Firewall Administration: The Game

Scott Thompson

4th Year Project Report
Computer Science
School of Informatics
University of Edinburgh

2017

Abstract

Managing the Firewall policy rules for a large network is a challenging task, even for a skilled system administrator. Learning these skills can seem insurmountable without the proper learning tools, particularly to those lacking the desire to become educated in firewall policy. The growing demand for properly trained computer security workers continues to be problematic in current society. In this dissertation, pre-existing computer security learning tools are investigated to reveal their inadequacies, and how they may be improved. In general, these current resources are seen as inaccessible to the common user. Factors such as significant required prior knowledge on the subject, considerable resource requirements, excessive setup procedures, and sometimes unnecessarily complex presentation of information, all contribute to the inaccessibility of current educational materials. The following research aims to address some of these issues by developing a new tool which simulates real firewall administration, in the form of an easily accessible online Flash game. Additionally, in presenting this game, I aim to contribute to the range of educational tools available for teaching firewall rule policy in system administration, as well as encourage the future development of additional learning resources in this field.

Table of Contents

List of Figures

Chapter 1

Introduction

Recent studies suggest that current and future demand for skilled Cyber Security workers is unmatched by the supply[11]. In response, many specialized education programs and funds are being supplied by governments globally, in an attempt to satisfy the immediate demand[7]. Most experts are in agreement that in order to meet the increased future demand for skilled security workers, standardized Computer Science teaching methods need to be addressed. A predominant problem surrounding current teaching practices is the lack of emphasis on Computer Security altogether. Many undergraduate Computer Science programs have no Computer Security requirements at all. While some universities offer such courses as electives, many do not offer them at all. Similarly, there is a lack of exposure to Computer Security in high school programs which offer general computer studies. In their paper titled *This is Not a Game*, Mark Gondree and Zachary N. J. Peterson describe issues with elective computer security courses, noting that "the technical complexities and mundane subtleties of computer security do not easily lend themselves to a lower-division college curricula". They layout two main issues with some Computer Security course curricula[9].

1. The lack of real-world context and social relevance
2. limitations on creativity and individual expression in overly-constrained coursework.

This research explores current learning tools in regards to these two issues, as well three others; high required base knowledge, high resource requirements and significant setup difficulties. The evaluation of prior work was then used to design a game addressing these issues. Through the development and evaluation of an alternative tool, I aim to contribute to the available resources used to learn firewall rules in computer security education.

1.1 Research Goals

The research presented in this dissertation is an investigation into the technology used to teach firewall administration skills. The goals of this research are as follows:

1. To determine the potential for improvement in existing tools designed to teach firewall administration skills.
2. Design an educational game to effectively teach firewall administration skills. To determine the viability of said game, the research will reflect on the following questions:
   (a) Does the game support the development of practical firewall administration skills and understanding the theory behind firewalls?

   (b) Does the accessibility of an online learning tool provide relevant benefit in comparison to other learning tools?

1.2 Paper Structure

The rest of the dissertation that follows is split up into five separate chapters.

Chapter 2 will review relevant literature and current available teaching tools, to provide background information on existing technologies used in teaching firewall administration skills. This chapter will also provide reasoning and justification regarding the purpose of this research.

Chapter 3 explores game design specifics, as developed for the purpose of this study. It will explore various aspects of the games design requirements, user interaction, and implementation.

Chapter 4 details the procedures for evaluating the game, as described in the previous design chapter. Evaluations will be made, based on observations of student participants interacting with the game.

Chapter 5 concludes the research presented in this dissertation and addresses the research questions posed in section 1.1. Additionally, this section will address the potential for future work which could further the presented research.

Chapter 2

Background

2.1 SEED: A Suite of Instructional Laboratories for Computer Security Education

The SEED project started in 2002 and was funded with a total of 1.3 million dollars from NSF. It is now used by hundreds of educational institutes[5]. The goal of the SEED project is to develop hands-on laboratory exercises called SEED labs for computer and information security education and help instructors adopt these labs in their curricula[5]. The SEED project, headed by Wenliang Du, has developed over 30 lab exercises which cover numerous learning topics which are relevant to this dissertation.

Participants in any SEED lab must install a virtual machine from a pre-built VM image, using either VMWare or VirtualBox. Some labs depend on particular images, while others require multiple virtual machines, each having a relatively large amount of configuration which is outlined in a six page User Manual[6]. After the setup, participants begin the lab sheet which provides an overview of the learning objectives for that particular lab. After completion of each task, participants are asked questions which explore the learning topics.

In 2008, Wenlian Du and Ronghua Wang published *SEED: A Suite of Instructional Laboratories for Computer SEcurity EDucation* [4], a paper discussing the SEED project. The publication addressed the effectiveness of SEED labs and discussed its encouraging results. When participants were asked to respond to the statement "The lab was a valuable part of the course", in reference to the IPSec lab, 69% of students strongly agreed, 29% agreed, and only 3% were neutral in their response. Similar positive results were found for other statements, such as "The lab sparks my interest in computer security". This evaluation, coupled with the fact that these labs are being used by hundreds of educational institutions demonstrate why SEED labs are considered the leading technology for computer security education. As such, it is most logical to examine SEED labs to discover areas with the potential for improvement.

One area of SEED labs in need of improvement is their method of providing feedback to users. Because there is no automated assessment provided, a student may get stuck on any given question or incorrectly answer lab questions. In the case of a supervised lab environment, it is infeasible for an instructor to insure that every student is receiving the correct feedback to learn the subject area sufficiently and in an efficient manner. In many cases, a student will be completing the lab outside of supervised instruction time, and as such do not receive instant feedback regarding the work they are doing. Efficiency could be greatly increased through the implementation of automated feedback and/or assessment.

Du and Wangs publication notes that students spent a lot of time just figuring out what exactly needed to be done. Additionally, they identify that the average length of time that it took students to complete a design and implementation lab was six weeks. It is impractical to know how long certain learning objectives should take to teach. However, it is evident that some parts of the lab can be revised to create a more efficient work flow, addressing issues such as the significant setup time seem imperative.

2.2 Elevation of Privilege - The Card Game

Elevation of Privilege is a card game designed to draw people who are not security practitioners into the craft of threat modeling. Elevation of Privilege was designed by Adam Shostack, in 2012, whilst working for Microsoft. The game is designed for 3-5 players and consists of 84 cards. Each card has one of the following six suits: Spoofing, Tampering, Repudiation, Information Disclosure, Denial of Service and Elevation of Privilege. Once the cards are distributed , players proceed in a clockwise direction around the circle, playing a card by reading it aloud and explaining how it applies to the system being threat modeled.

The educational effectiveness for Elevation of Privilege has not been formally evaluated, however in Adam Shostacks publication, *Elevation of Privilege: Drawing Developers into Threat Modeling*[22], the reception of the game was discussed. The game has apparent popularity, and is used widely for training purposes. Shostacks paper discusses lessons learned, design approaches and future work which can be used to identify areas of importance for improvement.

An important lesson learned from Shostack's research is that a game forces participation. A lack of understanding could not be concealed by nodding along and deferring to others as that lack of understanding is made clear when it is that players turn to play a card. This is advantageous to designs like a SEED lab in which deficiencies in understanding can more easily be concealed.

Another point that Shostack makes is the advantage of inexpensive graphic design being utilized. The game found some of its success as a result of its appealing art. Tester response was completely transformed when a more colorful version of the card game was introduced.

Finally, he concludes with the fact that not everyone is interested in participating in a card game. Some people have little to no experience with card games, thus, excluding numerous individuals who may otherwise have participated. This highlights the idea that the game should be designed in a manner which allows the most knowledge to be distributed, with the least amount of prior knowledge required from the participant.

2.3 Control-Alt-Hack

Control-Alt-Hack is a tabletop card game developed in 2012 by a team from the University of Washington, funded by Intel. Control-Alt-Hack is a complex computer security-themed card game designed to be entertaining, give a glimpse into white hat hacking, and highlight some lesser known aspects of computer security.

The game was formally evaluated in a 2012 publication, *Control-Alt-Hack: The Design and Evaluation of a Card Game for Computer Security Awareness and Education*[23], by the same team that developed the game. Feedback surveys and user studies were used to evaluate the game, with the game scoring high in social engagement and improving cyber-security awareness.

Criticisms of Control-Alt-Hack have been influential in my firewall game design, with much of the feedback suggesting that the length of time it takes to learn the game, as well as the actual game length is too long to hold participants interest. This re-emphasizes the need for improvement in user accessibility for such learning tools, as seen previously through a critical analysis of SEED labs. Additionally, some test subjects for both Control-Alt-Hack and Elevation of Privilege found that these card games were lacking in technical education value.

Chapter 3

Design

This chapter will provide an overview of the game's design, produced as a part of this dissertation's research. In this game, you play the role of a new hire in a system administration position. Your job is to manage the firewall and complete tasks assigned to you by your boss. This game has been designed for educational purposes, created to teach system administration skills for the IPTables application.

3.1 Specification

Section 3.1 outlines the specifications for the game, describing how the game was designed. Goals for the game will be identified and explained in this section. Additionally, the functional requirements will be outlined, based on the review of previous work.

3.1.1 Goals

Based off the original research goals, the following goals for the game were created:

1. The primary objective is to improve upon existing tools used to teach firewall administration skills
2. The game should emulate the experience of being a security system administrator in charge of managing Firewall policy rules.

The first goal addresses the shortcomings of current learning resources, as discussed in the introduction of this dissertation. It aims to provide a solution to issues such as inefficiency and accessibility in training methods, as proven by the current lack of trained cyber security workers. It is important to note that due to the scope of this research, it is infeasible to formally evaluate the success of this goal in a quantitative matter, against other tools. It is anticipated that the method and design used to create this game will be successful in achieving this goal, with an emphasis on conceptual improvement. Future research and development could include evaluation of the game in a more quantifiable way.

Managing firewall policy rules for a large network is a challenging task, even for a skilled system administrator. Testing and learning on a real network can be expensive and dangerous. As such, a game teaching these skills would make the learning process more cost efficient and most importantly, safer. The motivation of the second goal is to address the lack of teaching tools which provide real simulations of the problems faced by system firewall administrators.

3.1.2 Requirements

A set of requirements was initially drafted to accomplish the aforementioned game goals. The requirements were supplemented with features derived from the review of prior learning tools, as discussed in Chapter 2. The requirement set is not fully exhaustive as some requirements are self explanatory:

1. The game should simulate real system Firewall administration.
2. The game should teach skills for a real user-space application that allows system administration outside of the game (IPTables).
3. The game should be usable and designed for undergraduate Informatics students.
4. The game should provide real time feedback for correct and incorrect answers.
5. The game should be designed to maximize its' potential audience.
   (a) It should be designed to be widely accessible.
   (b) It should require as little prior knowledge possible.
   (c) It should have few hardware requirements.
6. The game should be colorful and have appealing art.
7. The game should have minimal setup time.
8. Game mechanics should not take a long time to learn.
9. The game should not depend on long play sessions.

3.1.3 Methodology

Agile software development advocates adaptive planning, evolutionary development, early delivery, continuous improvement, and encourages rapid and flexible response to change[1]. An Agile developmental methodology was chosen for several reasons. People and interactions are emphasized, rather than processes and tools[3]. This encourages continuous attention to good design for which end products are more likely to be have better usability because people are constantly informing the design. The flexibility afforded by this methodology when compared with other developmental methodologies allows an early and predictable delivery which is well suited to this project.

3.2 System Description

To enable the reader to fully understand the details surrounding the inner workings of the game system, this section provides a detailed description of the game. It describes details such as design decisions and their justifications. Additionally, the user interface and graphical interface are described and shown.

3.2.1 Overview

The developed game is classified as a browser game, A single-player computer game that is played over the internet on a browser[21]. Interacting with the user interface, the player participates in game play designed to teach firewall administration skills, based on the IPTables user-space application.

3.2.2 Game Flow

The following sections provide a walk through of a typical game play session, as seen by the intended user. Additionally, each section will describe design choices and implementation details, relevant to the section. Figure 3.1 gives a simplified illustration of the game flow.

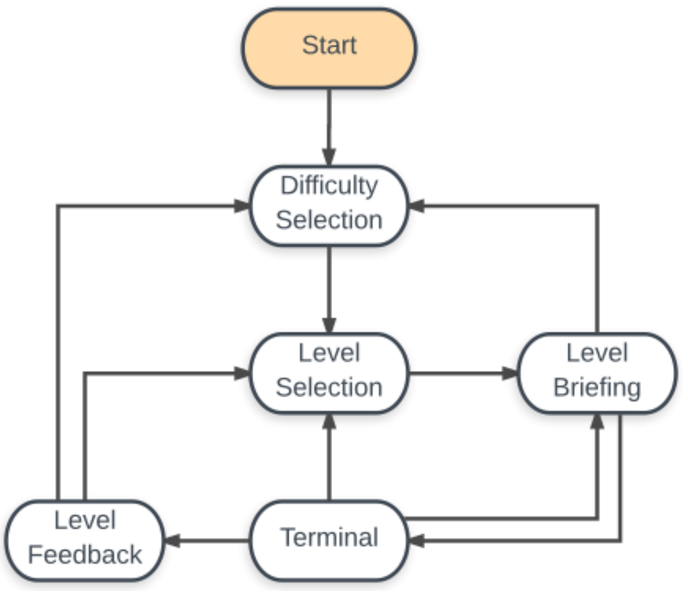


Figure 3.1: Game Flow Diagram

3.2.3 Start Screen

The start screen of the game is a users first impression, and should be visually appealing. The art on the start screen of this game was created to be aesthetically pleasing, aiming to attract and hold the attention of prospective players. Additionally, it was important to choose a visual style that appealed to the target demographic. With this in mind, it was decided that a work desk, situated in an office was the natural choice.

As shown in Figure 3.2, the start screen is a simplistic design. The only functionality is the start button, which the player clicks to continue. The simplistic nature of this screen was created to meet specific requirements which were neglected in previous attempts at Computer Security games (as described in the Background section). With the simplicity of this start screen, the game addresses the frustrations associated with difficult set up. For example, Control-Alt-Hack requires that players first fully understand the rules of the game, which could reportedly take up to 30 minutes[23], before players are able to interact with the actual game. To maintain player attention, I decided it was important to immediately engage the players, purposefully avoiding a bombardment of informational text, as to avoid discouragement before beginning the game.

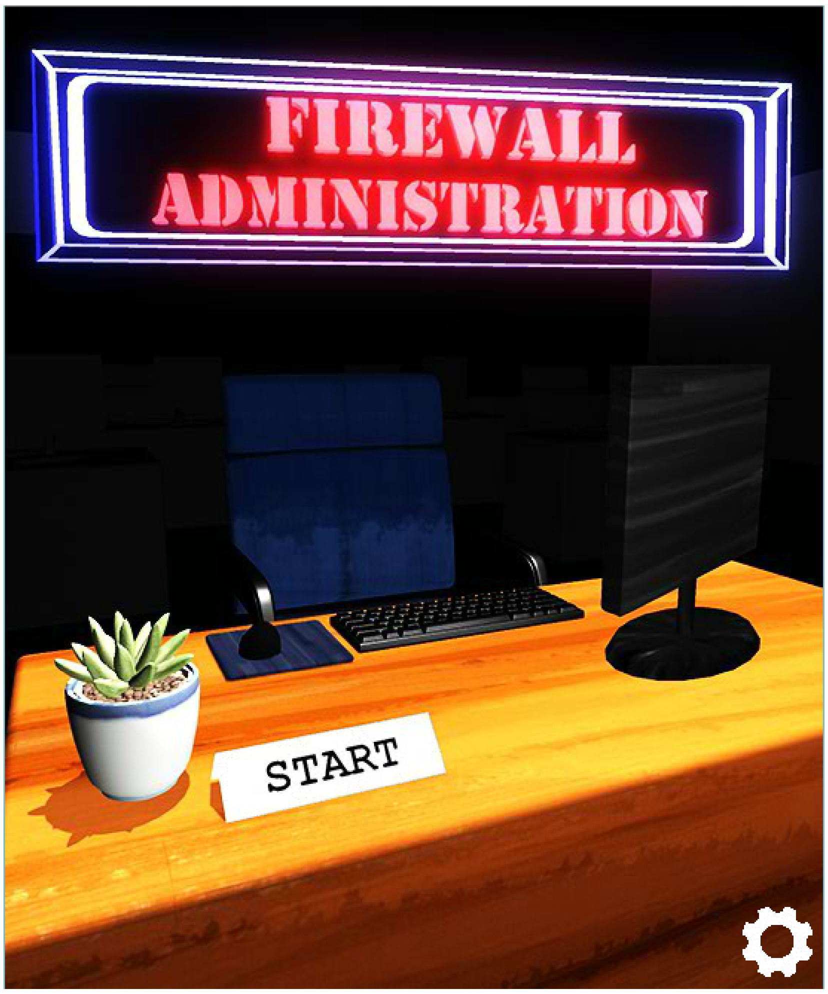


Figure 3.2: Start Screen

3.2.4 Difficulty Selection

The next screen, shown in Figure 3.3, prompts the player to choose the level of difficulty in which they wish to proceed. The difficulty setting influences what type of instructions the player will receive when given an in game task. There are three difficulty settings: Hard, Medium, and Easy. The player will choose the setting based on their confidence in their current skill set.

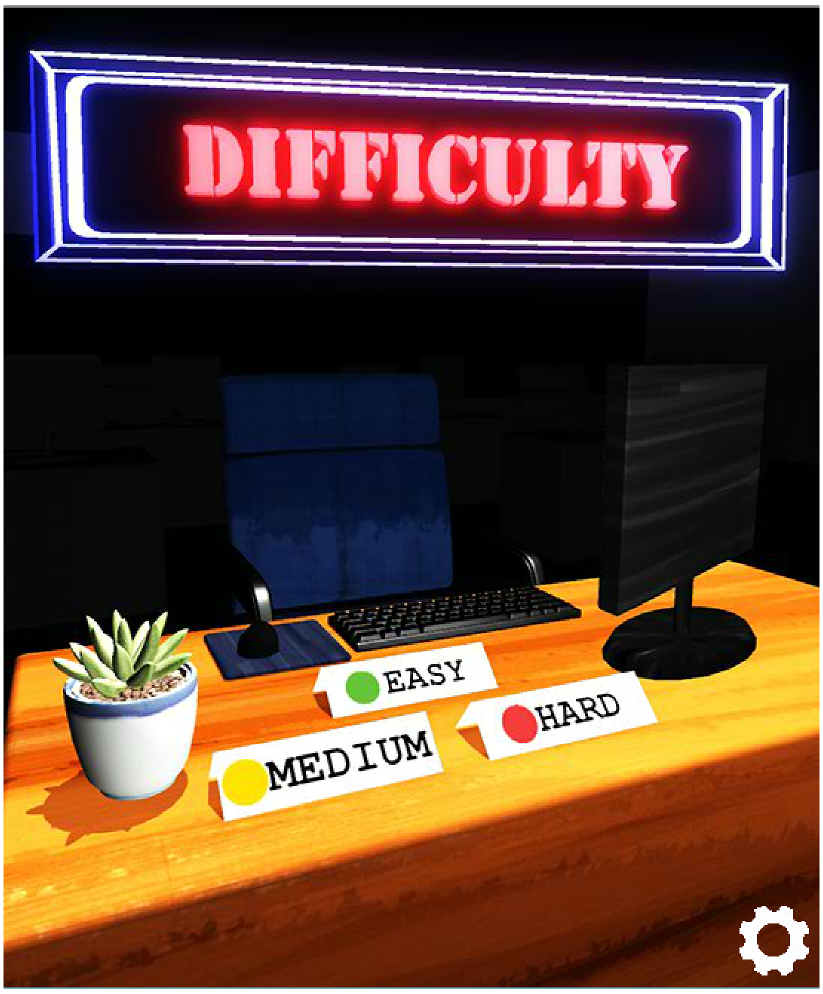


Figure 3.3: Difficulty Selection Screen

The justification for this game mechanic is to achieve the goal of maximizing the games potential audience. In the design process, it was decided that the game needed to be able to adapt to be a useful learning tool for people with varying background knowledge. Despite the tasks for each level being the same across difficulties, the ability to choose ones difficulty prevents the game from appearing tedious and unchallenging for more experienced players. In effect, this design feature decreases the odds of participants finding this tool ineffective.

3.2.5 Level Selection

The third screen reached by players is the level selection screen in Figure 3.4, in which the player chooses one of twenty-five levels to play. Each level tackles a different learning objective, with each level becoming increasingly more advanced. For example, in level one, the player simply must change a default policy for the firewall. In contrast, level twenty-four requires the player to create a new firewall chain with new firewall rules, routing all traffic from a specific IP address to this new chain.

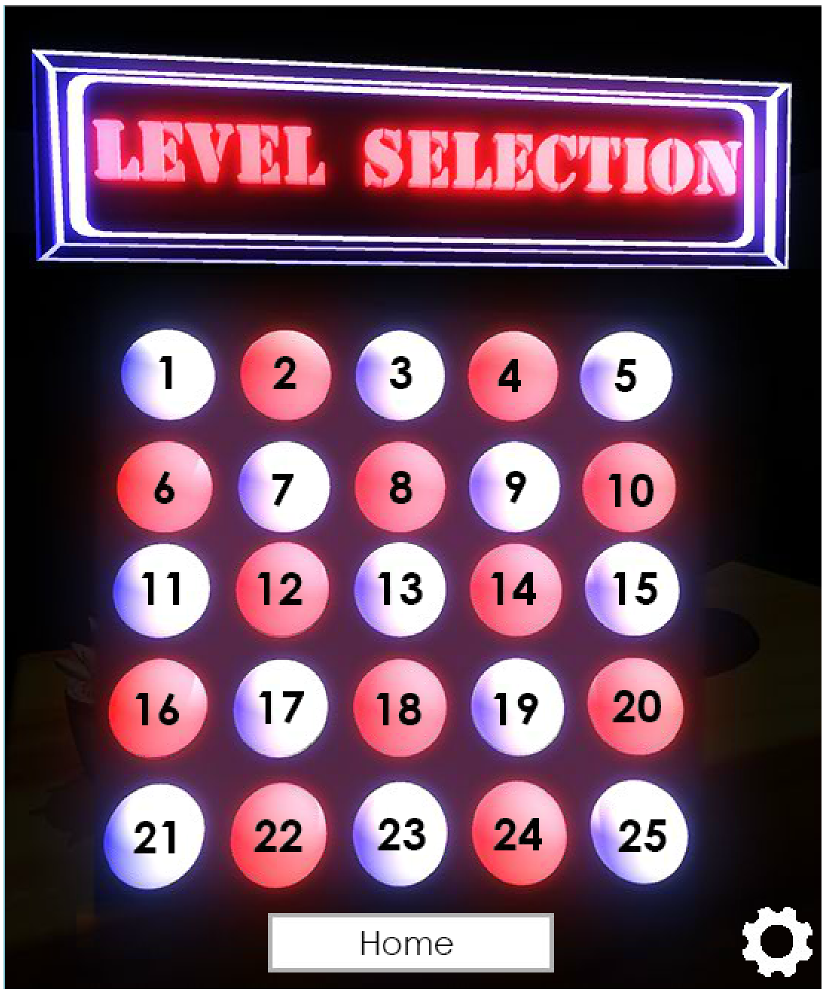


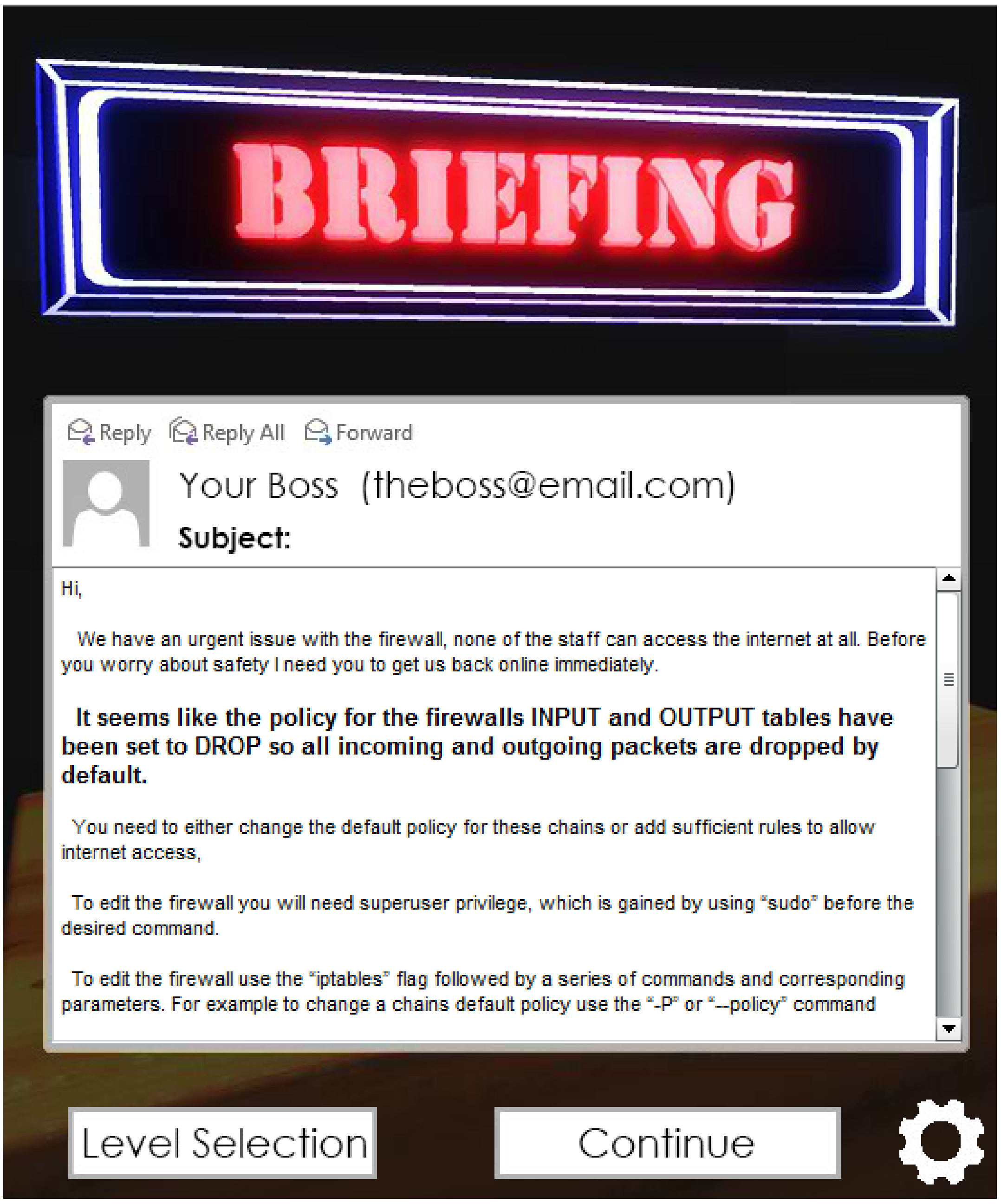
Figure 3.4: Level Selection Screen

During the design process, there were several clashing game requirements which made designing the levels selection screen more challenging than expected. For example, it was required that the game require as little prior knowledge as possible, while aiming to make the game widely accessible. By giving full access to all twenty-five levels at once, a player who chooses to immediately play level fifteen without completing the prior levels, may not have the required prior knowledge, leading to player frustration. Alternatively, by locking level access until previous levels are complete, advanced skill players will find themselves repeating redundant tasks, potentially deterring them from continuation of the game. Additionally, using the chosen implementation platform, which will be discussed later in greater detail, saving level progress across sessions is challenging, within the scope of this project. Applying locked levels to the game would make the game inaccessible and frustrating, as players would have to re-complete levels across sessions.

3.2.6 Level Briefing

Once a level is selected, the player receives an instructional briefing, as shown in Figure 3.5. The briefing describes the task that must be completed for the player to successfully complete the level. Once at this screen, there are three actions the player can take. After reading the description, the player may decide that they do not want to tackle this level for many reasons. Reasons may include, but are not limited to: previously completing this level, already possessing the knowledge being taught in this level, or not having the required knowledge to proceed. As such, the option to return to level selection is given. Another option is to change the difficulty setting. A player may decide to return to difficulty selection if they do not fully understand the instructions given, and would like an easier guidance description. Lastly, the continue button takes the player to the command line screen to complete the given task.

The instructions are posed as an email from the players boss. This type of role playing situation, in which the player assumes the roll as an employee being assigned tasks from his/her employer was implemented for several reasons. Primarily, one of the game goals was to emulate the experience of being a security system administrator. Having spoken to a professional system administrator, it was agreed that this idea achieved said goal.



Hi,

We have an urgent issue with the firewall, none of the staff can access the internet at all. Before you worry about safety I need you to get us back online immediately.

It seems like the policy for the firewalls INPUT and OUTPUT tables have been set to DROP so all incoming and outgoing packets are dropped by default.

You need to either change the default policy for these chains or add sufficient rules to allow internet access,

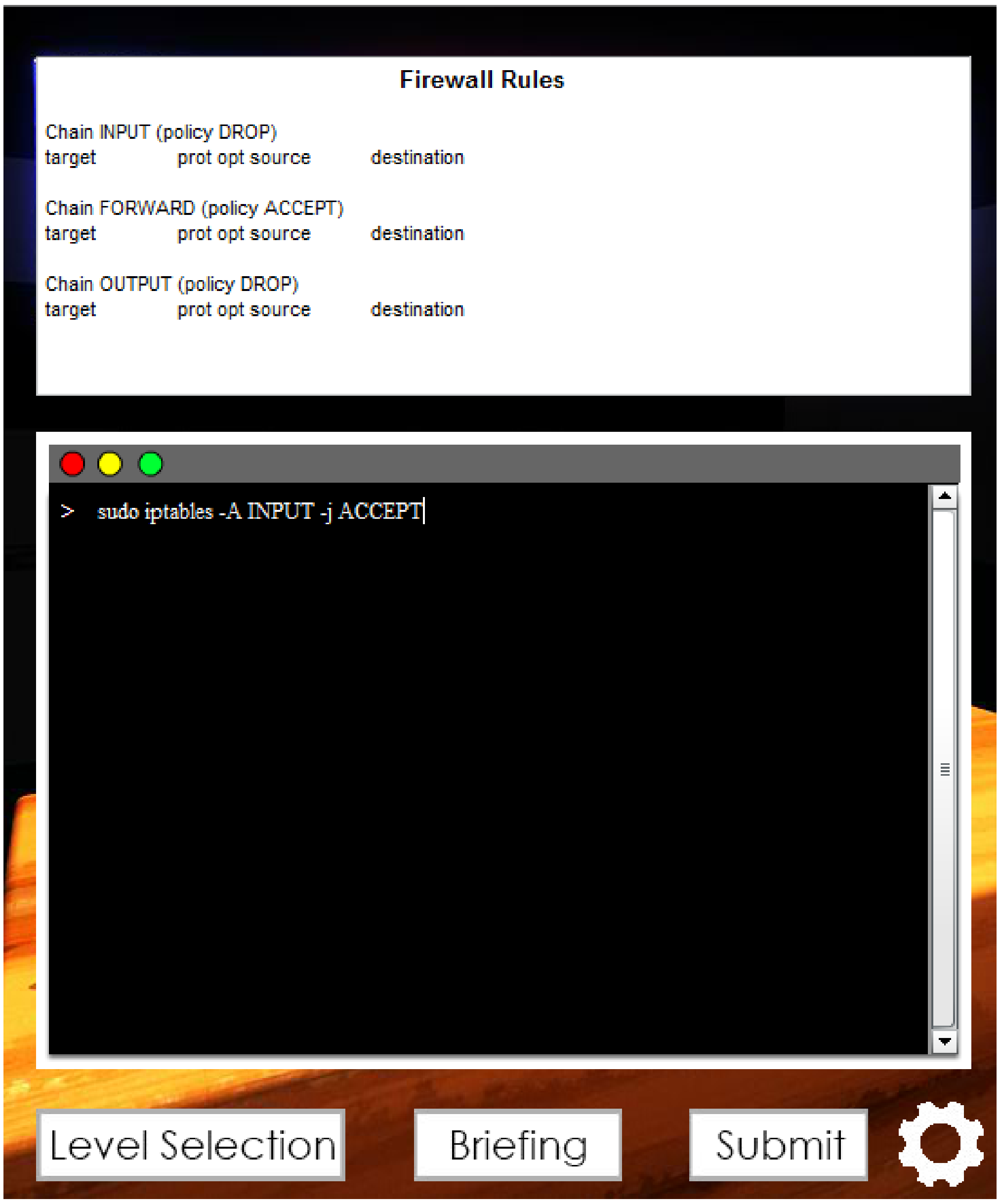
To edit the firewall you will need superuser privilege, which is gained by using "sudo" before the desired command.

To edit the firewall use the "iptables" flag followed by a series of commands and corresponding parameters. For example to change a chains default policy use the "-P" or "--policy" command

Figure 3.5: Level Briefing Screen

3.2.7 Command Line Screen

The next screen that follows is the Command Line Screen where players edit the firewall using the command line as shown in Figure 3.6 then submit the edited rule set for evaluation.


Firewall Rules

```
Chain INPUT (policy DROP)
target     prot opt source        destination

Chain FORWARD (policy ACCEPT)
target     prot opt source        destination

Chain OUTPUT (policy DROP)
target     prot opt source        destination
```

```
>  sudo iptables -A INPUT -j ACCEPT
```

Figure 3.6: Command Line Screen

This screen contains the following two design features which are discussed below:

1. In-Game Command Line Terminal
2. Firewall Rule Information Box

3.2.7.1 In-Game Command Line Terminal

The lower portion of the screen contains an in-game command line. A command line is a means of interacting with a computer program where the user issues commands to the program in the form of successive lines of text[12]. The in-game command line behaves similarly to a real command line, with the exception of only IPTables being supported. IPTables is a user space application program that allows a system administrator to configure the tables provided by the Linux kernel firewall (implemented as different netfilter modules) and chains and rules it stores[2]. The command line was implemented with use of regular expressions, a sequence of characters that define a search pattern[10], which parse players input. The input is then passed to control logic, which was built to mimic the IPTables application. The input is then analyzed, a response is generated, and if the command input is valid, the appropriate action is taken on the firewall.

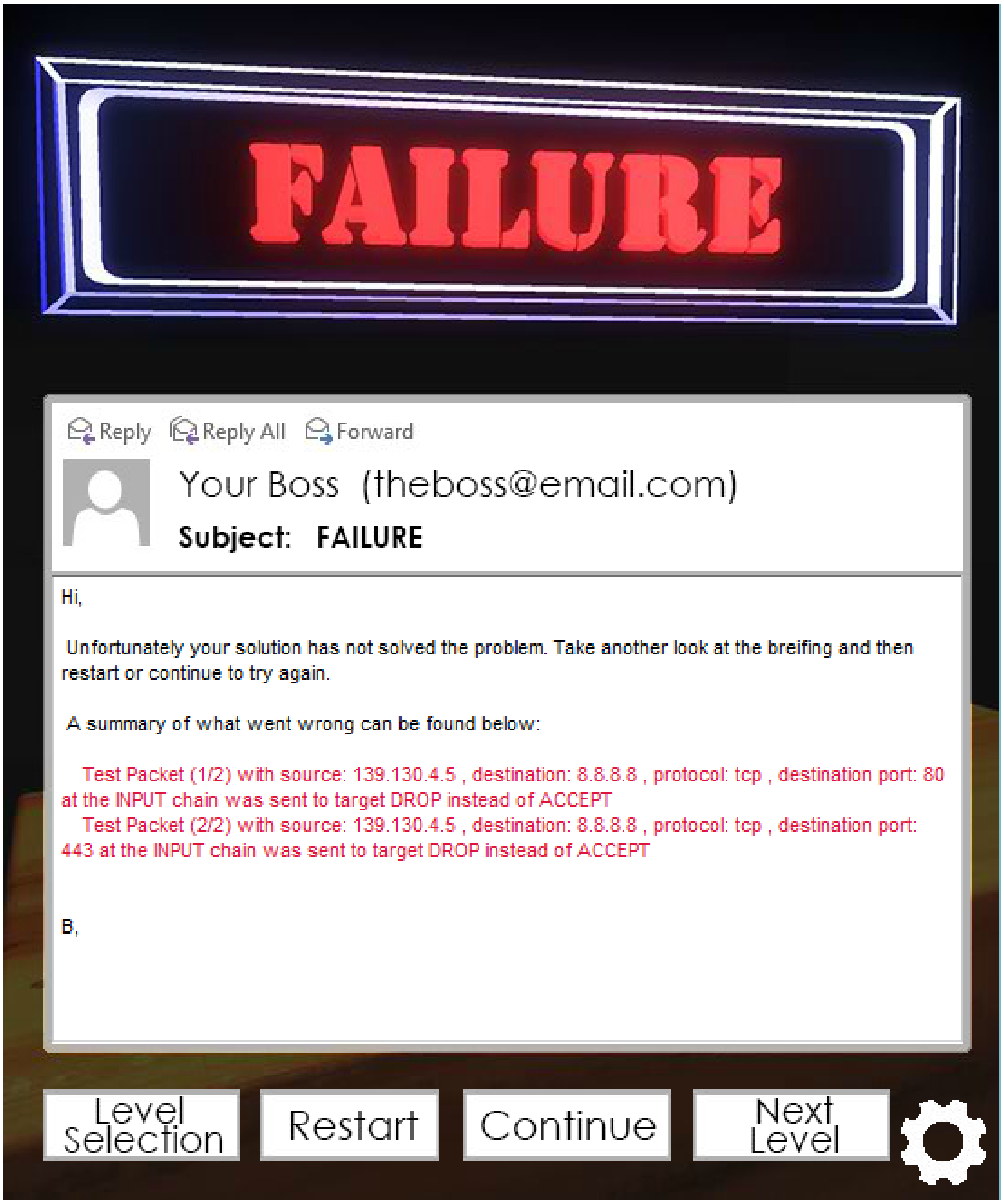
3.2.7.2 Firewall Rule Information Box

The upper portion of the screen is filled with an information box titled, Firewall Rules. The information displayed in the box is simply the list of current rules for the firewall.

Having the ability to see how the rule set is effected in real time as the player enters an IPTables command was deemed beneficial in learning for two reasons. First, the player knows immediately how their command affects the rule set. A player may not fully understand a command before using it, but with the opportunity to visually see its affect, comprehension is made easier. The second reason is efficiency. When learning IPTables from a regular command line, having to repeatedly list the rules manually wastes time.

3.2.8 Feedback Screen

After the player submits a rule set for level completion, they are presented with a feedback email. Dependant on the players submission, they will be presented with a success screen (Figure 3.8) or a failure screen (Figure 3.7). The purpose of this screen is to provide the player with a performance review. The player is presented with color coded text which describes test packets. Red text means the firewall incorrectly handled the test packet, while green text indicates correct handling of the test packet. The colors allow the player to easily identify what packets were incorrectly dealt with. The player can then examine the details of that packet (ie. source address or port number), which gives the information needed to fix the firewall. This automatic feedback system allows the player to see exactly how each packet is being handled, which allows for a greater understanding of their mistakes. As previously mentioned, one of the major issues with a tool like SEED labs, was the lack of user feedback, during the learning process. By providing an instantaneous response in this game, learning efficiency is improved.



Your Boss (theboss@email.com)

Subject: FAILURE

Hi,

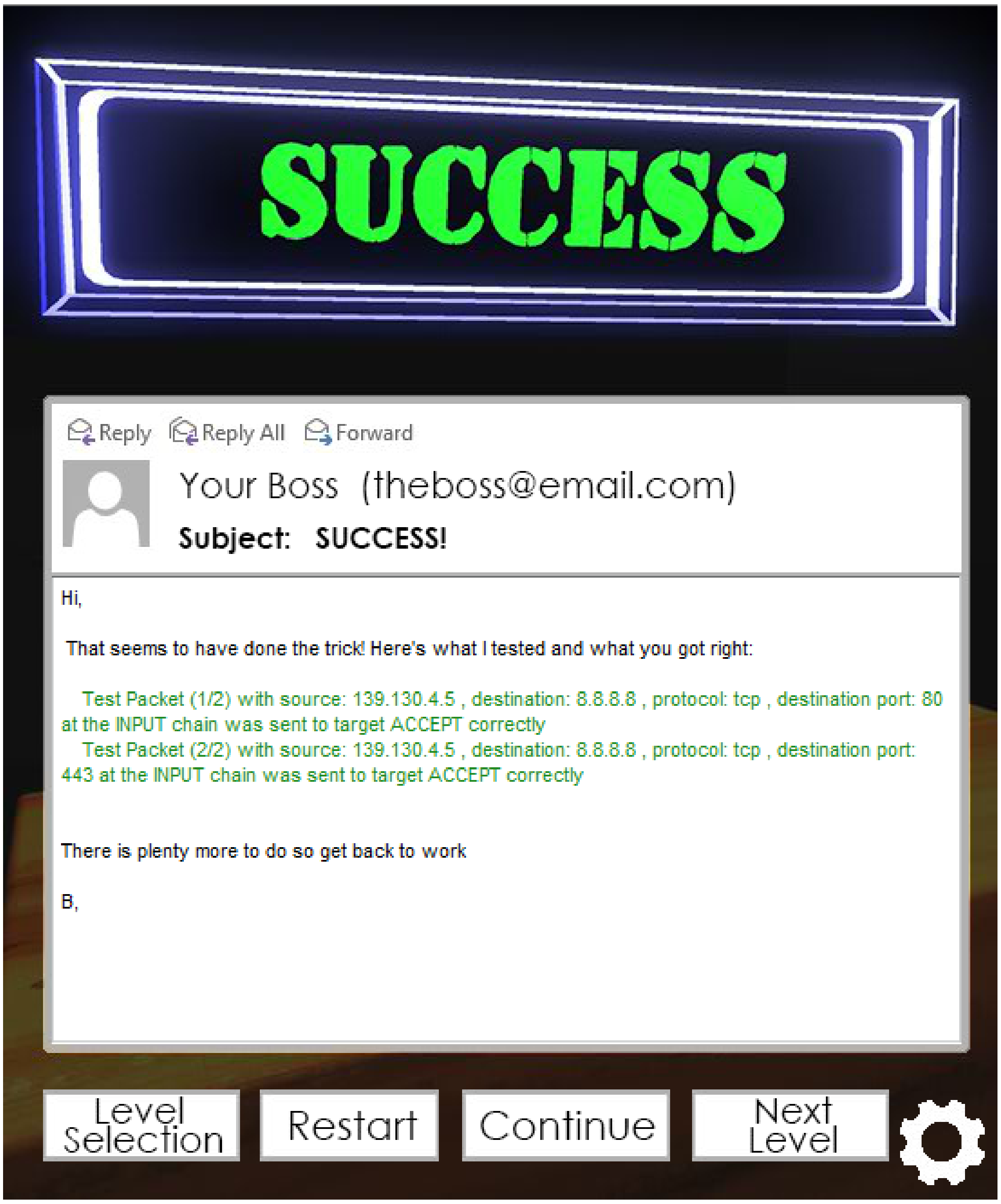
Unfortunately your solution has not solved the problem. Take another look at the breifing and then restart or continue to try again.

A summary of what went wrong can be found below:

Test Packet (1/2) with source: 139.130.4.5 , destination: 8.8.8.8 , protocol: tcp , destination port: 80 at the INPUT chain was sent to target DROP instead of ACCEPT
Test Packet (2/2) with source: 139.130.4.5 , destination: 8.8.8.8 , protocol: tcp , destination port: 443 at the INPUT chain was sent to target DROP instead of ACCEPT

B,

Figure 3.7: Feedback Screen - Failure



Your Boss (theboss@email.com)
Subject: SUCCESS!

Hi,

That seems to have done the trick! Here's what I tested and what you got right:

Test Packet (1/2) with source: 139.130.4.5 , destination: 8.8.8.8 , protocol: tcp , destination port: 80 at the INPUT chain was sent to target ACCEPT correctly
Test Packet (2/2) with source: 139.130.4.5 , destination: 8.8.8.8 , protocol: tcp , destination port: 443 at the INPUT chain was sent to target ACCEPT correctly

There is plenty more to do so get back to work

B,

Figure 3.8: Feedback Screen - Success

3.3 Implementation

The game was developed for Adobe Flash, a multimedia software platform used for production of animation, rich internet applications, desktop applications, mobile applications, and mobile games. Flash displays text, vector and raster graphics for animation, video games, and applications. It allows streaming of audio and video, and can capture mouse, keyboard, microphone, and camera input[24]. Flash was chosen as the development platform for this game because it has a wide user base and has minimal hardware requirements. It is a common chosen format for games, as it can easily be embedded in web pages and has over 400 million users[18].

3.3.1 Autodesk Maya

The graphics for this interface were created in Autodesk Maya, a 3D computer graphics software used to create interactive 3D applications, including video games, animated film and television, and visual effects[19].

Maya was chosen because it is an industry standard program for 3D modeling and animation, and enabled the artist to reach the visual target which this game aimed to convey. For this game in particular, it was decided that a semi-realistic workplace setting would be ideal, as using still life stock images would be unappealing, and traditionally painted or drawn images may not match thematically with this work. Using Maya, the set was modeled, textured, lit and rendered, and then complied in Adobe Photoshop, before being taken into Adobe Flash Professional CS6.

3.3.2 Adobe Flash Professional CS6

Adobe Flash Professional CS6 is an authoring tool which allows the user to create games, applications, and other content that responds to user interaction[16]. The studio is used to create vector graphics and design elements which are then positioned and defined on the Timeline and Stage. The master document produced is called an FLA file: **main.fla**, which when published, produces a small web format file (SWF): **main.swf**. This file is then embedded online with HTML and then anyone with Adobe Flash Player can play the game.

Adobe Flash Professional CS6 was the obvious choice when choosing a development studio, as it is the most up to date supported flash development tool.

The coding for the game was linked by Adobe Flash Professional CS6, but written in FlashDevelop.

3.3.3 FlashDevelop

FlashDevelop is an integrated development environment for the development of Adobe Flash websites, web applications, desktop applications, and video games. The resulting applications run in Adobe Flash Player. The primary purpose of FlashDevelop is enabling developers to edit, compile, debug, and publish a Flash ActionScript project[8].

The file produced is **main.as**, which contains all the game code in the object-orientated programming language ActionScript. This file is then used to in combination with the FLA file to publish the game.

Flash Develop was used in conjunction with Adobe Flash Professional CS6 because it has code completion, syntax highlighting, snippets, and other features which are not available in CS6. This allowed for a simpler and more efficient development process.

3.3.4 GitHub

The last implementation platform used for the development of this game was GitHub. GitHub is a web-based version control repository and internet hosting service[25]. Version control software is critical for most development as it allows you to keep track of code changes and backtrack if necessary. It additionally allows accessibility so you can work on different machines easily.

Chapter 4

Evaluation

The following chapter details the tests that were carried out to evaluate the game. The evaluation techniques used in this chapter utilized think-aloud protocol. The test subjects for the game were Undergraduate Informatics students from the University of Edinburgh, with varying background knowledge in firewall administration.

4.1 Aims

The main goal for this evaluation was to examine the interaction between the application and user, with the user providing feedback on the games usability. Through this evaluation, I aimed to understand the potential successes and failures of the game, when set against the games original goals and requirements. As with all research, the expectation was that from these usability tests, weaknesses in game design or implementation would be identified. In identifying problematic areas in the game, these shortcomings can be addressed and corrected in an iterative manner.

Due to the scope of this research, several problems have become apparent. Quantitative proof of the improvement in teaching capacity between this game and other existing tools infeasible. Even with a large budget, sufficient testing samples and no time constraints, results would still be inconclusive, as the tools each have different teaching objectives. Instead, this research has outlines why it is believed that this tool is a conceptual improvement over existing tools, and evaluates the usability of these concepts. Additionally, this section also addresses the fact that due to time constraints, some identified improvement plans must be categorized as Future Work.

4.2 Method

This section details the methods carried out for this study and provides explanation for the chosen methods. The Think-aloud protocol was used as the method of evaluation.

4.2.1 Think aloud protocol

Think-aloud is a protocol used to gather data in usability testing for product design and development. Think-aloud protocols involve participants thinking aloud as they are performing a set of specified tasks for the purpose of research. Participants are asked to say whatever comes into their mind as they complete the task. Test sessions are audio recorded which when analyzed, gives observers insight into the participants cognitive processes[17].

In his publication, *Thinking Aloud: The #1 Usability Tool*[20] Jakob Nielson describes the benefits of this evaluation technique. He notes that when you hear a participants misconceptions, the usually turn into actionable redesign recommendations. In addition, you learn why users interpret parts of the user interface incorrectly and why they find other user interfaces easier to use. Nielson also explains five other benefits which have been summarized as follows:

1. **Cheap.** No special equipment is needed and it doesn't take very long to collect data. It also does not require a large sample size of testers to gain meaningful and reliable results.
2. **Robust.** In contrast to quantitative usability studies, which can be filled with methodology problems in which the smallest mistake can make the findings misleading, think-aloud will still produce reasonably good findings even from a poorly run study. This is particularly relevant for this study, as I had little to no prior experiencing running an evaluation study .
3. **Flexible.** You can use the method at any stage in the development life cycle, from early paper prototypes to fully implemented, running systems. It can be used to evaluate any type of user interface with any form of technology which is excellent for Agile projects like this one in which testing is done early and often.
4. **Convincing.** Getting direct exposure to what users think about the project is the best way to make improving usability a priority.
5. **Easy to learn.** Learning how to instruct a basic think aloud evaluation is simple and does not require an experienced usability consultant professional.

For these reasons, the think-aloud protocol was the most suitable method for this evaluation, as it can be used to specifically target achieving evaluation aims.

4.2.2 Participants

As previously mentioned, think-aloud protocol does not require a large sample size of testers to acquire useful results. Nine potential participants were contacted, with five subsequently taking part. All participants were students from The University of Edinburgh, all who are currently studying to obtain an Undergraduate Degree with the School of Informatics. Three participants were third year students, while the remaining two were in their fourth year. Two of the third year students and both of the fourth year students had previously taken a computer security course which covered relevant firewall skills.

4.2.3 Procedure

Before conducting a formal evaluation, an ethics form was submitted to the School of Informatics, in accordance with the School Ethics Code[14]. Each participant in the study was seated at a desktop computer in an isolated room. An introductory script was read aloud, explaining the details of the testing. This script was adapted from an example script from the book *Dont Make Me Think: A Common Sense Approach to Web Usability*[15] by Steve Krug. The introductory script informs the participant about the procedure for thinking out loud, as well as providing the following assurances:

1. They can stop the testing procedure at any time and do not have to stay throughout the testing period.
2. They are being audio recorded but have total authority on if their words are published.
3. The game is being tested, not their ability or knowledge.
4. Questions may not be answered during the testing period, as the test aims to see how people do when using the game without supervision. However, upon completion of the test, any questions will be answered, to the best of the facilitators ability.

After providing these reassurances to participants, the actual task description is given, and can be described as follows: Explore and complete the game levels as you see fit for twenty minutes. Krug also suggests an additional light questioning period of approximately five to ten minutes. After the twenty minute testing period, participants were asked several informal, unrecorded questions to reveal additional information that the participants had to offer regarding the strengths and weaknesses of the game. After thanking each participant for their contribution to this study, participants were dismissed. After each session, the research was compiled into notes for incorporation into this dissertation.

4.3 Results

To present the results of this research clearly and concisely, the discussion is split into strengths and weaknesses of the game, as uncovered through testing.

4.3.1 Strengths

The game was well received, based on study participants usability test feedback. This section will highlight two design aspects that worked well and were relevant to discussion, including:

1. Feedback system
2. Graphical Interface

4.3.1.1 Feedback system

Throughout the think-aloud evaluation, several play patterns surfaced amongst participants. It becomes apparent through observing this pattern why the instant feedback system was a crucial component in adding to the successes of the game's design:

1. The player quickly skim reads the briefing email, not fully understanding the task or given IPTables command information.
2. The player enters the command discussed in the briefing email with varying correctness.
3. The player submits the firewall rule set. After level one, most participants got it incorrect on first try.
4. The player studies the feedback, focusing on packets which where incorrectly dealt with. The player identifies what they got incorrect and begin to understand the IPTables commands and given tasks.
5. The player normally returns back to the briefing with the new understanding. They confirm their new understanding and are reminded of the IPTables commands needed.
6. The player enters the correct IPTables commands and submits the correct rule set.

Several participants discussed why they chose to skim the email, then proceed directly to the command line. The general consensus was that the text based email, filled with firewall jargon, was harder to comprehend than when the same concepts were shown in a hands-on manner. This issue could be perceived as a failure in how the email relays firewall information, however, when asked about this, during the discussion previously noted, one of the participants disagreed. They explained that it was their preference to attempt a task without fully understanding all concepts prior and learn practically. They thought that this feedback system allowed for this method of learning. Many of the other participants had similar sentiments. They enjoyed being able complete the levels in a problem solving manner, which was again, enabled by the feedback system.

The feedback system allows the player to see how an added rule affects the firewall, with many noting that this was a major component in their increasing understanding of firewall systems. From the recorded sessions, the following three quotes were chosen to depict this:

It's nice that it tells me where I went wrong.

I don't think I could complete the level without being told what I'm missing ... good for me.

The boss email is a little confusing but since I can just see what should happen with the packets at this screen, then it makes sense what I should do.

4.3.1.2 Graphical Interface

The graphical interface was well received and received generally good reviews. In the words of several participants:

I like this plant I wish it was on my desk. - In reference to the start screen

Every time a red line goes green I feel like I'm becoming a better coder. - In reference to the feedback test packet text

This screen is so bright and satisfying when you finally get it - In reference to the success feedback screen.

Furthermore, through the probing discussion, participants noted that they especially liked the bright colours, the artwork, and the use of the email screen. One participant commented that the information presentation was successful because of the email screen design, which was thematically fitting. Another participant described the colors and art style as the perfect distraction from the boredom of having to learn about firewalls.

4.3.2 Weaknesses

From the usability test, the following areas were identified as needing improvement:

1. Text Copying
2. Level Selection Screen
3. Lack of an Introduction

The following sections describe problems with correlating participant quotes, followed by proposed solutions.

4.3.2.1 Text Copying

One problem, discovered through the first two evaluation sessions, was that the email text was selectable and could therefore be used to copy and paste into the command line terminal. Both of the first two participants quickly realized that the briefing email associated with the easy instructions contained the command needed to successfully complete the level:

Can't I just copy and paste this?

Is this allowed? - In reference to the act of copying and pasting the commands

Prior to the commencement of evaluation testing, it was known that the text was selectable. The decision to leave the text as selectable was made, as it was unknown whether this would impede learning, or aid it as a result of efficiency. The selectable text was deemed a design failure after the first two think-aloud tests for two reasons. By copying and pasting the command, the player was not learning the syntax or understanding the purpose of the command. This became apparent when the participant was unable to complete the subsequent task, which required an understanding of the previous task. After realizing they did not possess the knowledge to proceed, participants would return to the previous level and recopy the text, once again not understanding or learning the command, or how it works. Additionally, the command examples given in the briefing emails are examples, not actual solutions to the given problems. As a result, participants who used copy and paste became increasingly confused and learned inefficiently.

This section has described a design weakness and the way in which it impedes the games specified goals and requirements. Because this game is an agile project, it was possible to solve this issue before the final three evaluations took place. The text was made unselectable by changing the properties of the text object in ActionScript. In the remaining three think-aloud evaluations, retained knowledge from each level was greatly improved. These participants were more successful in recalling the required commands than in the first testing group.

4.3.2.2 Level Selection Screen

One major design issue identified by participants was the level selection screen. Think-aloud observations revealed the following statements:

What level did I just do ... I'm just going to pick one at random.

I should be able to cross out the ones I've done.

One participant said:

I already remember how to do this stuff when does new stuff come up?

Then the participant picked level seventeen and after attempting it:

I can't do this, I should have stayed at the beginning levels.

After reading a level briefing, another participant wished to revisit a previous level, but did so unsuccessfully, picking the wrong one twice and noting:

These should be named.

This selection of quotes display several obvious problems with the level selection screen design. While keeping all levels unlocked solves issues of varying background knowledge and general accessibility, it introduced a new problem of unclarity. This seemed to be a minor issue, as most participants played the levels in numeric order, using the next button to proceed. One solution to this problem might be to add a small text box during the introductory stage, suggesting that the players proceed in numerical order. Another solution would be to introduce additional graphics which identify previously completed levels.

In addition to unclear completion tracking on the level selection screen, it was also noted that the content of each level is not easily identifiable. The player is unable to identify what topics a level will include without selecting a level which is highly inefficient. Discussions with participants and general feedback from others who tried the game outside of formal testing echoed this design weakness. Both feedback and additional research suggests a solution which involves grouping levels into sections, with each section labeled with the topic being covered.

4.3.2.3 Lack of an Introduction

The final item identified as needing significant improvement was the lack of an introduction to the game. One player compared the first few minutes of game play to plunging into the deep end of a pool. This participant had no previous experience in Computer Security and found much of the basic firewall lingo confusing. Additionally, even players with minor experience were not always familiar with the vocabulary used. The lack of a clear introduction also created some confusion as to how the game should be played. While participants thought that the boss and employer setting was a well designed game environment, this setup is not clearly outlined at the start of the game. In summary, the game should not have assumed that participants would have prior knowledge of the necessary vocabulary. It should also have provided a better explanation of the game structure and roles, including role of the boss, employer, and command line. The absence of this feature was seen as a design failure because the requirements for this game specified that little prior knowledge was necessary, but it quickly became apparent that words like chains and policy were ambiguous and without prior knowledge, confusing. One participant noted:

I don't know what a chain is.

I'm currently reading this email ... my boss is telling me to change the policy of the firewall ... I know what policy means but not what it means for a firewall.

Although participants found the initial terminology confusing, when questioned about their frustrations after the session, the majority of participants explained that by the end of their testing period, they felt they had a good understanding of the once unfamiliar terms. Despite the successful learning curve of most participants, one participant admitted that they still had no idea what several of the words meant. Another participant expressed the opinion that the use of the email screen was relatively intuitive, but could have been better explained at the beginning of the game.

4.4 Design Goals and Requirements

This section concludes the evaluation chapter by revisiting the original game goals and requirements, and assessing how well the requirements were fulfilled as exemplified by the evaluation results.

4.4.1 Design Requirements

Chapter three listed the game goals and requirements. This section assesses whether or not the aforementioned requirements were met.

Requirement 1: The game should simulate real system Firewall administration.

The trade-off between complex, real system administration and game requirements, such as little required prior knowledge, was described and justified in Chapter Three. The target audience is not made up of trained professionals, but rather, undergraduate students which contributes to the perceived oversimplification and semi-unrealistic manner in which the game depicts firewall system administration. It can also be argued that despite the simplifying of some of the complexities of firewall administration in the game, because IPTables is used by professionals internationally for real system administration[13], enabling accessibility through simplification of system administration still allows for the above game design requirement to be met.

Requirement 2: The game should teach skills for a user-space application that allows system administration outside of the game.

The game teaches skills for the application IPTables. It is clear that this requirement was successfully met, as proved by observations of the participants recalling the knowledge they had just learned for application to incomplete levels. For example, during the testing period, participants with varying levels of prior knowledge demonstrated the ability to reuse commands that were taught in previous levels.

Requirement 3: The game should be usable and designed for undergraduate Informatics students.

All participants were undergraduate Informatics students, and all managed to use the game to learn administration skills. However, as previously noted, the game could be improved through the addition of an introduction section, increasing accessibility for those with absolutely no computer security background.

Requirement 4: The game should provide real time feedback for correct and incorrect answers.

This requirement was described in Section 4.3.1.1, explaining why this was one of the most successful features of the game.

Requirement 5: The game should be designed to maximize it's potential audience.

To describe this requirement, three sub-requirements were outlined. In order to assess the fulfillment of this requirement, each of the three sub-requirements need to be appraised:

The game should be designed to be widely accessible

This was achieved through the design decision to use Flash, one of the most commonly used and easily accessible multimedia software platforms.

The game should require as little prior knowledge as possible.

As previously discussed, this requirement is considered met, with the expectation of improvement through further iterations of the game.

The game should have few hardware requirements.

Nearly all modern day computers have the hardware capabilities to run an Adobe Flash application, it would be a challenge to find a computer that didn't.

Requirement 6: The game should be colorful and have appealing art.

As previously discussed, the games graphical design was well received by participants who thought it was one of the game's main strengths.

Requirement 7: The game should have minimal setup time.

The only setup required is to install Adobe Flash Player on the browser which the user wishes to play. Most users will find that they already have this installed for other daily use applications.

Requirement 8: Game mechanics should not take a long time to learn.

The command line is a well known concept for most Informatics students and the user interface is designed in such a way that game mechanics are intuitive. The evaluation sessions confirmed this, as participants took mere minutes before entering commands.

Requirement 9: The game should not depend on long play sessions

Throughout the 20 minute gameplay sessions, it is estimated that the average participant completed three to four levels. This demonstrates that a session in this game does not need to be a large time commitment.

4.4.2 Design Goals

This section assesses whether or not the game's design goals were met.

Goal 1: Improve upon existing tools used to teach firewall administration skills

This evaluation chapter has described numerous design features and why they are believed to be improvements while also using the evaluation results to further show the successes of these design features. Some of these features include:

1. Instant Feedback
2. Short Play Sessions
3. Colorful and Appealing Art
4. Widely Accessible
5. Little Required Prior Knowledge
6. Minimal Setup Time

The success of these design features suggest that the game is a conceptual improvement upon existing tools. Additionally, if the game continues to be iteratively developed, it is believed that quantitative improvement upon existing tools could be proven. As such, this design goal is regarded as achieved.

Goal 2: Emulate the experience of being a security system administrator in charge of manging Firewall policy rules.

Chapters Three and Four have described and justified the chosen balance between complexity and usability of the game in reference to this goal. The game has demonstrated the ability to teach skills applicable for system administration and as a result it can be concluded that this game does emulate the experience of being a security system administrator.

Chapter 5

Conclusion

This chapter summarizes the research done for the purpose of this dissertation and delivers the conclusions made, based on the findings of the game evaluation. The overarching research goals and themes will be addressed, and possible future work for the research and development of the game is given. The main contributions of this work is presented and a summary of outcomes are given.

5.1 Research Goals

The main research goal for this project was:

To determine the potential for improvement in existing tools designed to teach firewall administration skills.

This goal was achieved by researching three of the most popular existing technologies: SEED labs, Elevation of Privilege and Control-Alt-Hack. Through the critical review of these three technologies, the following areas needing improvement were identified:

1. Takes too long to use and to learn how to use the tool.
2. Takes too long to set up the tool.
3. Lack of technical educational value.
4. Lack of automated feedback to aid in learning.
5. Lack of appealing art to retain user attention.

These where chosen as the most important, recurring themes, with justification for informing the design of a new teaching tool, which forms the secondary research goal:

Design an educational game to effectively teach firewall administration skills.

To prove whether or not this research goal was achieved the following questions must be addressed:

1. Does the game support the development of practical firewall administration skills and an understanding of the theory behind firewalls?
2. Does the accessibility of an online learning tool provide relevant benefits in comparison to other learning tools?

The First research question has been discussed at length in Chapter Four and is regarded as achieved, through the use of the evaluation results as justification. This research has demonstrated the ways in which design concepts in this new educational tool provide relevant skill development. The improved design concepts allow a more efficient, accessible, and practical way to learn firewall administration skills. These design improvements are an important part of contributing to the growth of standardized teaching methods, responding to the future demand for skilled security workers.

5.2 Future Work

While some of the evaluation results revealed weaknesses within the game, solutions to these weaknesses were posed and implemented when possible. However, due to the time constraints, not all solutions could be implemented. The following two undeveloped solutions are intended to be seen as plans for future work. These solutions include:

1. Improve the design for the Level Selection screen.
2. Add an introductory section to introduce basic firewall vocabulary and set the scene of the game.

These two design updates are seen as applicable future work due to the Agile developmental methodology used for the project, which lends itself to this iterative improvement process.

Additionally, a formal evaluation of the quantitative teaching value of this game would provide verifiable evidence of the conceptual improvement shown in this dissertation. Finally, this educational game covers a narrow area within Computer Security, which as a subject is vastly underrepresented in the current education system. Future work could include expanding concepts showcased in this dissertation to develop other games, or an extension to this game which covers more Computer Security teaching topics, such as Secure Programming.

5.3 Discussion

This research has explored the potential improvement in existing teaching tools for firewall administration skills. This process first involved doing a background investigation into existing literature and technology to find potential areas of improvement. A design concept was then created from the drafted game goals and requirements. The first version of the game was then developed, and later evaluated using think-aloud protocol and iteratively improved, based on evaluation results. This dissertation offers a number of valuable observations that could be used by future researchers. This information includes:

1. A role playing system administrator game appears to be an effective way to teach firewall system administration skills.
2. In such a game, having a feedback system aids in the efficiency and engagement of learning.
3. Some learning topics in Computer Security struggle to retain student attention. A well designed graphical interface, as well as minimal set-up time helps to keep educational learning tools interesting.
4. To improve current learning tools, designs need to include maximum flexibility and accessibility. Because people have vastly varying background knowledge and hardware/software capabilities, the tool must be kept adaptive to reach the largest potential audience.

Computer security is one of the most serious economic and communal challenges faced by modern society, yet finding skilled security workers is becoming increasingly difficult. Teaching practices for computer security have been identified as severely lacking and this research hopes to contribute to the ongoing effort to correct this.

Bibliography

[1] AGILE ALLIANCE. What is agile software development? 2013.

[2] PABLO NEIRA AYUSO. The netfilter.org iptables project. 2017.

[3] ISTQB EXAM CERTIFICATION. What is agile model? advantages, disadvantages and when to use it. 2012.

[4] WENLIAN DU and RONGHUA WANG. Seed: A suite of instructional laboratories for computer security education. 2007.

[5] WENLIANG DU. Hands-on labs for security education. `http://www.cis.syr.edu/~wedu/seed/index.html`. Accessed: 2017-04-1.

[6] WENLIANG DU. User manual of the pre-built ubuntu 12.04 virtual machine. 2014.

[7] K. EVANS and F. REEDER. A human capital crisis in cybersecurity. 2010.

[8] FLASHDEVELOP. Main page. 2011.

[9] MARK GONDREE and ZACHARY N. J. PETERSON. This is not a game: Early observations on using alternate reality games for teaching security concepts to first-year undergraduates. 2015.

[10] JAN GOYVAERTS. What regular expressions are exactly. 2003.

[11] BOOZ ALLEN HAMILTON. Cyber in-security: Strengthening the federal cybersecurity workforce. 2009.

[12] WINE HQ. Text mode programs(cui: Console user interface). 2013.

[13] JEFF T. PARKER JESSEY BULLOCK. Wireshark for security professionals - using wireshark and the metasploit framework. 2017.

[14] FRANK KELLER. Research ethics procedure. 2017.

[15] STEVE KRUG. Dont make me think: A common sense approach to web usability. 2013.

[16] FREDERIC LARDINOIS. Adobe launches animate cc, previously known as flash professional. 2016.

[17] C.H. LEWIS. Using the thinking aloud method in cognitive interface design. 1982.

[18] COLIN MOOCK. Actionscript for flash mx - the definitive guide. 2012.

[19] MICHEAL J MUWANGUZI. Maya 2011. 2011.

[20] JAKOB NIELSON. Thinking aloud: the #1 usability tool. 2012.

[21] D SCHULTHEISS. Long-term motivations to play mmogs: A longitudinal study on motivations, experience and behavior. 2007.

[22] ADAM SHOSTACK. Elevation of privilege: Drawing developers into threat modeling. 2012.

[23] ADAM SHOSTACK TAMARA DENNING, ADAM LERNER and TADAYOSHI KOHNO. Control-alt-hack: The design and evaluation of a card game for computer security awareness and education. 2013.

[24] CHRSISTINA WARREN. The life, death and rebirth of adobe flash. 2012.

[25] ALEX WILLIAMS. Github pours energies into enterprise raises 100 million from power vc andreessen horowitz. 2012.