CounterMeasures: A Game for Teaching Computer Security

Craig Jordan, Matt Knapp, Dan Mitchell, Mark Claypool and Kathi Fisler
Computer Science and Interactive Media & Game Development
Worcester Polytechnic Institute
Worcester, MA 01609, USA
email: {claypool,kfisler}@cs.wpi.edu

***Abstract*—Computer security has become vital for protecting users, applications and data, yet the field still faces severe shortages in skilled professionals. Typical methods to teach security from textbooks and academic papers are not engaging and take considerable time. Our hypothesis is that a security game that closely emulates real-world systems can improve learning about computer security above and beyond just reading technical documents. Our security game, *CounterMeasures*, provides a game-type environment for learning and practicing security skills through a series of guided objectives. CounterMeasures uses a real, interactive shell for input and targets a real server for exploits to provide an environment resembling security systems currently deployed. Evaluation with 20 test subjects illustrates the merits and shows the potential of our approach.**

I. Introduction

The increase in computer and network usage, with over 146% growth in users since 2000, along with the overall lack of technical knowledge by the public, has made computer security vital for protecting legitimate users and applications from harm [1]. As computer technology continues to spread, so has the importance of security, leading to an inability of security specialists to keep up with new, increasingly sophisticated attacks. U.S. federal officials have estimated there are only about 1000 security specialists in the United States where between 10,000 to 30,000 are needed [2].

Despite the importance of learning about security, it can be difficult to begin acquiring the relevant skills. Many security resources require a significant background in programming, networking, system administration, operating systems, and even previous security knowledge. This high level of prerequisite knowledge makes current security resources generally unsuitable as entry points for the beginner. Specific information about security is scattered and difficult to find at best, and very technical and specific to a particular technique, at worst. Adding to this problem is the lack of environments in which to practice computer security skills. Few texts or online resources provide a practical application of security techniques, instead only teaching the theory behind the techniques. Practicing computer attacks or securing server defenses requires various exploits that are preferably executed in a legal, harmless environment. Not only is it illegal to break into systems that others own, but it can even be illegal to subvert software that was legally purchased. Practicing attacks or defenses also runs the risk of damaging the system on which the exploits or patches are run.

Games are becoming a popular tool for education for many subjects [3]. Games can engage and interest learners for long periods of time and can both teach and test complicated concepts, such as computer security. Computer security is an abstract, technical subject which seems well suited for learning via simulations and games.

Online security game-type competitions feature two or more teams that implement several different services and keep the services running securely, all while trying to attack the other teams' computers. The teams are scored after each service implementation deadline and each team receives a set number of points for correctly implementing the service. Any team that breaks into an opposing system is awarded a large number of points and the opposing team is given time to re-secure their system. For a security novice, starting in such an environment can be intimidating, and from our personal experiences, even a player skilled in computer science may be unable to assist without being skilled in computer security. Thus, while there are a small number of games that teach security or allow practicing of security-type concepts, to the best of our knowledge, there are no games that teach a player computer security, allow the player to practice what they learn, and then test the player on what they have learned.

Our work presents a game called *CounterMeasures* that teaches specific techniques used by security experts and allows the player to practice these techniques in a game environment. CounterMeasures assumes no previous security knowledge, but does assume an introductory level knowledge of college computer science. CounterMeasures provides a single resource for security basics and includes a hands-on environment for practicing security skills, all while making the content engaging in a game-type environment. Experiments with CounterMeasures allow us to test two hypotheses: 1) Participating in a training simulation that closely emulates real-world systems is a better platform for learning security concepts than reading about security concepts in a technical document, and 2) Learning practical knowledge about computer security from a game is more engaging and less time consuming than learning the same material through textbooks and academic papers.

In order to test these hypotheses, CounterMeasures teaches players knowledge and skills using the tools and techniques of security experts through a series of missions. Training mis-

978-1-4577-1934-9/11/$26.00 ©2011 IEEE

sions assist the player in security fundamentals while teaching and testing individual security skills. Live missions provide less assistance on individual skills, but require application of the fundamentals through game-type objectives. For testing, three training missions teach scanning, buffer overflows, and format strings exploits and one live mission tests the use of the three previously learned skills, all within a flexible framework that can be extended with more missions.

CounterMeasures was developed using Flex/Flash running in an Adobe Air client. The client connects to a Java server, sending commands to a virtual machine and receiving mission information from XML files. This implementation provides a user-friendly, graphical interface for players, while still having at its core a command-line shell and real back-end servers that can be used to test exploits.

User studies with CounterMeasures ran over 3 days and attracted 20 participants. An experimental group played CounterMeasures and a control group read from a packet of condensed computer security information, with both groups completing the objectives present in the final mission. The results show that the control group took approximately twice as long as the experimental group to complete the mission even though both groups show approximately the same level of learning. The experimental group reported more engagement playing the game over the textbook-type learning of the control group.

The rest of this paper is organized as follows: Section II provides a brief overview of computer security education material and game-type environments; Section III describes the CounterMeasures design and implementation; Section IV details the user studies designed to evaluate the effectiveness and engagement of CounterMeasures in teaching computer security; Section V analyzes the results of the user study; and Section VI presents conclusions and possible future work.

II. Related Work

Books on computer security range from basics such as setting and storing passwords [4] to advanced topics on stack overflows and other code and system exploits [5], [6]. While these books provide a good basis for learning about security and hacking, they do not always provide enough context. Samples in the books can be tried, but a most of the material is presented in a theoretical fashion without much opportunity for practice. Our project attempts to provide the same theory as the books but also gives the user some practical experience to aid in learning the concepts.

Erickson's book *Hacking* [7] explains computer security techniques in a broadly accessible way and also includes a Linux environment the reader can use to follow the book's examples. This provision for interactive testing can be an improvement over a reading-only approach, but still does not provide the game-type objectives and guidance afforded by CounterMeasures. In fact, the control group in our experiments (see Section IV) is comparable to using Erickson's book, allowing for a direct comparison with the CounterMeasures approach.

There are about a dozen games that use computer security concepts as part of their gameplay mechanics. These games range from simulations that utilize real applications and exploits to puzzle games where hacking is the theme of the game but is trivial to the core gameplay. A few notable games include: *Street Hacker*,[1] a single player game in which the player participates in a story through mission-based gameplay. The interface is that of a fictional operating system with similar functionality to many distributions of Linux. *Hacker Skills*,[2] a browser-based hacking simulation with gameplay varying from actual exploits to logic puzzles utilizing Web languages and development practices. The scope of the puzzles are all contained within Web development, but the game does promote self-learning. *Cypher*,[3] a browser-based multiplayer game where teams complete mission-based gameplay. Although gameplay involves various pseudo-programs (e.g. password crackers and encrypters), the computer security is not realistic but instead is simply a theme used to inform the story and interface. Our project draws upon theses ideas of a game with real security concepts, mission-based objectives, and gameplay-type incentives to both inform and provide practice for computer security.

Several Web sites provide sets of challenges designed to test some security knowledge or hacking skills. More notable sites include: *HackThisSite*[4], a Web site with challenges designed to teach Web hacking (the ultimate challenge is to try and break into the site itself); *SmashTheStack*[5] and *OverTheWire*[6] focus on hosting security competition wargames; and *CrackMe*[7] focuses on reverse engineering of exploits. While these sites provide realistic environments on which to practice, most do not provide guidance or advice on how to proceed. CounterMeasures provides a guided learning environment.

Several organizations run online events that provide a real-time, realistic hacking game. In *DEFCON*[8] teams capture computers, keep services running, and protect the computers they have captured. The SANS institute[9] hosts *Netwars*, a game designed to test knowledge learned in the SANS computer security courses. *Wargames*[10] provides team versus team games where each team is given a real virtual machine to secure while attacking the others'. While these competitions are helpful for applying computer security practices, most provide little or no introduction to the material, and participants are expected to already know most of the techniques they need to succeed. CounterMeasures helps to emulate the hands-on aspects found in the games, but with a strong focus on helping users to learn security topics, too.

[1] http://www.streethacker.com/
[2] http://hackerskills.com/
[3] http://cypher.extremecast.com/
[4] http://www.hackthissite.org/
[5] http://smashthestack.org/wargames.php
[6] http://www.overthewire.org/wargames/
[7] http://www.crackmes.de/
[8] http://www.defcon.org/
[9] http://www.sans.org/
[10] http://p6drad-teel.net/˜windo/wargame/

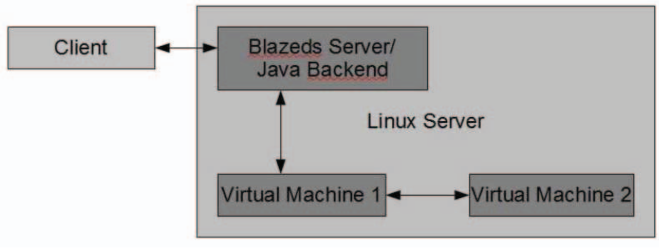


Fig. 1. CounterMeasures System Architecture

III. Counter Measures

This section outlines the game elements and technical implementation of CounterMeasures.

A. *Game Design*

CounterMeasures is designed as a single-player game. The player is guided through several missions, each teaching a new aspect of security. Each mission has a title, a description, a score for completing the mission, a skill as the focus of the mission, objectives required to complete the mission, help given to guide the user, and a list of commands learned during the mission. The missions build upon each other, allowing the player to utilize previously learned skills in each new mission. Players are given a fully functional shell that runs player commands while working on a mission. In order to still provide a sandbox environment, some commands are blacklisted (i.e. not allowed) to prevent the user from causing damage to the test environment, as well as help guide the user in their learning. As a player completes a mission, the score is incremented according to the difficulty and objectives of the mission. For more difficult missions, the player can seek help, but at the cost of subtracting from the score.

B. *Implementation*

CounterMeasures uses a client-server architecture. The CounterMeasures client is installed on the player's machine. This client application is developed using Adobe Flex[11] with Java, with Adobe AIR[12] to allow Flex to run on the desktop. Although tested on Linux, AIR and Java make the client easily portable to other operating systems. The client program contains the graphical user interface (GUI), through which the user interacts, sending commands to the server which handles calculations, retrieval of data, and running commands.

Figure 1 depicts the CounterMeasures architecture. The server is a Linux PC, running a fully patched version of Ubuntu (v10.04, Lucid Lynx) and configured to allow incoming connections from clients. The backend is implemented in Java, powered by BlazeDS[13] for easy integration with Adobe Flex and Adobe AIR, and a Tomcat Web server,[14] an open source Java Servlet and JavaServer Pages technology. The server has two virtual machines (vms), powered by VirtualBox[15], an open source virtualization product. Each vm runs BackTrack 4,[16] a Linux distribution designed to be used for penetration testing and with several built in security tools and vulnerable versions of programs. One vm is used for client connections and runs player actions, and the other vm is used as a target for player attacks. The server uses jsch,[17] a Java secure shell library, to connect to a virtual machine. Vulnerable programs have been compiled with the `-fno-stack-protector` option to allow stack smashing, where appropriate. The server keeps an Oracle database with player information, including login, password, and mission information for the player.

Upon startup, the Flex client establishes a connection to the BlazeDS server through the Tomcat Web server. JDBC is then used to connect from the BlazeDS server to the Oracle database to gather user information, such as score, completed missions, and current mission. Commands typed by the player on the client are piped over the network to the server, which sends the commands to the virtual machine to be executed, returning the results. The player thus appears to have an active shell embedded in their client GUI.

Mission information is stored in XML, with the schema compiled using XML Beans,[18] allowing the Java server code easy access to the mission information. Each mission has a title, description, score, skill, a list of objectives, a list of help content, and a list of commands that the player needs to learn and make use of during the mission. The objectives stored in the XML file are stored as text, with the logic for checking for completed objectives embedded in the Java server-side code.

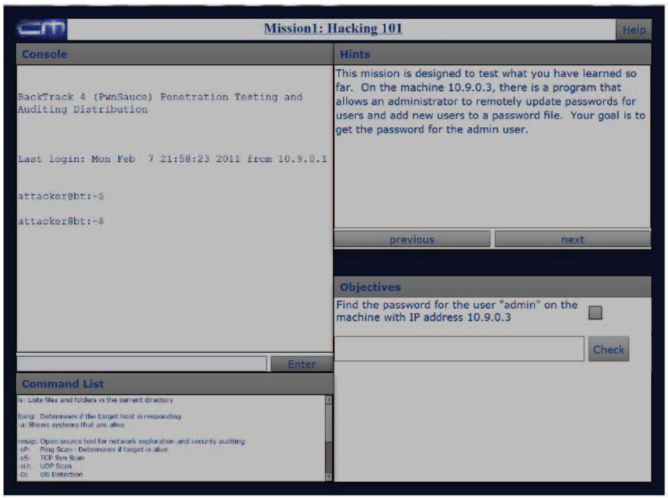
C. *Game Interface*

Figure 2 shows the game interface for CounterMeasures. The console is in the upper left of the screen, where the player enters the commands to run and the output from the commands is displayed. The pane to the right of the console contains help and hints on commands as well as insights into completing the next objective. The objectives pane in the bottom right provides the list of objectives and shows progress towards completing the mission. The bottom left pane shows a list of all the commands the player has learned in previous missions.

D. *Mission Content*

Gameplay is provided through *training missions* and *live missions*. Training missions focus on one specific aspect of security with the objectives clearly laid out for the player in the help pane. Live missions are more complicated than training missions and combine several aspects of security that were already covered in previous missions. Live missions only have basic help, leaving the players to figure out how to complete the mission objectives primarily using their past knowledge.

[11] http://www.adobe.com/products/flex/
[12] http://www.adobe.com/products/air/
[13] http://opensource.adobe.com/wiki/display/blazeds/BlazeDS
[14] http://tomcat.apache.org/
[15] http://www.virtualbox.org/
[16] http://www.backtrack-linux.org/
[17] http://www.jcraft.com/jsch/
[18] http://xmlbeans.apache.org/



```
BackTrack 4 (PwnSauce) Penetration Testing and
Auditing Distribution


Last login: Mon Feb  7 21:58:23 2011 from 10.9.0.1

attacker@bt:~$

attacker@bt:~$
```

This mission is designed to test what you have learned so far. On the machine 10.9.0.3, there is a program that allows an administrator to remotely update passwords for users and add new users to a password file. Your goal is to get the password for the admin user.

Find the password for the user "admin" on the machine with IP address 10.9.0.3 □

```
ls: Lists files and folders in the current directory

fping: Determines if the target host is responding
-a: Shows systems that are alive

nmap: Open source tool for network exploration and security auditing
-sP:    Ping Scan - Determines if target is alive
-sS:    TCP Syn Scan
-sU:    UDP Scan
-O:     OS Detection
```

Fig. 2. CounterMeasures Game Interface

Training Mission 1 - Finding Your Target: This training mission teaches players the basics of scanning a remote system given an IP address. Players learn how to determine if a system is alive, what operating system it is running, what ports are open, and what services are running on those ports.

Objectives:

- Determine if the system is alive using `fping` and `nmap`
- Scan the system and determine the open ports using at least two different tools
- Determine what operating system the system is running

Training Mission 2 - Bypassing Authentication Description: This training mission teaches the basics of a buffer overflow attack. Players must exploit a specific program using this technique.

Objective:

- Get the key protected by the program "adminKey"

Training Mission 3 - Discovering Passwords: This training mission teaches the basics of format string vulnerabilities. Players learn how to take advantage of insecure code and use format strings to access program memory.

Objectives:

- Get the database password used in the program "access_database"

Live Mission 1 - Hacking 101: This mission tests players on a combination of skills learned in training missions 1-3.

Objectives:

- Crack the password for the user "admin" on the system with IP address 10.9.0.3

IV. Evaluation

CounterMeasures is designed to teach users about computer security in a way that is fun. Evaluation focuses on testing two hypotheses: 1) Participating in a training simulation that closely emulates real-world systems is a better platform for learning security concepts than reading about security concepts in a technical document, and 2) Learning practical knowledge about computer security from a game is more engaging and less time consuming than learning the same material through textbooks and academic papers.

The target demographic for this project is people who have a general technical background, but without necessarily having prior computer security experience. This includes college students studying computer science, information technology, or related fields.

Evaluation proceeded through a user study with two main measures for assessment: performance statistics and questionnaires. All users were given a preliminary questionnaire[19] testing basic computer security knowledge. Users were then divided into two groups, an experimental group and a control group. The control group read a condensed selection of reading material on the security topics covered in the game for approximately 25 minutes (the approximate duration of one game session). The experimental group played through the three training missions in the game. Both groups completed the final objectives of the live mission (crack the password for a given user at a given IP), with the experimental group playing the Live Mission 1 and the control group provided the same objectives and a Linux shell on a PC to use. For each user, the time to complete the objectives was recorded, as well as the number of help screens needed if an experimental user. After the experiment, all users were given a questionnaire to test security-related knowledge similar to those on the first questionnaire, with additional questions related to the users' enjoyment of the experiment and interest in further security education.

The study took place in private laboratory exclusively used by those taking part in the experiments. There were two laptops setup running CounterMeasures for users in the experimental group, and two PCs with a shell setup for users in the control group. All computers were physically separated, so that a user at one computer could not see what a user at another computer was doing. As users arrived, they were given the pre-study questionnaire and then randomly assigned to either a laptop running CounterMeasures or to a PC with the shell. After completing the study, all users were given a post-questionnaire to complete.

Users were solicited from among WPI undergraduate students. An email inviting participation was sent to all Computer Science and Interactive Media & Game Development majors. An oral pitch was also made to all participants in an introductory operating systems class. Students were informed that users were needed for a study related to computer security. Students were told that they could expect the study to take approximately 30 minutes and that pizza would be provided to those who participated. Students in the operating systems class were also told they would received a bonus point towards their class grade for participating.

[19] The full text to this and all other questionnaires can be found online at: http://www.cs.wpi.edu/˜claypool/papers/security-game/

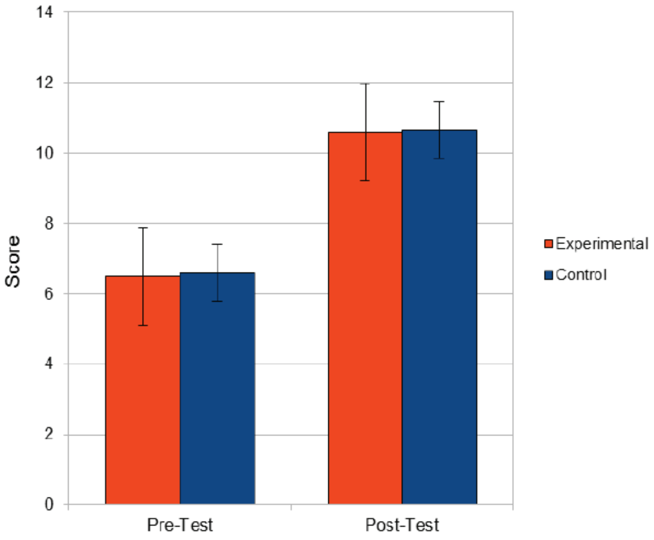


Fig. 3. User Score on Security Test Questions

V. Results

This section presents the results of the experiments, including a summary of participation, and objective and subjective measures of performance.

A. Participation

There were 20 participants total, divided evenly into 10 participants in the control group and 10 participants in the experimental group. All of the users were undergraduate students at WPI and most users came from the operating systems course. All users had Computer Science backgrounds and were interested in computer security, but 19 out of 20 users indicated they had little to no computer security background.

B. Objective Measures

The questionnaires allowed testing of security related knowledge both before and after the experiments. The maximum possible score for both pre- and post-tests was 14 points.

Figure 3 depicts the users scores before and after the experiment. The y-axis is the score and the x-axis clustered by pre- and post-test, with the average score for the experimental and control groups indicated by different color bars. Each average is bounded with 95% confidence intervals.

The average scores in the experimental and control groups were approximately the same in the pre-test, suggesting that the computer security backgrounds for both sample groups were about the same. The average scores in the experimental and control groups were also approximately the same in the post-test, suggesting that playing CounterMeasures and reading materials provided the same amount of knowledge. The clear separation between the post-test scores and the pre-test scores shows a significant gain in knowledge. For both groups the most common mistake in the post-test was on indicating all the possible effects of a buffer overflow.

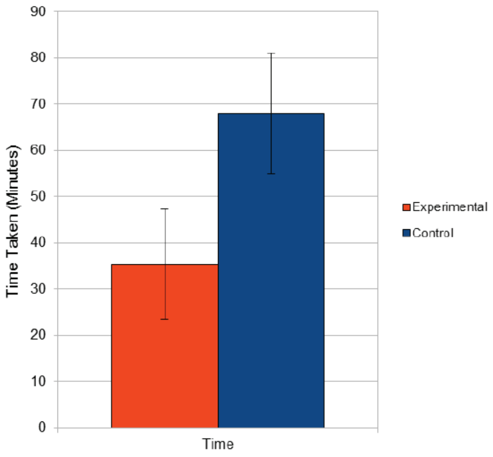


Fig. 4. Time to Complete the Final Objectives

Figure 4 shows the time taken by each experimental group to complete the final objectives (those of Live Mission 1 for the game players). The y-axis is the time taken, in minutes, with the two bars representing the average times for experimental and control groups, each bounded by a 95% confidence interval.

Experimental group users that played CounterMeasures took on average about half the time of control group users that read material in completing the final objectives (to crack the password for given user at a given IP address).

C. Subjective Measures

Figure 5 depicts the interest in computer security for both groups of users before (top) and after (bottom) the experiment. The y-axes are the distribution of answers on the 5-point scale provided, and the x-axes are the number of users that provided each response. The bars represent the total number of users that gave each response, with the experimental and control groups differentiated by color.

From the graphs, the distribution of interest in computer security for the control group users that read material is about the same both pre- and post-test. This suggests that reading about computer security does not, in itself, pique interest in the topic. In contrast, the distribution of interest in computer security for the experimental group that played CounterMeasures has a noticeable shift with more users very interested in security post-test versus pre-test (6 versus 2). This suggests that being engaging in active learning about computer security via a game can enhance interest in the topic.

Figure 6 depicts users opinions on their perceived learning of computer security material once the experiments had completed. The y-axis is one of the four possible responses, and the x-axis is the number of users that provided each response, with the experimental and control groups indicated by different colors.

Both groups of users felt they learned something about computer security from the experiment as there are no responses in

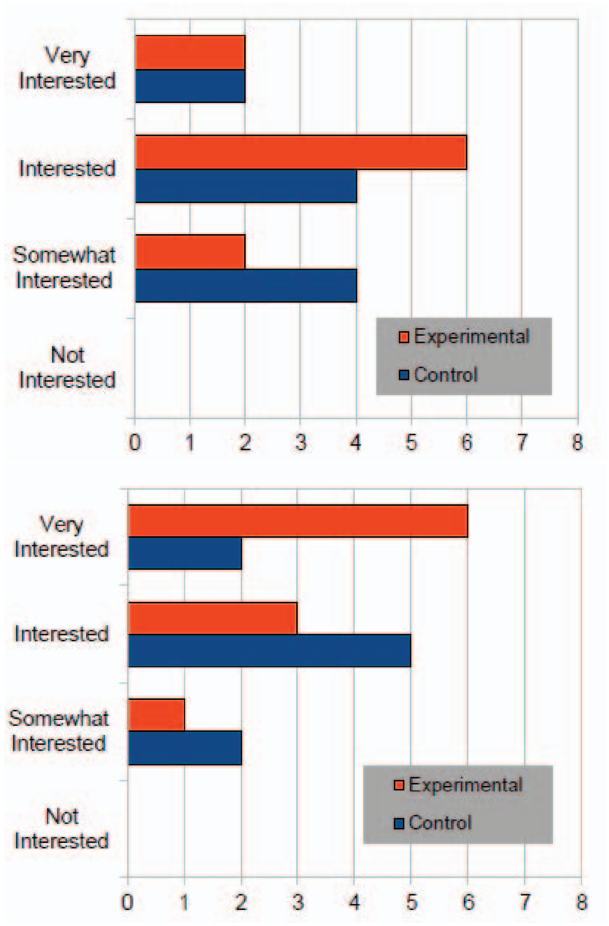


Fig. 5. Interest in Computer Security Before (top) and After (bottom) Experiment

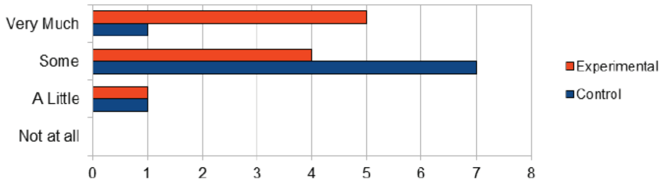


Fig. 6. Users' Opinions on Perceived Learning

the "Not at all" category. The experimental group that played CounterMeasures has a noticeably larger number of users (5) that thought they learned "Very Much" compared with the control group that read material (only 1).

Overall, post-questionnaires showed 18 of 20 users to be "Somewhat Interested" or "Interested" in additional games teaching computer security, and all the users in the experimental group enjoyed playing CounterMeasures "Some" or "A Lot".

VI. Conclusions

Computer security is increasingly important for protecting users' data and applications. However, learning computer security through traditional reading materials can be challenging and time consuming. Computer games have the potential to overcome these challenges by guiding computer security education while still engaging players in the topic.

The purpose of this project was to design, implement and evaluate a game called *CounterMeasures* that teaches specific techniques used by security experts. CounterMeasures is designed with a graphical interface to present help and objectives, but uses a command-line shell at the core for realistic access to computer security tools and techniques. CounterMeasures teaches about scanning systems, buffer overflows and string format vulnerabilities in a game-type environment with goals, objectives and a scoring system. CounterMeasures is implemented using a Flash client that sends player commands to a Java server which executes commands and provides game feedback via virtual machines. Our user study to evaluate CounterMeasures had 20 participants, providing a comparison of effort, learning and engagement with the game compared to equivalent reading materials.

Analysis of the results indicates that CounterMeasures teaches the same amount of knowledge as does reading on the computer security techniques tested, but in about half the time. Users generally found CounterMeasures more enjoyable and generally more immersive than the reading material, and only the CounterMeasures users wanted to pursue additional computer security learning opportunities after participating in the experiments.

In the future, CounterMeasures can be fairly easily extended in the form of additional missions. Missions consist of an entry in an XML document, with descriptions of the mission name, the objectives, scoring information, help and hints available. Additional game-type elements could include time constraints and other advanced scoring mechanisms along with a narrative with plot developments. A multiplayer version of CounterMeasures could provide head-to-head competition by following a common "capture the flag" format, where competing teams gain access to various insecure servers setup for the game. Additional structure could be provided in the forms of hints on how to defend, as well as attack, computer systems. A Flash front-end implementation could reach a wider audience.

References

[1] I. W. Stats, "Internet Usage Statistics, Population and Telecom Reports for the Americas," Jun. 2010, [Online at: http://www.internetworldstats.com/stats2.htm].

[2] D. Systems, "U.S. Cyber Challenge - Major Shortage of Cyber Security Workforce Professionals," Jan. 2011, [Online at http://www.usadefenseindustryjobs.com/2011/01/30/u-s-cyber-challenge-major-shortage-of-cyber-security-workforce-professionals].

[3] M. Prensky, "Digital Game-Based Learning," *ACM Computers in Entertainment*, vol. 1, no. 1, Oct. 2003.

[4] V. Authors), *Security - For Dummies*. Wiley, 2010, [Online at http://www.dummies.com/store/Computers-Internet/Security.html].

[5] J. Scambray, S. McClure, and G. Kurtz, *Hacking Exposed: Network Security Secrets & Solutions*. Osborne/McGraw-Hill, 2001.

[6] D. Gollmann, *Computer Security*. Wiley, Mar. 2001, ISBN 978-0-470-74115-3.

[7] J. Erickson, *Hacking: the Art of Exploitation*. No Starch Press, Jan. 2008, ISBN 978-1593271442.