A Security Laboratory for CTF Scenarios and Teaching IDS

Sebastian Roschke, Christian Willems, and Christoph Meinel
Hasso Plattner Institute (HPI), University of Potsdam
P.O.Box 900460, 14440, Potsdam, Germany
{sebastian.roschke, christian.willems, meinel}@hpi.uni-potsdam.de

***Abstract*—Modern attacks are using sophisticated and innovative techniques. Teaching and training programs have to focus on the practical aspects of security, i.e., on attack techniques and defense methods, such as Intrusion Detection Systems (IDS). Team exercises provide an effective method to increase practical experience in security, as each team can gather knowledge on offensive as well as defensive techniques while learning from each other. The proposed laboratory provides effective security training by offering practical scenarios to practice attack and defense. The scenarios are parameterized and configured automatically to improve manageability. By using virtualized components as well as dedicated network infrastructure components, we can create complex scenarios that reflect real network environments. We designed and implemented several scenarios with various levels of difficulty. Finally, the teaching experiences for students and teachers are described based on the execution of the scenarios in multiple training sessions.**

I. Introduction

Teaching practical IT security is a major part of IT security education. To know how to secure a computer system, it is vital to have an impression of how attackers work and think, and what tools they actually use [2] [6]. A training of defensive techniques only will not allow the students to compete with the innovativeness of attackers, which keep developing new techniques and tools efficiently. Attack techniques become more sophisticated and valuable, as defenders do not know what tools are available and how they are used to perform attacks. Convenient frameworks (e.g., Metasploit [8]) provide possibilities for unexperienced people to attack networks and hosts. Education and awareness are very important tools to face the dangerous situation.

Intrusion Detection and Prevention Systems (IDS/IPS) have been widely used in practice for identifying malicious behaviors in network environments. An effective IDS should be capable to detect various types of attacks, its variants, and possible evasion techniques. IDS can be classified based on the protected objective and according to the detection model. There are two major detection models: anomaly based and signature based detection. Objective-based classification divides into host-based intrusion detection systems (HIDS) and network-based intrusion detection systems (NIDS) [1]. A major problem with IDS is the deployment and configuration complexity. To use an IDS efficiently, an administrator needs to deploy it at critical points within the network, which are not always easy to find. Additionally, the IDS needs to be configured correctly to prevent the generation of huge amounts of alerts that are not caused by real attacks, but by random adverse network traffic.

Especially for effective intrusion detection, the knowledge on the attacker's methods and tools is essential. To teach practical security and IDS, a laboratory is developed and presented in this paper. It consists of a movable server rack with dedicated networking components as well as multiple servers running complex virtual machine (VM) based networks. Furthermore, we designed multiple scenarios that can be parameterized and customized easily. Each scenario provides multiple ways for attack and defense, e.g., by different IDS deployments and configurations. The defense methods can be tested in an environment with real attack traffic and real attackers (impersonated by students) that try to circumvent the security measures. Finally, we describe some important lessons learned during the execution of the scenarios in multiple training sessions.

The rest of the paper is organized as follows. Section II introduces approaches for tele teaching in general as well as for IDS/IPS. Section III describes the proposed laboratory and the implemented scenarios. Furthermore, our approach to IDS teaching is described. Section IV shows the major experiences we made by teaching security with the scenarios and the lab. Section V gives a short summary of our contributions and a brief outlook on the future work.

II. Related Works

In [6], a set of scenarios and a testbed is described to provide practical security training. The testbed network used dedicated hosts to provide targets and attacking systems. Multiple iterations of scenarios are described, distinguished in *Treasure Hunt (TH)* scenarios and *Capture The Flag (CTF)*. In a TH scenario, the students are working as attackers to find hidden treasures in the network, e.g., certain files, passwords, etc. In a CTF scenario, the students are separated in two teams to perform attack and defense of a network. As no virtualization is used to implement the scenario, the management is very complex and difficult to handle. Furthermore, the scenarios are recreated for every session which requires a huge effort. The teaching approach is appreciated by the students and improves their knowledge on security in general. We share the opinion that high quality security teaching needs to involve practical exercises. We use the basic idea of TH and CTF scenarios and provide parameterized scenarios that are easy to reuse

Sebastian Roschke, Christian Willems, Christoph Meinel:
"A Security Laboratory for CTF Scenarios and Teaching IDS"
in Proceedings of the 2nd IEEE International Conference on in: Education Technology and Computer (ICETC 2010),
IEEE Press, Shanghai, China, pp. 433-437, 6, 2010. ISBN: 978-1-4244-6367-1.

for multiple sessions. Those can even be modified slightly to increase or decrease the difficulty of the scenario. Furthermore, we use virtualization to improve the managability of the scenarios.

In [5] and [4], approaches are described to teach practical intrusion detection. [4] focuses more on manual detection by analyzing log files and attacked systems, which are simulated. Therefore, it is more related to forensics. [5] focuses on performing IDS/IPS trainings remotely. The network is based on dedicated hosts connected by a HUB, i.e., each network host can see each network traffic. The attack traffic is simulated by configured hosts running predefined attacks automatically. This approach is useful to train the configuration of a single IDS sensor and the configuration of the connection to an IDS management system. The major problem is that it does not reflect a realistic network. This approach is not capable of teaching best practices on different IDS deployments within a realistic network. By using virtualized networking in combination with dedicated network infrastructure devices, we can create huge complex networks that reflect realistic environments. Furthermore, our solution is easier to manage and to customize.

Tele-Lab IT-Security [3] was designed to offer hands-on experience exercises in IT security without the need of additional hardware or maintenance expenses. The existing implementation of *Tele-Lab* provides access to the learning environment over the Internet. The approach uses virtualization to simplify the management of scenarios. Although *Tele-Lab* is capable of creating complex networks, it does not offer a way to run CTF scenarios. Furthermore, it is not capable of operating dedicated network infrastructure components, such as real switches or routers. As related project, *Tele-Lab* can offer the scenarios described in this paper in the future.

III. A Flexible Security Laboratory

A. Architecture

The laboratory uses real hardware components as well as virtualized components. It is a movable server rack containing several servers and network components. We use a *Foundry FastIron Edge X624* as main backbone switch and router. The main backbone is capable of routing *IPv6* and *IPv4*, which provides us the possibility to use *IPv6* based networks in the scenarios. Besides the second backbone switch (*HP ProCurve 2810*), we use multiple *Netgear FS108* switches for networking. The WiFi connections are provided by two *Linksys WRT54GL* access points, which are working as DHCP servers as well. As servers, a *Dell PowerEdge* and a *Fujitsu Siemens Workstation* are used. Both systems contain $4GB$ main memory and $500GB$ hard disk space. The servers are running *Debian Linux 5* and *VMWare Server 1*. Both systems have multiple network interfaces. One is used for the management of the system and the others are used for the networking in the scenario. We use about 25 different VMs for the scenarios. We have one dedicated firewall and multiple VM-based firewalls which are based on *Debian Linux* and *iptables*.

Figure 1 shows the software architecture of the integrated components. The scenario is implemented by the VMs and the scenario network. Both are created when a scenario is loaded. The VMs are created by the *VM Creator*. To configure the networking on the VMM server, we use the *Network Creator*. To configure the parameterized scenarios, each VM has a small *Bootstrapping Service* (BS) running, which is used to configure it when the scenario is loaded. This service is shutting down after configuration of the VM. The BS is communicating with the *VM Configurator*. The simple management system consists of the *VM Creator*, the *Network Creator*, and the *VM Configurator*. The management system controls the scenarios and can be configured and operated by the user.

Scenario	Objectives	Type
1	Reconnaissance and Password Attacks	TH
2	Wifi and Remote Exploitation	TH
3	Attacking Firewalled Networks	CTF
4	Attacking Web Applications and Deployments	CTF

TABLE I
Scenario Overview

B. Scenarios

The laboratory works with general scenarios to teach practical security. A general scenario consists of the following properties:

- Objectives
- Tasks and Phases
- Architecture
- Software and Hardware
- Parameters
- Type

The *Objectives* specify the topics of this scenario, e.g., *Reconnaissance*, *Attacks on Passwords*, *Remote Exploitation*, etc. The *Tasks* specify the things the students are supposed to do to solve the scenario successful. Tasks can be grouped in *Phases*, while several *Phases* make up a whole scenario. The *Architecture* describes the involved hosts, services, connecting networks, and network infrastructure of the scenarios. A detailed description of the architecture is provided by specifying *Software* and *Hardware*. The *Parameters* define the modifiable parts of the scenarios, e.g., passwords of the users, IP addresses of the hosts, etc. The *Type* defines the general type of a scenario, e.g., a CTF scenario, a TH scenario, etc. Table I shows the scenarios described in this paper.

Scenario 1. The first scenario shows the first stages of an attack and possible countermeasures. It covers reconnaissance as well as simple attacks on accounts and passwords. The scenario consists of three phases. The first phase covers the reconnaissance with the basic task to gather as much information on the network as possible. This includes finding the target machines and running services. Additional information, such as version numbers, service banners, OS and service fingerprints, MAC addresses, and network traffic, might be very handy to solve the task. The second phase covers getting access to the systems. In this phase, the students have to choose the target machines and to guess account names and passwords. In this scenario, there are several accounts with

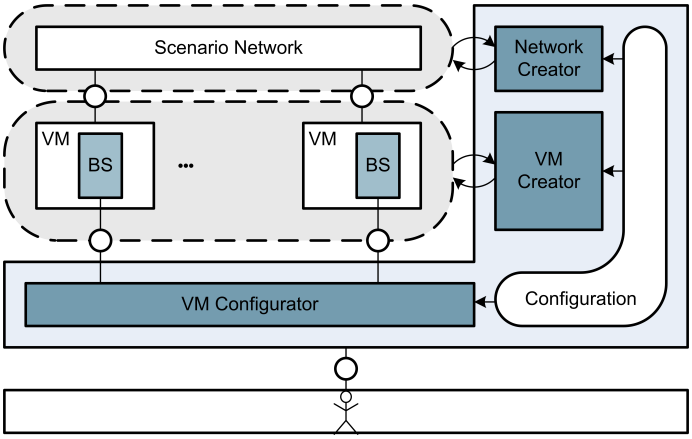

Fig. 1. CTF Laboratory Design

weak passwords that can easily be guessed. Apart from the standard accounts, such as *root* or *administrator*, there are a few simple accounts leaked by service banners (e.g. an FTP [10] server with a banner *Bob's FTP server*). The third phase covers password cracking after accessing the systems. In this phase, the students need to use common word lists to crack the passwords from the password storage on the accessed systems (e.g. *shadow* file or *NTLM hashes*). Furthermore, the students need to modify the wordlists (e.g. with permutation or adding of numbers) to find all remaining passwords in the list.

The scenario consists of five hosts within one subnet. Three of the hosts are running a *Linux* based OS and two are running *Windows*. There are remote access services enabled on the machines (e.g. *ssh* on Linux and *telnet* on Windows). Furthermore, there are multiple services running on each machine, such as an FTP [10] service, an HTTP [11] service, an SMTP [12] service, an SMB [9] service, etc. Each service provides several information on version number, service type, and possible users. There are no additional requirements concerning the services in this scenario, i.e., the current stable and fully patched version can be used. The passwords configured need to be weak, i.e., they can be guessed easily or they can be found in a common word list. The scenario can be parameterized by changing the services running on each host and the information being leaked. Furthermore, the account names can be changed as well as the corresponding passwords. The scenario is a TH scenario, as the whole group of students needs to find certain treasures.

Scenario 2. The second scenario shows a more sophisticated attack and possible countermeasures. It covers WiFi attacks as well as remote exploitation techniques using attack frameworks. This scenario consists of four phases. In the first phase, the students need to get access to the network by using WiFi attacks, e.g., WEP Replay attacks or WPA dictionary attacks. In this phase, the students could work together to perform a WEP attack, as only one student needs to do packet injection into the encrypted WiFi connection. All other students only need to sniff WiFi traffic to attack successfully. After getting access, the students need to examine the network again in the second phase. The basic task is to gather as much information as possible on the target network and hosts. This time, version numbers of services and OS versions are critical to know. In the thrid phase, the students need to get access to the hosts by actively exploiting the services running on the hosts. This can be done by using exploit frameworks, such as *Metasploit* [8]. As the services are vulnerable, the students can get access to the hosts. In the final phase, the students need to find hidden secrets on the host, which can be combined to break a simple encryption.

The scenario consists of three hosts in one subnet. Two of the hosts are running a Linux based OS and one is running Windows. One of the Linux hosts is running an SMB service with version *3.0.23* that is vulnerable to the *LSA Heap Overflow* described in CVE-2007-2446 [7]. The other Linux host is running a simple service that is vulnerable to a buffer overflow. This service does not follow any specified network protocol, but does only receive a string and writes this string to a log file. This service needs to be exploited manually, as no public exploit is available. The Windows based machine is running an SMB service and is based on *Windows XP SP3* which is vulnerable to a stack corruption as described in CVE-2008-4250 [7]. The services need to be exploitable which requires a specific OS version and type. The riddle can be solved by finding every hidden secret on the machines. The secrets are hidden using OS specific methods, such as multiple file streams within one NTFS file on Windows. The secrets build up a cipher text which needs to be based on a weak encryption scheme, such as a Cesar cipher. The scenario can be parameterized by using different vulnerabilities of different services. The hiding mechanisms of the secrets can be changed as well as the secret itself. This scenario is a TH scenario, as the students need to find several treasures.

Scenario 3. The third scenario shows a sophisticated attack through a firewall using a WiFi and possible countermeasures. It covers remote exploitation techniques using attack frameworks and penetration of security measures, such as firewalls and WiFi encryption. This scenario consists of 4 phases. In the first phase, the student have to get access to the WiFi, which is encrypted by WPA (with a weak password) or WEP. In the second phase, the students have to gather as much

information as possible on the network and on the firewall. The third phase covers the first exploitation through the firewall. The first task is to successfully exploit one host behind the firewall and the second task is to enable this host to perform further attacks inside the network. Both tasks are tricky to solve, as exploitation through a firewall needs connect-back shellcode and enabling the host to perform attacks needs a several tools to be installed on the compromised host. Again, exploitation can be done using *Metasploit*. In the fourth phase, the students need to compromise the rest of the network, which can also be done using attack frameworks or by gathering useful information on one host, such as passwords, private keys, etc.

The scenario consists of five hosts behind a NAT firewall, one WiFi access point running a DHCP server. The access point is provided by the *WRT54g* appliance. The firewall is connecting two different subnets, the attacker subnet and the target subnet. Three of the hosts are running a *Linux* based OS and two are running *Windows*. The Linux hosts have remote access services enabled (e.g. *ssh*). Furthermore, there are multiple services running on each machine, such as an FTP [10] service, an HTTP [11] service, an SMTP [12] service, an SMB [9] service, etc. Some services are accessible through the firewall. However, there is only one accessible service that is vulnerable. Other vulnerable services are only accessible inside the target network and can be exploited in the last phase. The firewall is represented by a dedicated server and the hosts are virtualized. In this scenario, we are also using an SMB service with version *3.0.23* that is vulnerable to the *LSA Heap Overflow* described in CVE-2007-2446 [7]. Furthermore, we use the Windows SMB and the IIS[13] FTP server services, which are vulnerable to a stack corruption as described in CVE-2008-4250 [7] and to the *Microsoft IIS FTP Server NLST Response Overflow* described in CVE-2009-3023 [7]. There are a lot of vulnerable services that could be used instead. The scenario can be parameterized by changing the firewall rules and making different services accessible from the outside. Furthermore, services can be disabled and enabled to provide more different possibilities to gain access to the network. This scenario can be used as TH scenario or as CTF scenario, where two groups of students try to attack respectively defend the target network.

Scenario 4. The fourth scenario shows a complex attack on a realistic network environment, including a firewall, several hosts providing one complex web application. It covers remote exploitation techniques, reverse engineering, and penetration of security measures, such as firewalls. The scenario consists of four phases. In the first phase, the students have to analyze the network and the connected hosts. Furthermore, they need to find running web applications and services. In the second phase, the web applications need to be reverse engineered to find vulnerabilities during the attack. The third phase covers the exploitation of the vulnerabilities to get get access to one of the hosts. This needs to be done manually, as no public exploits are available for the provided web application. In the last phase, the students need to compromise the remaining hosts in the network by reverse engineering the application and analyzing the compromised host. This exploitation is also done manually based on the web application and the involved hosts.

The scenario consists of three hosts in two subnets. The attacker and the target subnet are connected by the proxy host. All hosts are running Linux and have no vulnerable binary applications running. One host works as frontend host and is running an *nginx* web proxy providing several local and remote web applications. One web application is located on the backend server and is running on Apache. This self-made application has several exploitable vulnerabilities. The two backend hosts have restrictive firewall configurations allowing communication with the frontend host only. The vulnerable web application is based on PHP and is vulnerable to Cookie Forgery, SQL-Injection, and PHP-Code-Injection. All of those vulnerabilities need to be exploited to compromise the host. The two remaining hosts can only be compromised by accessing them from the first compromised host, e.g., by using available private keys for remote access. All the involved hosts are running on virtual machines. This scenario can be parameterized by configuring the vulnerabilities in the web application. The implemented application provides possibilities to increase the difficulty of successful exploitation, e.g., by encoding the cookies that need to be forged, or by using filtering techniques against SQL injection, leaving only a few possibilities to exploit successfully. Furthermore, web applications can be disabled and enabled to increase obfuscation. We also provide several common web application that are vulnerable, e.g., *phpmyadmin*. This scenario can be used as TH scenario or as CTF scenario as well, as it is suitable for two student groups to practice attack and defense.

C. Analyzing IDS Deployments

A major task for the defending team in the CTF scenarios (i.e. Scenario 3 and 4) is to secure the scenario by deploying and configuring IDS and IPS solutions. We provide the possibility to deploy network-based sensors, such as Snort [15], as well as host-based sensors, such as Tripwire [17] and Samhain [14]. Basically, the utilization of anomaly-based scanners (such as Bro [16]) is not applicable, as we have no chance to create a training data set and suitable thresholds for the scenarios. In the scenario networks, there is only attack traffic, which can not be used as training data. Therefore, we concentrate on signature-based sensors. There are two problems to be solved for deploying IDS sensors in the scenarios: finding good locations for the sensors as well as configuring the sensors in a suitable way. Finally, the topic of intrusion response is handled in the scenarios as well.

Possible IDS deployments of Scenario 3 are shown in Figure 2. The deployment of the IDS in the network of the Access Point (AP) and the firewall (FW) allows only to detect attacks. To execute countermeasures, the IDS would have to communicate with the AP, which is not possible in the specific scenario, as the AP has no interface for such purpose. Deploying the IDS on the FW would enable active countermeasures, e.g., by dropping malicious packets. Using an IDS in the internal network enables detection of attacks to one of the internal hosts $h_1 - h_n$. To implement

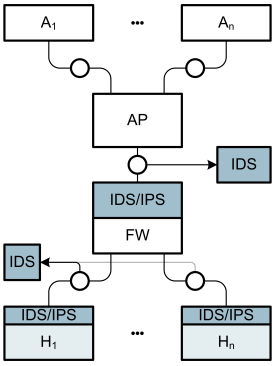

Fig. 2. Scenario 3: IDS Deployment

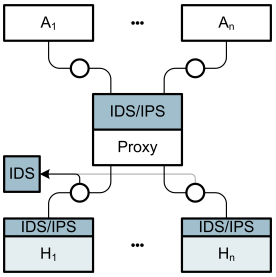

Fig. 3. Scenario 4: IDS Deployment

countermeasures, the students need to setup firewall scripts on the internal hosts for blocking the attacker. Possible IDS deployments of Scenario 4 are shown in Figure 3. The usual deployment of IDS in this scenario is on the proxy server, as one can monitor all incoming traffic. On the proxy server, the detection of attacks as well as instant countermeasures are possible. A deployment in the attackers network is not suitable, as the defending students have no control over this network. As described in Scenario 3, the IDS could also be deployed in the target network as well as on the different target hosts, while a deployment on the target hosts enables countermeasures.

In both scenarios, we work with signature-based IDS only, as no training data is available, which would be needed for anomaly-based IDS sensors, such as Bro [16]. We allow host-based as well as network-based IDS sensors to be deployed in the scenarios. The network-based sensors detect the network based attacks as they create malicious traffic matching specific signatures. Furthermore, suspicious traffic, such as clear text shell commands that are used to control a system remotely, can also be detected. Generic signatures, such as *Shellcode Detection Rules* can also be used to detect a broad class of attacks in the scenarios. Host-based sensors detect malicious behavior on the specific hosts, such as the execution of unknown code, the reading or modification of critical files, or the establishing of connections as well as the creation of unknown listening sockets. The used rules for host-based IDS are policy based and need to be created by the students. The correct setup and creation of suitable rules for the scenarios is a difficult task for the students. But by solving this task, it is much easier to defend the scenario at runtime, as several automations can block attackers at runtime.

IV. Teaching Experience

We used the scenarios in our lectures on practical network security for two years. The lab can handle up to 30 students per session. The first two scenarios are usually finished in 90 minutes, while Scenario 3 and 4 last up to 4 hours in the teaching sessions. The record for Scenario 1 is at 35 minutes and for Scenario 2 at 47 minutes. The Scenario 1 can be finished by approximately 95 percent of the students. Scenario 2 can be finished by approximately 84 percent of the students. The Scenarios 3 and 4 have been used as CTF scenarios, i.e., two student teams compete with each other. For both cases, only approximately 50 percent of the scenario has been solved by the students. The session was aborted after 4 hours. The students consider the practical exercises with the lab as very useful and important. Especially the possibility to touch a real system and do real networking with hardware seems to impress the students.

In the CTF scenarios, the balance of the two teams is one of the major challenges. Basically, the network defenders can inherently prepare themselves better, as they have the chance to find out about all vulnerabilities of a scenario before. The attackers face multiple challenges at once: 1) they have to overcome the obfuscation techniques applied by the defenders, 2) they need to choose the right arms out of a huge repository (e.g., use the right exploit for many services), and 3) often they have to perform guessing-based attacks, i.e., the CTF scenarios are complex and it is not easy for the attacking team to find the right spot to attack. This might be the main reason why the CTF scenarios are usually not finished completely.

Building new scenarios needs a huge effort for the teachers. Thus, we are working with the parameterized ones. However, the testing of the scenarios, i.e., using all possible ways to attack the network, needs to be done beforehand to make sure that the scenario can be solved. Furthermore, the rules for the attackers and defenders need to be flexible through the session to keep the balance, e.g., if the attackers can not solve the first task, they might get a hint. Generally, such kind of sessions are not easy to control, as with multiple involved teams and individual students, the situation might become very complex. Nevertheless, such experiences are very valuable for the students and teachers in general.

V. Conclusion

For teaching practical security and IDS, a flexible laboratory is presented in this paper. It consists of a movable server rack with dedicated networking components as well as multiple servers running complex virtual machine (VM) based networks. Multiple scenarios are designed for being parameterized and customized easily. Each scenario provides multiple ways for attack and defense, e.g., by different IDS deployments and configurations. The defense methods can be tested in an environment with real attack traffic and real attackers (impersonated by students), who try to circumvent the security measures. Finally, we describe some important lessons learned during the execution of the scenarios in multiple training sessions, e.g., that students really appreciate such

exercises and that tutors can use and customize the designed scenarios easily.

The next step is to enable the execution of the complex scenarios remotely. We will do this by using the *Tele-Lab* [3] technology. Furthermore, more scenarios will be created based on real world examples. It should be possible to create a scenario network by simulating an existing network. By using this method, the students can easily attack and defend a real network without the danger of disturbing daily work or destroying critical components.

REFERENCES

[1] S. Northcutt: *Network Intrusion Detection - An Analyst's Handbook*, New Riders, June 1999.

[2] Bishop, M.: *Academia and Education in Information Security: Four Years Later*, In: Proceedings of 4th National Colloquium on Information System Security Education, Washington, DC, pp. 30-32, 2000.

[3] Willems, Ch., Meinel, Ch.: *Tele-Lab IT-Security: an Architecture for an online virtual IT Security Lab*, In: International Journal on Online Engineering (iJOE), vol. 4, issue 2, pp. 31-37, 2008.

[4] Rowe, N. C., Schiavo, S.: *An intelligent tutor for intrusion detection on computer systems*, In: Computers and Education, vol. 31, issue 4, pp. 395-404, 1998.

[5] Lahoud, H.A., Tang, X.: *Information Security Labs in IDS/IPS for Distance Education*, In: Proceedings of the 7th Conference on Information Technology Education (SIGITE'06), ACM Press, pp. 47-52, 2006.

[6] Vigna, G.: *Teaching Hands-On Network Security: Testbeds and Live Exercises*, In: Journal of Information Warfare, vol. 3, issue 2, pp. 825, 2003.

[7] Mitre Corporation Common vulnerabilities and exposures. CVE Website: http://cve.mitre.org/ (Accessed January 2010).

[8] The Metasploit Project, Website: http://www.metasploit.com/ (Accessed January 2010).

[9] SMB NetBIOS RFC 1001: *http://www.rfc-editor.org/rfc/rfc1001.txt*

[10] FTP RFC 959: *http://www.rfc-editor.org/rfc/rfc959.txt*

[11] HTTP RFC 2616: *http://www.rfc-editor.org/rfc/rfc2616.txt*

[12] SMTP RFC 821: *http://www.rfc-editor.org/rfc/rfc821.txt*

[13] Microsoft Internet Information Server (IIS): *http://www.iis.net/* (Sep 2009).

[14] Samhain IDS: *http://www.la-samhna.de/samhain/* (January 2010).

[15] Snort IDS: *http://www.snort.org/* (January 2010).

[16] Bro IDS Website: *http://www.bro-ids.org/* (January 2010).

[17] Tripwire: *http://www.tripwire.com/* (January 2010).