE. The cloning process is then started for each machine, with the clone images previously selected in the reservation.

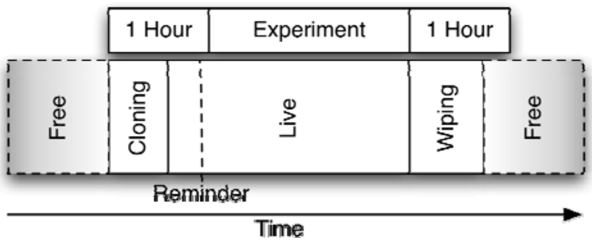


Figure 5. Machine lifetime

Five minutes before the experiment begins, an email is sent to the user who made the reservation, reminding of the upcoming experiment. This email has all the information about the machine setup, what is installed on each machine and which IP addresses that should be used to access the machines.

After the experiment is over, a hard disk wiping operation is initiated.

If a user chooses to reinstall a machine from the web interface, the same cloning operation is used as for the time controlled cloning.

E. *Machine lifetime transitions*

When a machine is cloned or wiped, it is placed on a dedicated VLAN where no “live” machine can ever reside. This is a security measure to ensure that nobody is able to tamper with the cloning process. All transitions between machine states and VLANs must be delicately handled and therefore machines are always powered off during the transitions (Fig. 6).

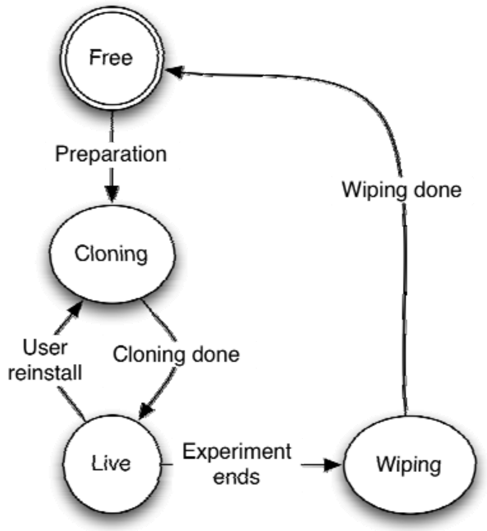


Figure 6. Machine states and transitions

When a cloning or wiping operation is initiated, the machine is booted into the cloning tool. By fetching an automation script from ConfUSE, the machine is instructed to do the steps needed. First, a callback is sent to the system to indicate that the boot process was successful and it will begin the operation, then the cloning/wiping itself is carried out. After that is done, another callback is sent to the system indicating the operation was successful. This callback can trigger new state transitions. After a clone is successfully written, the machine will automatically be changed to “live” mode and will then be booted from the newly written contents of the hard drive. In “live” mode, it will also be placed on the VLAN assigned earlier. A wiped machine will be powered off and marked as “free”, now available for new experiments.

F. *Firewall control*

When a student chooses to begin an experiment, the IP address for his/her machine is saved. The address can then be used to generate the needed firewall configuration to only allow access from the users machine to the machines of the experiment. The firewall software used, OpenBSD

Packet Filter [12], can do stateful inspection, which means responses are allowed if the connection was initiated from the user's machine. This makes it possible for the user to use whatever protocol he/she wants to access the experiment. This is a freedom under responsibility; while no connections can be initiated from inside the laboratory, you have to keep in mind that the return data on a user-initiated session can also contain harmful information. In any case, the only external machine affected would be the machine from which the user is logged in. A firewall filtering policy for incoming protocols could be used, but it would limit the experimentation possibilities.

Because of the possibility that the user's IP address changes during an experiment session, the system is also capable of updating the IP-lock address. The stateful inspection saves initiated connections in a table, and before allowing the new address to be used this table must be cleared of any connection-states belonging to the old address.

IV. DISCUSSION

A. *Virtualization*

It is common that other laboratories use virtualization as a way to create isolated and controlled environments. While it has some advantages, it also leads to limitations in the kind of experiments that can be performed.

Virtualization platforms can utilize the hardware resources better and will most certainly be cheaper to set up and run, depending on what hardware and software platform that is used. The startup of a Virtual Machine (VM) is faster that doing physical writing to a hard drive, so the startup time of an experiment will be shorter. After the experiment is over, the VMs disk content can be discarded, which means you do not need to wipe the complete hard drive. It will also be easier to prepare installations, as the teacher can do that at his/her own machine in the virtualized environment, without the worry for hardware incompatibility.

On the downside, using virtual machines can lead to unwanted behavior. If many experiments are going to be run on the same machine, they have to share the hardware resources between each other, which means that they also influence each other. Experiments which are resource intensive will take longer to perform and results from performance measurements will not be reliable. It can also influence results from experiments depending on concurrency, one example from the security domain is race conditions (See [13]). Running several virtual environments on the same machine will also lead to penalties on hard drive access.

In a virtualized environment the hardware is emulated and therefore will not allow direct hardware access. This will introduce new bugs related to the emulation and will probably hide bugs in the real hardware. Experiments that require direct hardware access will therefore be impossible, as would experiments depending on hardware bugs.

For many experiments, these negative aspects may not be a problem and virtualization can still serve a good purpose as a remote experiment solution. If needed, the presented laboratory solution can run a virtualized environment. In this case the machine will be prepared with the virtualization software and the teacher or the students can themselves provide the virtual machine image and take advantage of the simpler preparation procedure given by the virtualized environment.

Another interesting aspect of virtualization is security. The main contributor to software vulnerabilities are complexity and by introducing a virtualization layer the complexity is increased even more. As no software is perfect, there may be ways for users to break out of the virtualized environment [14]. This would lead to undefined behavior of the whole laboratory.

B. *Wear and tear*

Using PDUs to force the machines to reboot puts some extra strain on the machines, as the power is abruptly cut without any prior notification.

Older hard drives had problems with contact start-stop (CSS) cycles, which happens when you have a power failure, or as in our case, cut the power to the hard drive. Historically this decreased the lifespan of the hard drive. Modern hard drives use different techniques to avoid this problem. One solution is CSS landing zones, which are specially prepared regions on the platter, to where the read head return automatically when the power is cut. Other solutions avoid the problem altogether by moving the read head away from the platter [15]. Most hard drives guarantee a start-stop cycle count of 50000 [16][17] or nearly 3 years (1000 days) of operation at 50 cycles per day.

V. OTHER USES

The aspects important to remote security experiments can be important in other contexts as well. For instance, when setting up a remote controllable machine, there is always a chance that it will be tampered with. Machine cloning (and containment) can help here, as the user will always be presented with a clean and controlled environment. The antenna laboratory [18] at BTH has already adopted the technology, where students need remote control over proprietary antenna experiment hardware and software. Similar to that, the laboratory could be used to enable student access to expensive and licensed software (if the end user license agreement allows it), without needing to install it on their home computers. A variation of this has been used when having off-campus courses. When visiting other schools or companies, you cannot always install what you need on their computers. Instead, a clone image can be prepared in advance, containing what the course requires. The laboratory can then be used as a course platform, only requiring the remote login software to be available on the off-campus machines.

VI. RELATED WORK

This paper focuses on the construction of the laboratory and not its specific usages. There are many areas where this work could be of help, not only in security-centered laboratories, but also in any form of e-learning solution requiring specific software or experiment environments.

Other dedicated security laboratories exist around the world [19]. Some even with remote access [20], but most, unlike us, uses virtualized environments.

Emulab [21], at the University of Utah, focuses on network emulation and uses similar techniques, e.g. machine cloning and scripted control. Their work focuses more on research and simulation, while our aim is to be more pragmatic and easy for students to use. Their clone utility, Frisbee [22], uses multicast and informed file system compression techniques to cut down the clone time and network congestion. The scalability factors of the remote security laboratory have not yet been studied in-depth, but the Frisbee solution could be worth investigating when and if scalability problems arise.

VII. CONCLUSIONS AND FUTURE WORK

This paper has shown that it is possible to create a remote controlled security experiment environment. By controlling firewalls and switches, the experiments network environment can be isolated from the outside world and from each other, but can still allow remote control. A cloning system is used to achieve reproducible experiments, where a machine is restored to a given state. To ensure no data leakage between experiments run on the same machine, machines are completely wiped from all experiments data after the experiment is over.

An informal inquiry has shown that the students are pleased with the possibilities to be able to do experiments from their home instead of going to the university when experiments cannot be run on their own machines. They also appreciate the possibility to be able to reinstall machines, if they make a mistake while experimenting. Teachers say that the laboratory is very useful for some kinds of experiments and can save a lot of time and at the same time give students more time to experiment freely. The laboratory is still in its infancy and no research has been made on how the lab is perceived as a didactic tool in the education.

There are many ideas of how to refine the laboratory and how to improve the usage and usability of it and there are still many areas of improvement. As noted above, scalability has not yet been fully explored. Right now, the cloning times are masked from the user by the time-controlled cloning (the machine preparation is done without the user's knowledge). That will change if "on demand experiments" are introduced, where an experiment can be started as soon as possible after the machines have been prepared.

Better means of hardware diagnostics should be explored; both health tests and runtime status reports could be very helpful for administrators and developers. Other experiment related probes would be interesting, giving students and teachers the ability to trace and log experiment events.

Ways of giving users even more control over the computer hardware should be explored. For example, a remote controllable Keyboard/Video/Mouse-switch (KVM) could be used to give direct display output. Other means of control could also be considered, as for example serial terminals, either directly connected or by using IPMI [23] and serial over LAN. This will of course introduce new problems, e.g. tampering with the BIOS. Under any circumstances, no matter how much a student has tampered with the hardware, the integrity of the laboratory has to be guaranteed.

REFERENCES

[1] Johan Wieslander, "Case Study of a Security Laboratory", Advanced Topic in Computer and Information Science, "unpublished", 2004.

[2] Smarty, http://www.smarty.net, 2008-05-28.

[3] Openlabs development page, http://svn.openlabs.bth.se/trac/openlabsweb, 2008-05-28.

[4] 802.1Q - "IEEE Standard for Local and metropolitan area networks. Virtual Bridged Local Area Networks", 2006.

[5] Per Mellstrand, "Informed System Protection", 2007, pp. 163-177.

[6] Roberto Ierusalimschy, Luiz Henrique de Figueiredo, Waldemar Celes, "Lua 5.1 Reference Manual", 2006.

[7] Intel Corporation, "Preboot Execution Environment - (PXE) Specification", Version 2.1, 1999.

[8] FreeBSD manual page for growfs(8).

[9] Microsoft Corporation, "Automating and Customizing Installations", Updated 2003.

[10] Peter Gutmann, "Secure Deletion of Data from Magnetic and Solid-State Memory", 1996.

[11] The Open Group Base Specifications Issue 6, IEEE Std 1003.1, 2004 Edition.

[12] Jeremy C. Reed, "The OpenBSD PF Packet filter book", August 2006.

[13] John Viega, Gary McGraw, "Building Secure Software – How to Avoid Security Problems the Right Way", September 2001.

[14] Common Vulnerabilities and Exposures, CVE-2008-0923, Published 2008-02-25.

[15] Patricia Kim and Mike Suk, "Ramp Load/Unload Technology in Hard Disk Drives" whitepaper, 2007.

[16] Maxtor Corporation, "Maxtor DiamondMax 10" specification, 2006.

[17] Hitachi Global Storage Technologies, "Hitachi Deskstar 7K1000" specification, 2007.

[18] http://antenna.openlabs.bth.se, 2008-05-28.

[19] Brian Hay, Kara L. Nance, "Evolution of the ASSERT Computer Security Lab", 2006.

[20] Chistian Willems, "Tele-Lab IT-Security: an architecture for an online virtual IT security lab", 2008.

[21] Brian White, Jay Lepreau, Leigh Stoller, Robert Ricci, Shashi Guruprasad Mac Newbold, Mike Hibler, Chad Barb, Abhijeet Joglekar, "An Integrated Experimental Environment for Distributed Systems and Networks", October 2002.

[22] Mike Hibler, Leigh Stroller, Jay Lepreau, Robert Ricci, Chad Barb, "Fast, Scalable Disk Imaging with Frisbee", April 2003.

[23] Intel Corporation, Hewlett-Packard Company, NEC Corporation, Dell Inc., "Intelligent Platform Management, Interface Specification - Second Generation v2.0", 2006.

AUTHORS

J. Zackrisson is with the Department of Signal Processing / School of engineering / Blekinge Institute of Technology, PO Box 520, SE-372 25 Ronneby, Sweden. (e-mail: johan.zackrisson@bth.se).

C. Svahnberg, is with the Department of Software Engineering / School of engineering / Blekinge Institute of Technology, PO Box 520, SE-372 25 Ronneby, Sweden. (e-mail: charlie.svahnberg@bth.se).

This article was modified from a presentation at the REV2008 conference in Düsseldorf, Germany, June 2008. Manuscript received 27 June 2008. Published as submitted by the authors.