A Multi-user Adaptive Security Application for Educational Hacking

Emily Crawford, Yi Hu

Abstract— This paper illustrates a project that explores the new horizon of information security exercise paradigm through the construction of a research application that is purposefully designed for exploitation. Such an application provides students an opportunity to study computer security with an ethical, realistic, and comprehensive environment to carry out hands-on practices of attacking an application to better understand the motives and techniques of real hackers. Empirical comparison with existing security exercises was also carried out. Moreover, pedagogical and development issues related to such a security exercise were investigated and incorporated into the design considerations of this research application.

***Index Terms*—Computer Security, Hacking, Security Education, Web Application**

I. Introduction

As our dependency on technology continues to increase, so does our need for computer security. College courses focusing on computer security offer students insights on various techniques and strategies used by hackers to exploit computer systems and applications. However, many of those courses provide little practice in the application of computer security. An important aspect in learning about the various attacks and how to prevent them not only lies in the knowledge acquired from books and lectures, but also in practice. Students can gain a deeper understanding of how attacks work and how to better prevent them by becoming the "hacker." Within ethical boundaries, it is extremely beneficial for students to further their understanding of these attacks by actually performing them and seeking to understand the motives and techniques used by real hackers. The skills gained by performing these attacks can in return be applied to thoroughly securing real world applications from various threats that exist today.

Learning computer security via hands-on exercises does, however, pose an ethical challenge within the classroom since these exercises give students the opportunity to perform actual attacks employed by a real hacker. Due to the exploitive nature of those attacks, it is imperative that the attacks take place within an isolated environment dedicated to such exploitations. Attacks should not take place against live sites and applications.

This research project explores the new horizon of information security exercise paradigm by building a research application that mimics a fully functional online bookstore, but is purposefully designed to allow for exploitation that yields desirable results that would be sought after by a real "hacker." In addition, multiple users may exploit the application at the same time, creating competition among the users. The engine of the application is designed to accommodate the skill levels of individual users and adapts to the difficulty level of the application accordingly so that the application is not too easy or too difficult for a user learning computer security through exercises offered by the application.

II. Background and Motivations for This Research

Research efforts exist that aim to determine the most beneficial environment for students to practice security lessons. Studies presented in [3, 5, 6, 7] illustrate the importance of holding practical security exercises in a laboratory environment that mimics the real world scenario. Researchers also built network security test beds and live exercises for students to practice hands-on network attacks and defenses [6, 8, 9].

There are some existing security applications that provide students with an ethical environment to practice carrying out attacks in a hands-on environment. However, we feel that these security applications do not provide the optimal environment for learning computer security through practicing attacks. While applications such as WebGoat [1] provide interactive exercises for performing attacks, they provide the exercises in a modular environment. This means that users are provided with separate mini-security exercises to be completed in a step-by-step order, only being provided with information and components required to complete the attacks. In addition, the individual mini-exercises are not related to each other in terms of the application context. For example, one exercise is related to gaining access to an employee database and another exercise might be related to exploiting an online shopping website. The attacks do not link together for achieving an ultimate goal. Such modularity is beneficial to beginners, but does not meet the levels required by more advanced users.

Another existing web security application, HackQuest [2], provides a vast array of security exercises varying from attacks dealing with logic to Javascript vulnerabilities. HackQuest divides itself into series of exercises, but does not provide the user with step-by-step exploitation instructions. Unlike WebGoat, HackQuest forces the users

Manuscript received July 16, 2011. This work was supported in part by the Northern Kentucky University Research Foundation Student Undergraduate Research & Creative Awards (SURCA). Emily Crawford is with the Department of Computer Science at Northern Kentucky University, Highland Heights, KY 41099 USA (e-mail: crawforde1@nku.edu).

Yi Hu is with the Department of Computer Science at Northern Kentucky University, Highland Heights, KY 41099 USA (e-mail: huy1@nku.edu).

to figure out their approaches of attacks to solve the exercises, but the environment provided is still modular. The users are still presented with exact structures and tasks for performing individual exercises, thus the application lacks the reconnaissance aspect of actual attacks to some extent.

The drawback of this setup is that by targeting only a specific application scenario per exercise, users are not learning the full impact of these security vulnerabilities in a real world scenario. While users must formulate their approaches of attacks, they are unrealistically shielded from the remaining components of the application. Users therefore lack the experience of scoping out an application based on motives that inspire real hackers and based on the real techniques used to achieve those motives – a real world understanding vital to properly securing applications. Thus, due to the modular nature of existing security applications, these exercises are not completely effective for learning the security concerns that would be intertwined in a commercial application.

At the University of Wisconsin-Eau Claire's Department of Computer Science, Paul J. Wagner and Jason M. Wudi also researched the pedagogical and management issues relating to the development of a cyberwar laboratory exercise [5]. Wagner and Wudi developed what they refer to as a "cyberwar" laboratory exercise where students worked in teams to secure a computer system and then tried to gain access to other systems present on the network, including the systems used by other teams. Wanger and Wudi both believe that better understanding security issues lies in the application of the attacks so that students can better understand the mindset of an attacker.

Another research effort also aims to provide students with hands-on experience in a real-world scenario. An Internet Role-game for the Laboratory of a Network Security Course [3] by Luigi Catuogno and Alfredo De Santis, focuses more on the maintenance of network services rather than just the hands-on experience of attacking an application or another computer. This research proposes a "role-game of the Internet" which was designed as part of lab activity of their Network Security course. This activity has students working together to perform tasks over a simulation of the Internet while preserving the security and availability of featured network services.

An approach using VMware VCenter lab manager for carrying on various network security exercises is also illustrated in [10].

Numerous security exercises exist in the form of network and cyberwar competitions, but few exist in the form of a comprehensive online application. While overlapping exists, network attacks differ from the attacks that threaten online applications. Although network security does affect the level of security of an online application, it can only provide so much protection. If an application is not coded and validated properly, then no amount of network protection can compensate for the lack of application security.

Our research application gives students a holistic experience in exploiting different vulnerabilities in a single application so that they can learn security concerns and the impacts of security vulnerabilities in a real world scenario. Also, the engine of the application is designed to accommodate the skill levels of individual users and adapts the difficulty level of the application accordingly. As a result, users with various levels of computer security expertise will feel that the application is appropriately challenging. In addition, we verified the effectiveness of the new information security exercise paradigm proposed. Empirical comparisons with existing exercises such as WebGoat and HackQuest were also carried out. Moreover, pedagogical and development issues related to building such a security exercise were investigated.

III. Design and Goals

This research project explores the new horizon of information security exercise paradigm through the construction of a research application that mimics a fully functional online bookstore, but is purposefully designed to allow for exploitation that yields desirable results that would be sought after by a real "hacker." It produces an application for "ethical hacking" that users can use as an educational tool to better learn computer security concepts through hands-on practice. This security application improves upon existing applications by offering a realistic and adaptive setting that allows students to seek out vulnerabilities in an ethical environment through their own tactics. The GUI of the home page of the application is illustrated in Figure 1. No hints for exploitation are provided, so the users have to try various reconnaissance steps to identify vulnerabilities.

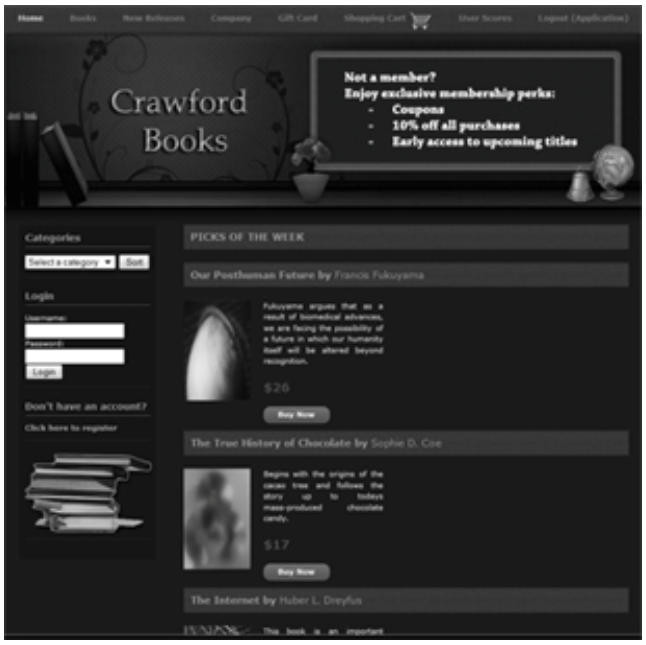

Figure 1. Overview of the home page of the research security application mimicking an online bookstore.

A. Architecture of the application

The application is designed to mimic a real online bookstore as well as to provide an ethical, realistic, and comprehensive environment to users. Figure 2 shows the architecture of the application. This entails creating three different layers for our application:

ISBN: 978-988-18210-9-6
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

- *Web interface*: This is the graphical user interface that the users interact with.
- *Application engines and vulnerability configuration component*: Application engines include the exploitation scoring engine and business transaction engine. The exploitation scoring engine is responsible for scoring a user's performance. Business transaction engine is the master control unit for various business transactions of a bookstore. The vulnerability configuration component is for adaptive configuration of the application in the way that makes the application to be appropriately challenging.
- *Databases*: They include *Book Store DB* and *Exploitation Scoring DB*. They store all of the user and application data including application contents, user accounts, user scores and other performance information.

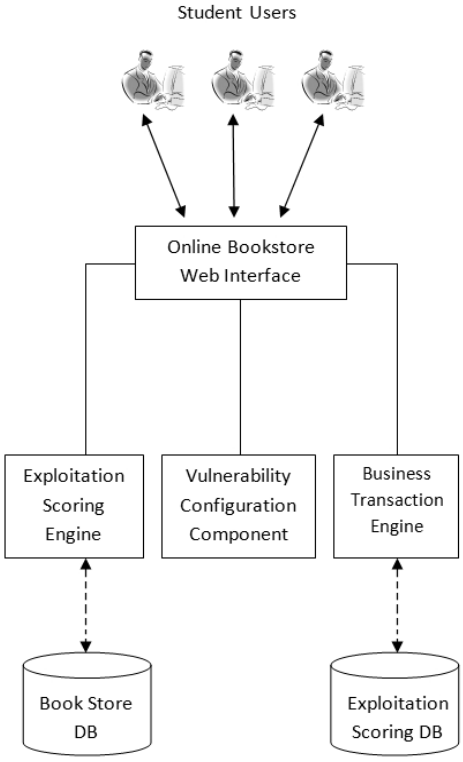

Figure 2. The Architecture of the Research Application

B. Design of the Adaptive Application

In order to accommodate various levels of user skills, the application allows for the adaption of contents to user skill levels. We designed an algorithm to assess the user skill level and dynamically adjust the contents of the application as a result. This method allows for the tailoring of the attack complexity to the user's security expertise. The goal is to avoid deploying an application that is either too simple or too complex for users. The complexity of the application will be determined by the user and task attributes such as:

- The difficulty level of each exploit
- The number of points that a user scores per time unit
- The skill level of an average user
- The bound on the error of estimation of the skill level of an average user

Based on the measurements of the above criteria, the application can then determine how advanced a user's skill level is and will appropriately adjust itself through logic checks within the vulnerable sections of the application.

The design of the adaptive engine is as follows. A task may include multiple exploits. Each exploit of a task is assigned a difficulty level. In addition, different varieties of an exploit are designed with different level of difficulties. Let us define the difficulty level of an exploit n as $d(n)$. The difficulty level of a task t is defined as:

$$d(t) = \frac{(d(n_1) + d(n_2) + ... + d(n_k))}{k}$$

where k is the number of exploits needed to perform a task.

We designed each exploit in the way so that each may be varied in three different difficulty levels, i.e., *easy*, *medium*, and *difficult*.

The security skill of each user is evaluated based on the number of points scored per unit time. In order to know how advanced the skill of a particular user is, we need a baseline, i.e., the number of points scored per unit time by an average user. The baseline of the average user expertise level is generated by using an empirical study. It is created by using a statistical sampling approach for estimating the mean number of points scored per unit time for a group of users with the similar technical background. Since this mean number may vary for different samples, our method will calculate the bound on the error of estimation for the mean of the number of points scored per unit time for an average user.

Let us start by defining the observed skill level of a user based on the training practice sessions of using the application. Let $\mu(t)$ be the mean number of points that a user scores during time unit t. We use the sample mean $\hat{\mu}(t)$ to estimate the population mean $\mu(t)$ based on a random sample of user practice sessions. In addition to estimating $\mu(t)$, our method places a bound on the error of estimation. $B(t)$ is defined as the bound on the error of estimation of the mean number of points that an average user scores during time unit t. Then we have:

$B(t) = 2\frac{s}{\sqrt{n}}$, where s is the sample standard deviation and n is the sample size.

When the size of the training sample data is large enough (i.e., $n > 30$ for most samples), the distribution of the sample mean of the number of points scored by a user in time unit t is almost normal according to the central limit theorem [13] . We define the 95% confidence interval of the mean number of points that a user scores in time unit t as $CI(t)$. Then we have:

$$CI(t) = \hat{\mu}(t) \pm B(t)$$

It means that with 95% confidence, we estimate the mean number of points that a user scores in time unit t is at least $\hat{\mu}(t) - B(t)$ and at most $\hat{\mu}(t) + B(t)$. We designed an algorithm for defining the difficulty level of the exploit of a task as follows.

Algorithm:

Input: $\hat{u}(t)$, $B(t)$

s = 0; // Initialize total points earned

ISBN: 978-988-18210-9-6
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

```
d = 0; // difficulty level of the next exploit
       // easy: -1; medium: 0; difficult: 1
st =start time; //assign the current time to st
While (the final goal of the game is not met)
{
  Configure the game with difficulty level of d for the next
  exploit;
  If the user successfully finishes the next exploit
  {
     p = points earned for the exploit;
     s = s + p; //calculate the current total score
     Display points earned and the total score to the user;
     ct = current time; //assign the current time to ct
     et =ct – st; //elapsed time
     pp = s/et; //pp indicates the points earned per unit time
     if pp < μ̂(t) − B(t)
        d = -1; //next exploit with difficulty level: easy
     if (pp > μ̂(t) − B(t) ) and pp <( μ̂(t) + B(t) )
        d = 0; //next exploit with difficulty level: medium
     if pp > μ̂(t) + B(t)
        d = 1; //next exploit with difficulty level: difficult
  }
}
```

It can be seen from the algorithm that when the user's points earned per unit time, i.e., *pp*, is less than the lower bound $\hat{\mu}(t) - B(t)$, the difficulty level for the next exploit is set to the easy level. Similarly, when the user's points earned per unit time, i.e., *pp*, is larger than the upper bound $\hat{\mu}(t) + B(t)$, the difficulty level for the next exploit is set to the difficult level. Otherwise, the difficulty level is set to the medium level for the next exploit.

C. Goals for the Users of the Application

With this application, student users are provided with an isolated, ethical environment to practice carrying out various attacks that compromise the application. Users have various goals to work towards, and are awarded points based on the number of goals they are able to achieve. The four goals are:

- Download a book for free.
- Gain membership for free
- Add money to your gift card for free
- Delete another user's score

Each goal will be achieved using a variety of attack methods. In addition, the difficulty levels of exploits are adjusted dynamically to accommodate the user's expertise. Moreover, some goals are more difficult to achieve. For instance, less complex goals include adding money to a gift card for free and more challenging goals include stealing a session ID using PHP Injection in a stored cross-site scripting attack for deleting another student's score. It was our goal to make every possible attack and vulnerability as relevant as possible to the application's purpose. We wanted to avoid awarding "empty points" to users for simple attacks such as issuing a Javascript alert text box via reflected cross-site scripting. Each attack and vulnerability combines to progress users towards a realistic goal similar to that of a real hacker. The following provides a high-level example of how a user can delete another user's score in a real-time attack that exploits several vulnerabilities:

- Deleting another user's score:

This is the most challenging goal of the application. A user must log in as another competing user. The login form for the overall security application login is protected against SQL-Injection, so students cannot rely on common attacks to achieve this goal. The student will instead have to steal another student's session ID.

The student will need to keep in mind two factors from previous exploration of this application in order to execute the theft of a session ID. Upon downloading an informational brochure earlier in the application, the student will need to take note of its download URL: http://www.crawfordbooks.com/capstone/admin/download_file.php?filename=membership-details.pdf

The student should conclude that this URL could potentially allow the download of any file from the server. In addition, the student should also take note from previous explorations that the application sends out receipts via email during order confirmation. With this information in mind, this is how a student will begin the goal of deleting another user's score:

1. Determine what script is used to send the receipt emails by performing a directory listing on the application's admin directory. It can be noted that email.php is used for this purpose from Figure 3.

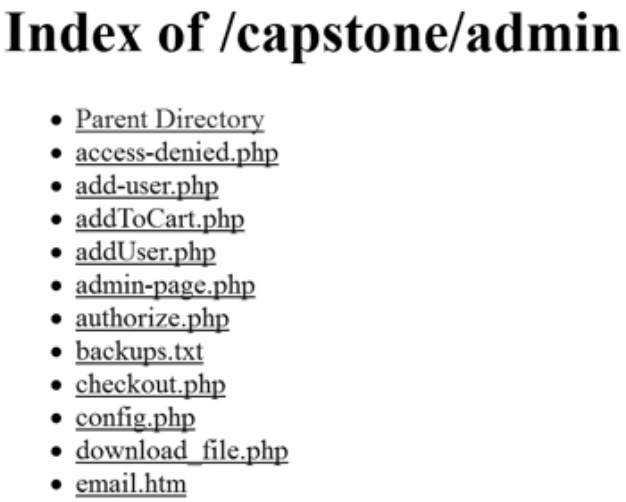

Figure 3. The Apache index listing for the admin directory.

2. Use the download_file.php script discovered earlier to download the email.php script found during the directory listing.
3. Inspect the contents of the script and take note that the email script is invoked through a simple function that accepts an email address and message as its two parameters. This can be seen from Figure 4.

```
function sendMail($email,$message)
{
        //Start session
        if(!isset($_SESSION))
        {
                session_start();
        }
        $DBConnect = @ mysqli_connect(HOST, USER, PASSWD);

        if(!$DBConnect)
        {
                die("Not able to connect to database.");
        }

        if(!mysqli_select_db($DBConnect, DATABASE))
        {
                die("Not able to select database.");
        }
        if(isset($message))
                $message = $message;
```

Figure 4. A portion of the content of email.php.

ISBN: 978-988-18210-9-6
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

4. Browse to the comment page for any book and insert the following PHP code into the comment form:
<?php sendMail("emily2025@msn.com", session_id()); ?>.
This is exploited through a comment page that does not impose any sanitization on user input. The page is illustrated in Figure 5.

Figure 5. An example of the PHP injection and stored cross site scripting attack on the comment page for a book.

5. After performing step 4, which is a stored crossed site scripting attack combined with a PHP injection attack, continue checking the provided email. When another student browses to the exploited comment page, the PHP script that was injected into the database as a stored cross site scripting attack will be executed upon the rendering of that page. This is because the logic for the comment page prints out whatever value is stored as a comment in the database table. The comment stored in the database at this point is actually PHP code. As the comment page is rendered, that PHP comment from the database is treated like normal code within the page and is thus executed. Since it is PHP code, which is not visible on a web page, the victim user will see no signs of the PHP and will not suspect an attack against him.
6. Once a session ID is emailed, the user can use Tamper Data to modify the cookie's PHPSESSID value to be the value of the session ID just received (See Figure 6).

Cookie	PHPSESSID=cfmuv8t46540n08173a7elcb56

Figure 6. An example of changing the cookie field from Tamper Data to allow the user to exploit another user's session

7. Now the user can log in as the victim user, navigate to "Logout (Application)" in the application and select "Logging out without saving" to delete the victim user's progress and score in the application.

D. Implementation of the Application and Discussions

When constructing this research application, the first goal was to create a fully functional online bookstore. Once the application was operational as an online bookstore, different vulnerabilities with various difficulty levels were deployed throughout it. By creating a fully operational application, this not only provides students with an ethical environment in which to perform exploitations, it also provides them with a real word setting in which to search for vulnerabilities. It gives students the benefits of learning in an environment that better mimics a real world setting versus a modular environment that is currently offered by other similar applications.

As for the actual construction of the application, it was built on an Apache, MySQL, and PHP framework. Once the framework was created, we started the development of the application by implementing the business functions of a working online bookstore. Vulnerabilities were introduced afterwards. This is because realistic vulnerabilities depend on a realistic environment. To make the online bookstore functional, it needs to allow users to perform such operations as:

- Register for an account
- Login to their accounts
- Logout of their accounts
- Browse the book selections
- Comment on books
- Manage their shopping carts
- Manage their gift cards
- Apply for membership
- Place an order

When the bookstore application was fully functional, it was time to begin implementing the vulnerabilities. The application is currently vulnerable to the following attacks:

- SQL-Injection
- Stored cross-site scripting
- Reflected cross-site scripting
- PHP-Injection
- Session hijacking
- Path traversal
- Source code disclosure
- Form field tampering
- Directory listings

All of the vulnerabilities lead up to a goal. These goals represent realistic motives that a real hacker might have when attacking a web site or application. The goals help in creating a realistic environment, moving away from modular exercises present in many existing applications.

An additional tool, Tamper Data, is also available to the users. Tamper Data is a free add-on for Firefox that allows users to view and modify HTTP/HTTPS headers and POST parameters. This can allow for the switching of sessions and for the modification of POST parameters. For example, users can change data stored in hidden form fields and session ID's in order to steal another user's session in a real time attack.

There are no step-by-step instructions for students, they must explore the site and figure out their own methods for exploitation. The goals all exist concurrently; the application is not divided up into separate, sequential exercises.

If successful with attacks, the user achieves a goal, obtaining points in the process. If unsuccessful at exploitation, the user will have to reformulate the approach and continue the attempts. At any point, the user can view the current score board and compare his score against other users' scores. This adds an element of competition to the application. Even if a student is unable to obtain all four

ISBN: 978-988-18210-9-6
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

goals, they will still receive points for individual attacks in the process. The application keeps track of what attacks are performed and sets a flag per individual attack. The flag prevents a user from repeating the same attack to increase the score. Users are also able to log out of the application and save their progresses if they wish to continue at a later time or they can discard their progresses and start over.

We assessed the learning outcome and effectiveness of the proposed security exercise paradigm. Feedbacks gathered from this security exercise showed that students are enthusiastic about playing the security role-game. Student users felt that it is an excellent tool for learning the theoretical and practical concepts of application security.

As part of the development of this application, research has been done on existing applications and their approaches on providing such an environment, how differences in the environments may potentially affect students' capacity to learn computer security (such as modular versus full scale security applications), and pedagogical and development issues related to building such a security exercise.

Table 1. A breakdown of comparisons between our security application and two other security applications

Application Comparison Factors	Online Bookstore Application	HackQuest	Webgoat
Free account registration	X	X	X
Multi-user	X	X	X
Multitude of attack types	X	X	X
No attack instructions	X	X	
User progress stored	X	X	X
Account management	X	X	
Comprehensive environment	X		
User expertise adaption	X		
Configurable application	X		
Real time student user attacks	X		
Competition mode	X		

It is also our desire for this security application to serve as a solid framework for future development in the field of computer security. It is vital that students learning computer security be provided with appropriately challenging hands-on practices of attacks that threaten applications. As noted earlier, there is a larger abundance of network based "cyber wars" that are well documented, but there is a lack of security exercises in the form of a comprehensive application. Both in practice and construction, this security application can serve as an example of comprehensive security application exercises that students can better learn computer security concepts from and in return, apply those concepts to the creation of secure applications.

IV. Conclusions

This paper illustrates the design and implementation of an adaptive research application that gives users a realistic experience similar to the experience that a hacker would encounter. Such an approach provides student users with the hands-on practice of executing attacks that they have learned in class as well as experiencing the mindset of a hacker. Simply knowing the syntactical signatures of attacks is not enough for properly protecting an application. Our application serves such needs in the form of a comprehensive, adaptive, and manageable security application. The components of our application improve upon the limitations encountered in existing modular security education applications by providing users with realistic settings and appropriately challenging exploits. In addition, the creation of such an application serves as the foundation for future adaptive security education application development.

References

[1] WebGoat Project, http://www.owasp.org/index.php/Category:OWASP_WebGoat_Project.
[2] HackQuest Application, http://hackquest.com/
[3] Catuogno, L. and De Santis, A. , An internet role-game for the laboratory of network security course. SIGCSE Bull. Vol. 40, Num. 3, Aug. 2008.
[4] Abler, R., Contis, D., Grizzard, J. and Owen, H., Georgia tech information security center hands-on network security laboratory. IEEE Transactions on Education, 49(1):82–87, 2006.
[5] Wagner, P. and Wudi, J., Designing and implementing a cyberwar laboratory exercise for a computer security course. In Proceedings of the 35th SIGCSE Technical Symposium on Computer Science Education, Norfolk, Virginia, USA, March 03 - 07, 2004.
[6] Mateti, P., A laboratory-based course on internet security, In Proceedings of the 34th SIGCSE technical symposium on Computer Science Education, New York, NY, USA, 2003.
[7] Stuttard, D. and Marcus P., The Web Application Hacker's Handbook: Detecting and Exploiting Security Flaws, Wiley, 2007.
[8] Tjaden, B. and Tjaden, B., Training students to administer and defend computer networks and systems, In ITICSE '06: Proceedings of the 11th annual SIGCSE conference on Innovation and technology in computer science education, New York, NY, USA, 2006.
[9] Vigna, G., Teaching Hands-On Network Security: Testbeds and Live Exercises, Journal of Information Warfare, Vol. 3, Num. 2, 2003.
[10] Lewis, J. and Lunsford, P., TLS man-in-the-middle laboratory exercise for network security education. In Proceedings of the 2010 ACM Conference on information Technology Education, October 2010.
[11] Yu, Y. 2007. Designing hands-on lab exercises in the network security course. Journal of Computing Science in Colleges, Vol. 22, Num. 5, May 2007.
[12] Wang, X., Hembroff, G. C., and Yedica, R., Using VMware VCenter lab manager in undergraduate education for system administration and network security, In Proceedings of the 2010 ACM Conference on information Technology Education, October 2010.
[13] McClave, J. and Sincich, T., Statistics, 10th edition, Prentice Hall, 2006.

ISBN: 978-988-18210-9-6
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)