Permission Impossible: Teaching Firewall Configuration in a Game Environment

Sibylle Sehl
University of Edinburgh
School of Informatics
Edinburgh, UK
sibylle.sehl@googlemail.com

Kami Vaniea
University of Edinburgh
School of Informatics
Edinburgh, UK
kvaniea@inf.ed.ac.uk

***Abstract*—Topics in Computer Security, such as firewalls, can seem inaccessible or very difficult to beginners. That perceived inaccessibility is a serious problem at a time when countries like the United Kingdom are facing a shortage of skilled computer security professionals and consequently need more students to consider careers in the area. This project aims to bridge this gap by providing an engaging and friendly game-like environment for both young computer scientists and the general public to learn about firewalls in a fun and educational way. In this work, we present the design of Permission Impossible, an online game designed to teach people both with and without a computer science background about firewalls. We discuss an iterative design process where we consulted with firewall administrators, evaluated an existing networking board game, and created our own online game. Early evaluations suggest that the game is accessible, and that people from multiple backgrounds can use it to learn about how firewall rules are constructed and how a firewall operates.**

***Keywords*—component, formatting, style, styling, insert**

I. Introduction

Computer security is a growing area, and one necessary for the future safety of countries like the United Kingdom (UK). Currently there are insufficient students graduating with cyber security experience to fill all the available jobs, while at the same time, the public's interest in security and privacy topics has never been higher. In order to increase the number of people choosing degrees in both Computer Science and Computer Security, it is necessary to attract more undergraduates and high-school students to such disciplines early in their educational development [16].

The new UK National Cyber Security Center (NCSC) has also identified the pipeline problem as one of their largest blockers to getting good applicants to fill their own vacancies. To help solve it, they are currently engaging in large scale efforts to involve young students, particularly girls, in cyber security camps to encourage interest in advance of selection of secondary school course options [6]. Yet when students return home from the camps there are very few self-study tools aimed at their level. As one of the author's female students put it: "I beat all the guys at a coding camp, then I came home and tried to find something I could do online to prove to myself that it wasn't a fluke, but I couldn't find anything I could do outside of capture the flag challenges which are all competitive and require existing knowledge."

In this work, we focus on the building of such a middle ground tool aimed at people who are interested in the details of computer security, but do not yet have the experience to participate in a public capture the flag type competition. We decided to start with the topic of firewalls. For those not familiar with the term, a firewall regulates traffic on a computer network to ensure that only traffic matching a set of rules may pass between computers. We selected firewalls for several reasons. First, the term "firewall" is reasonably well known to the general public as something that has to do with computer security. The term now appears regularly in mass media such as major films, though it is rarely explained. The result is that while the general public likely does not understand what a firewall is, they do recognize the term and are likely curious about it. Second, firewalls are conceptually simplistic technology compared to say anti-virus systems, which may use machine learning and signatures. Firewalls simply direct or block traffic based on a set of pattern matching rules. The last reason comes from a personal frustration of the authors that teaching firewalls requires an overhead of Virtual Machine (VM) knowledge before the concept of firewalls can even be approached. The VM requirements effectively limit firewall education to situations where a VM lab environment can be pre-setup and tested; therefore preventing walk-up-and-try firewall education.

In this work we take an iterative design approach to the problem of creating such an engaging tool. To understand the problem we conducted two expert interviews with system administrators who currently administer firewalls professionally as well as an educational games expert. We also conducted an evaluation session with an existing security board game called [d0x3d!] followed by a short focus group discussion to generate ideas around security game education. The outcome of these explorations was the decision to build a video game which would be very interactive and support both scaffolded learning and fast feedback. Our experts highlighted how vital a conceptual level understanding of firewall function is for them. They have existing tools that will identify syntax errors, but is-

Research funded in part by a Google Research Award.

European Workshop on Usable Security (EuroUSEC) 2018
23 April 2018, London, England
ISBN 1-1891562-54-1
https://dx.doi.org/10.14722/eurousec.2018.23006

sues like rule order require a strong conceptual understanding. Similarly, our focus group identified the importance of both learning firewall terminology and also applying it in such a way that the learner forms a strong mental model of how the firewall works internally.

We therefore decided to build an online video game entitled Permission Impossible. The game was designed to be highly accessible to a general audience; while at the same time not shying away from technical firewall topics such as chains, packets, and ports. It is comprised of a combination of instructional screens which explain basic concepts and a drag-and-snap style interface which allows easy firewall configuration, paired with animations which show the results of the configurations to further assist the player in forming a good mental model of the system.

We conducted an initial evaluation of the game with five participants who completed a pre-test on firewall concepts, played the game, completed the System Usability Scale (SUS) [5], and finally a post-test which was nearly identical to the pre-test. We also evaluated with a further 5 participants who only completed the SUS.

All participants showed some improvement in firewall knowledge between pre- and post-tests. The SUS also resulted in a 88.25 average score which shows a high level of usability. The game was quite accessible and fun for students with a range of experience, including those with computer security experience and those who have never programmed before. We also observed that even those with prior computer security experience found some of the levels in the game challenging and felt that they had learned something from it.

While we have not yet had the opportunity to test the game with people younger than 18, comments from the game players suggest that they feel their younger siblings would likely enjoy it and find it educational.

II. Background

A. Firewalls

A firewall is a device, software, arrangement, or equipment that limits network access, be it a software layer or a physical box [7]. While there are many types of firewalls, all of them work by examining the traffic passing across them and applying a set of rules to the traffic to determine if each packet will be allowed through (accept) or discarded (drop) [17].

Writing firewall rulesets correctly is easier said than done and many problems can potentially occur. Common issues include: rule ordering errors, verbose and hard to understand rules, and keeping the number of rules to a minimum [7], [17]. Wool et al. looked at errors in deployed firewall rulesets, they found multiple common errors spanning from allowing "any service" inbound and outbound, insecure access such as unencrypted access to the firewall and using implicit rules with regard to TCP, UDP and ICMP [30]. In a follow-on paper they further concluded that the errors cannot be avoided by using more advanced pieces of software, since many errors occur due to user-specific rules [31]. Even experienced system administrators, in charge of managing a firewall, still struggle to grasp key concepts, leading to even greater misunderstandings in a bigger organizational context [19].

In order to narrow the scope of this work, we focus on stateful software firewalls, in particular a firewall implementation called *iptables*. We selected *iptables* partially because it ships with many versions of Linux as well as in common consumer devices like home routers. It is sometimes suggested that *iptables* is a good introduction to managing your own firewall due to its simplicity, though it is generally not the tool of choice at an enterprise level.

As the name *iptables* suggests, *iptables* contain tables, which in turn contain chains, which contain rules. We focus on the *filter* table which deals with the traditional firewall behavior of filtering packets. *Iptables* expresses firewall restrictions with rules which are categorized into *chains* – ordered rule sets applied to a particular type of traffic. There are three system chains: *input*, *output* and *forward*; however, the user also has the ability to specify his own chains [2]. The *forward* chain is used to route packets that are not destined for or originating from the local host, whereas the *input* and *output* chains are related to incoming and outgoing network traffic respectively.

Firewall rules are traversed in order. When a packet arrives at the firewall in the *filter* table the packet is matched against the first rule: if it matches the rule, a target such as *accept*, *drop* or *reject* determines what happens next. If the packet does not match the rule, then the next rule is evaluated. Ordering these rules in a sensible way is of utmost importance and is an error that is encountered by many system administrators [30]. An example of a common rule ordering error is to write an overly general *accept* rule before a specific *drop* rule, effectively guaranteeing that the *drop* rule will never be encountered.

The below expression is an example of a new rule command in *iptables*:

```
iptables -A INPUT -i eth0 -p tcp --dport 80
-m state --state NEW, ESTABLISHED -j ACCEPT
```

The -A relates to specifying whether the rule is added, inserted or deleted. Add (-A) means that the rule is added at the end of the rule set as opposed to being inserted elsewhere. The capitalized INPUT is the chain into which the rule is inserted, the -i is the interface (wired network interface eth0) and the -p define the protocol (tcp) the rule applies to. The –dport 80 is the destination port 80. "State" refers to the fact that iptables is stateful and tracks which connection each packet belongs to. In this case, the rule refers to new connections (where the packet is a new packet and not part of a previous connection) and established connections. The target of ACCEPT at the end specifies that packets that match the rule should be let through the firewall.

B. Firewall usability

The general public has heard of firewalls and associates them with security, but has little understanding of what they are or their function [14], [24]. When it was first released in 2000 ZoneAlarm skyrocketed in popularity partially due to the relatively user-friendly nature of its interface [4]. Yet, subsequent studies find that users do not know what a firewall is for, only that they should install one. A study be Ion et al. found that when asked what advice they would give their friend on how to stay safe online, 17% of users recommended installing a firewall as compared to just 3% of security experts [18].

Raja et al. investigated the mental models of end users regarding personal firewalls [24], they find that users have a poor understanding of what a firewall does and consequently experience a mismatch between what the firewall does and what they needed it to do. Their study concluded with the recommendation that firewalls be integrated into other security technology because what users really needed was an all-in-one solution and were not getting what they needed by installing many different technologies.

The goal of this work is to look at firewalls as a more general technology from an education and interest building standpoint rather than to instruct users in how to configure their personal firewalls. Based on the above research, it is interesting to note that end-users generally have minimal understanding of what a firewall's purpose us and this leads them to have larger problems like thinking that a firewall protects against phishing. One problem with the helpful graphical interfaces on firewalls is that they abstract away the more core concepts making it challenging for an overly interested user to get a solid understanding of how the technology works.

C. Educational Games

There is growing research in the field of educational security games in an attempt to teach difficult and sometimes hard to grasp topics in an engaging fashion. Such games have the potential to attract a wide audience and can be used to teach children, students, and adults alike [8], [12], [22]. There is some debate about the distinction between educational games and edutainment games, in which edutainment games present the "skill and drill" repetitive learning of a task whereas educational games tend to be played by using strategies, testing hypotheses, and advanced problem solving [11].

D. Computer Security and Firewalls in Games

Several games have attempted to teach computer security terminology and concepts to beginners and experts alike. We discuss some of the most relevant games below, for a more comprehensive review of computer science themed games we refer the reader to a systematic review by Battistella et al. [3].

Computer Security seems like a perfect match for educational games. The topic itself is interesting and easily lends itself to adversarial situations commonly used in game design. Indeed there have been many capture-the-flag style games created from the perspective of a "hacker" or "penetration tester" trying to break into a system. One example is the OWASP Security Shepherd [23] training modules which combine instruction with challenges where the learner can attempt to use their new knowledge to break into the system. Similarly, Carnegie Mellon University's PicoCTF security game allows players to "reverse engineer, break, hack, decrypt, or do whatever it takes to solve the challenge" [28]. One common theme among these types of games is that the player is trying to break into a system by exploiting security failures, rather than trying to configure a secure system. The games are not always beginner friendly, particularly for people who are either less aggressively inclined or who do not already understand a large number of potentially complex computer science topics.

The last few years have seen the development of more beginner-level security tabletop games, partially due to their natural friendliness (no computers at all) and their ability to promote discussion within a group of players, rather than solving problems alone. Recent tabletop games include: [d0x3d!] [15], Control-Alt-Hack [10], Android: Netrunner [1], Protection Poker [29], and Elevation of Privilege [20]. Control-Alt-Hack and [d0x3d!] are specifically intended to be played by members of the general public with the goal of teaching both interest and terminology. We further discuss and evaluate [d0x3d!] in Section III-C.

There have also been several academically created video games, namely: CyberCIEGE [8], SecurityEmpire [22], and Anti-Phishing Phil [25] for the wider computer security context. Some of these have been successful in an educational context, with students, teachers and even parents reporting that they enjoy these games. CyberCIEGE is a single-player video game in which the player can play through a number of different security scenarios and present people with a resulting simulation of that scenario. The game is aimed at security professionals and assumes a good background in computer security. SecurityEmpire is a multi-player competitive game in which students manage a green energy company, which requires using good security practices. The game is aimed at high-school aged players and aims to teach people select fundamental concepts from the area of Information Assurance.

III. Requirements Gathering

To further define the learning goals of our game we first conducted three exploratory expert interviews and a design workshop with three students.

A. Firewall Administrator Expert Interviews

Prior work on the types of conceptual problems experienced by firewall administrators is surprisingly sparse and primarily focused on the resulting errors rather than the misunderstandings that lead to the errors [14]. To better understand firewall management challenges we therefore decided to conduct interviews with two local Firewall administrators who were contacted through personal connections of the authors.

Interviewee A had 20 years of experience in networking and was working on the university networking teaching space as well as a local ISP in his free time. Interviewee B had 3 years of experience and was also working for the same local ISP full-time. Their ISP uses the firewall tool *ipfw* (common Firewall tool for FreeBSD) and the automation engine Ansible.

The interviews were semi-structured as to make sure key questions were asked while also allowing the interviewees to talk as much as possible. Questions included asking about their typical work day, a retrospective question about the last time they had to edit a firewall rule, prior difficult firewall editing experiences, and experiences teaching others about firewalls as part of their job. Some initial game design sketches were also shown to gather feedback. The interviews were audio recorded with interviewee consent.

1) Results: Expert A described his work as including maintenance of a network, extending it, and reacting to problems that might arise in it. He explained that automation for maintenance and extending the network purposes has increased and that he does not "reliably remember syntax" of firewall

rules anymore as tools help him handle syntax. Instead, he stressed the importance of teaching higher-level concepts such as chains, traffic classification, and evaluation of rules rather than particular tools.

Expert B similarly explained that spelling and syntax errors were rarely a problem for him. He regularly uses an automation tool called Ansible which takes care of many of the menial errors. He also felt that firewall education should focus on explaining core concepts, rather than include any other added complexity. Prior to joining the local ISP, Expert B only had basic experience with *iptables* and added that "once you know one tool, it's easy to transition", highlighting the need for a focus on concepts rather than exact syntax. Finally, he recommended that we use simple scenarios such as allowing web traffic and introducing a default policy of *deny* or *drop* to illustrate common rules often found in a firewall.

B. Educational Games Expert Interview

We also consulted an expert with more than 30 years of experience in designing and creating educational technology as well as more than 7 years experience in game-based educational approaches. We explained the educational goals of the game and provided some initial game ideas for feedback. The expert recommended including a character that needs help in achieving an objective rather than simply teaching the player as it would work well with beginners and provide more motivation to approach a variety of problems. She also suggested we employ a scaffolded learning approach by using a number of different levels, as doing so would not only signal a sense of achievement and progress to the player but also offer a good approach to introduce more complex topics later in the game and build upon previous knowledge.

C. Design Session

As a design start-point we also ran an hour long session centered around the board game [d0x3d!] which is aimed at beginners and is designed to teach basic vocabulary in an informal and fun setting. We selected [d0x3d!] because it is a networking game, which is the closest type of game to firewalls that we could find. In it the players work as a team to accomplish the goal which facilitates group communication providing us with information about the game play, learning goals, and how various game mechanics assisted or confused the process of learning concepts.

[d0x3d!]: is an open source computer security board game [15]. Players take the role of white hat hackers that need to navigate through the network to reclaim digital assets such as personal identifiable information and authentication credentials to avoid the data being made public. In order to decrease the competitive notion between players and reduce barrier to entry, it is played in a collaborative fashion, meaning that players either succeed together in reclaiming stolen digital assets or lose together by being detected by the network admin.

The game is intended to be played by people who have little to no experience with computer security. Peterson, one of its authors, regularly uses it in high school outreach to get kids interested in computer security because it is accessible and promotes group discussion.

Participants: We asked a Computer Science PhD student who previously worked as a network administrator professionally (participant A), a Computer Science undergraduate student (participant B), and a Psychology PhD student working in Human Computer Interaction (participant C) to play [d0x3d!] and provide feedback. The lead author also participated in playing the game, to fill out the players and facilitate the discussion.

Protocol: The session was conducted in a lounge-type room with couches and low tables. The room was reserved to ensure minimal interruption. A video camera was used to record the table and the voices of the participants. It was turned on after consent was obtained.

The session started by informing the participants about the goal of the session and providing them with informed consent. They were then given a short pre-test to collect demographics and ask basic questions about prior firewall experience. The researcher then described how to play the game and provided each player with a short sheet describing the key rules for [d0x3d!] for reference. The researcher provided rule clarifications throughout the game play. Because we are interested in Firewalls in particular, we used an initial network setup for [d0x3d!] which placed security nodes in key network positions rather than randomly. Creating custom network node layouts is encouraged in [d0x3d!] as a way to think about how different network layouts might be easier or harder to defeat.

Participants played the game for roughly 30-35 minutes, when the game ended naturally as no further moves could be made. The participants were then asked to fill out a post-game questionnaire which contained the same questions as the pre-game questionnaire, less the demographic questions. We then asked the participants to discuss their experiences as a group.

1) Results: Participants all grasped the gameplay, rules and aims of the game quickly, including participant C who had no experience in networking or security. Despite the new vocabulary that the game introduced such as "compromising", "zero-day exploit", "patch" and "loot", participant C seemed very engaged and asked the other more experienced players for input and collaborated effectively. Participant B was a little more quiet but grasped the rules very well and explained them to the other two players if necessary. Participant A clearly found the layout of the nodes counter-productive from a learning perspective, likely because he was more aware of how actual networks function than the other two players.

During the post discussion everyone agreed that [d0x3d!] had primarily taught them new terminology and little else about networking, which is in line with the findings of [22]. Participant B expressed that you do not need to think about the context at all in order to play it and participant A added that even his 8-year old son could play the game, despite the game being advertised for ages 12 and upwards. None of the participants felt that a completely random setup of tiles in the network would be sensible and all agreed that there should be a certain network that resembles reality. Even though we had arranged the nodes in a more logical format, the structure of [d0x3d!] effectively forces all network components into a grid shape, or "soup" to use participant A's terminology. The result is that players spend large parts of the game "chasing around for a picture" rather than thinking about the network

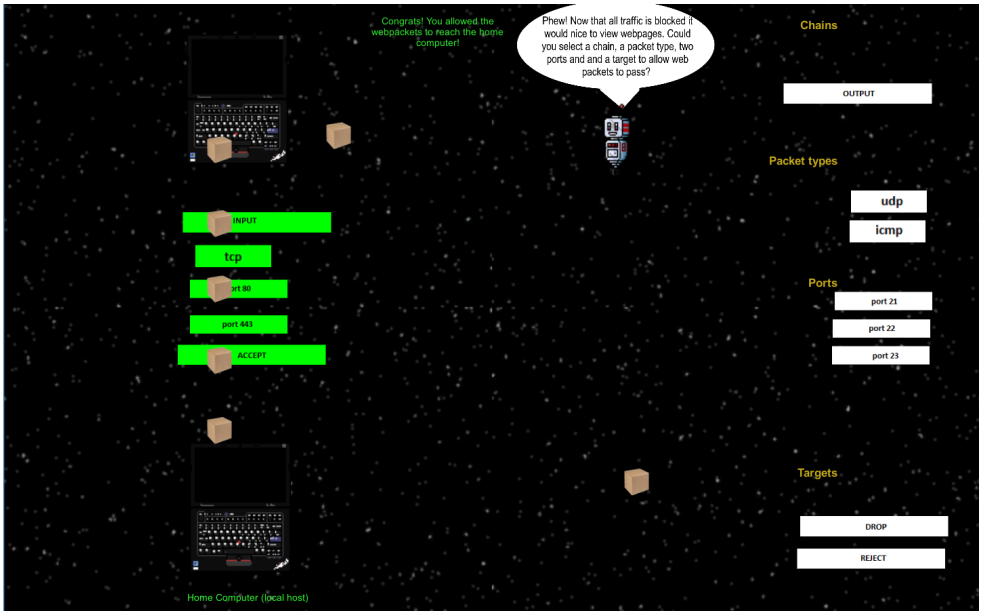

Fig. 1. Game screen for the first design of the game that required players to construct a firewall rule. The player can move elements in drag-and-drop fashion.

layout logically or engaging with the actual functionality of the different components.

These results are somewhat to be expected as the primary goal of [d0x3d!] is to teach terminology and generally get young people interested in security, rather than teach the complexities of actual network administration.

IV. Initial Game Design

A. Design Goals

Based on our readings and requirement gatherings described above, we made several high-level design decisions about the game we wanted to create. First and foremost, we wanted the game to be accessible to a general audience. Someone with no computer science experience should be able to easily play the game and come away with an improved understanding of what a firewall is and how it works.

We also wanted to avoid attack and defense terminology. Many security games are combative in nature either directly or through their use of language. It was important that our game be as open and welcoming as possible, we hoped to achieve this partially by avoiding attack and defense language and instead focusing on the concepts of building and sorting.

Terminology development is clearly important as observed by prior game designers [10], [15]. Our board game players also pointed to the value of both learning the terminology as well as being able to apply it in a meaningful way such that both the term and the meaning were learned.

Firewall concepts such as chains, rule evaluation order, and default policies were clearly called out by our interviewees as important. They were also identified as causes of real-world firewall error by researchers [30], [31]. We therefore decided to focus on these firewall concepts in particular.

Finally, we wanted to facilitate accurate mental model development. One observation we made of [d0x3d!] and Control-Alt-Hack is that they both taught terminology such that a player had difficulty applying the knowledge in other situations. Interviewee B highlighted how important it was for him to be able to re-apply firewall concepts to new languages. One aspirational goal for the game is that a player be able to apply the gained knowledge to read an actual *iptables* rule.

B. Game Mechanic

The first iteration of the game was loosely based on the game "Lemmings" [9], [26] which enjoyed popularity in the early 90's. In the original Lemmings game, players had to quickly direct the progress of a line of characters which descended from the top of the screen in order to help them exit via a safe path. In our game the packets took the place of the Lemmings traveling from one computer to the next and the player's goal was to setup the firewall so only the correct packets went to the destination computer. The initial design focused solely on packets traveling from top to bottom (Figure 1). Players could place building blocks in the path of the packets to alter the path, with blocks turning green and letting packets through and other building blocks that stayed white and blocked them from passing through. Similar to the Lemmings game, packets started falling as soon as the player started the level requiring quick action, somewhat mimicking the interrupt-driven nature of administration work.

The building blocks contained different elements commonly found in firewall rules such as ip addresses, ports, and protocols. The box size was short for ports and associated numbers, whereas commands as ACCEPT and DROP as well as Chains had longer building blocks. We intended for the blocks to be placed in order, with the DROP and ACCEPT block at the end as the default rule.

Four initial levels were created using Unity [27], with each level increasing in difficulty. The game could be controlled using the mouse in a "drag and drop" style allowing players to place the blocks anywhere on the screen. We had

intended block placement to be limited to the path between the computers; however, the initial prototype enabled placement anywhere. Similarly, the prototype did not yet have a point system. Feedback to the user was limited to "You won!" or "You lose!" with a short explanation as to why. No tutorial was given ahead of the game, as it was intended to have the player learn through experimentation with the interface.

1) Small Scale Evaluation: We conducted an initial evaluation of the game prototype with two participants: a complete beginner, inexperienced with computer networking and security (participant A) and a Computer Science Masters student who had taken classes in Networking and Security but still described herself as a novice (participant B). They therefore fit well with the intended target group.

The participants were invited to come play test the game together. Similar to the board game testing, they were provided with a consent form, pre/post-tests and a video recorder was used to capture the content of both screens and the voices of the participants.

We initially assumed that a walk-up-and-use style game was possible and that players would slowly build an understanding of the game mechanics through experimentation. Blocks could be easily dragged and changed color when they interacted with packets. Similarly, packets were set to change direction when encountering certain blocks, providing fast and visual feedback. Our participants, however, struggled with the meaning of different game elements. Participant A had difficulty with terms like "packet" and therefore had trouble visually identifying one. Participant B understood that the packet icon indicated packets of data and that movement showed the transmission of data from one network to another. She was also able to correctly recognize common networking elements like the port numbers but had difficulty with UI elements such as what the rectangles did.

The initial screen setup was also not condusive to instruction reading. The packets started falling as soon as the level loaded which encouraged the players to start dragging boxes around to stop the falling of the packets without really understanding what they were trying to accomplish. As game play progressed to higher levels the behavior of the two participants increasingly differed. Participant B read the instructions and correctly dragged the game elements into the packet path to direct the packets. Participant A persisted in not reading the instructions and dragged many objects around the screen. Participant B explained to participant A that here it helped that "she studied computer science", meaning that she knows what "tcp" and "ports" mean, despite her lack of knowledge when it comes to firewall rules. She was able to put a firewall rule together according to the structure, without fully realizing that she did. Participant A, on the other hand, dragged every object on the screen around randomly.

The game play outcomes indicated several core problems with our approach. First, the game needed more explicit guided learning as opposed to experimentation. The idea of sorting packets was good and made sense, but could not be trivially learned by the players through trial and error. Explicit instruction was needed. Even participant B, who had taken a networking class, had trouble with the concept of rules and default behavior. These concepts are non-obvious and need to be explicitly addressed in the game instructions. They also need to be broken up across multiple levels so that each concept is explained individually. Similarly, terminology was a problem. Participant A finished the game being able to say "tcp" without really understanding what it meant. Though he was curious to find out.

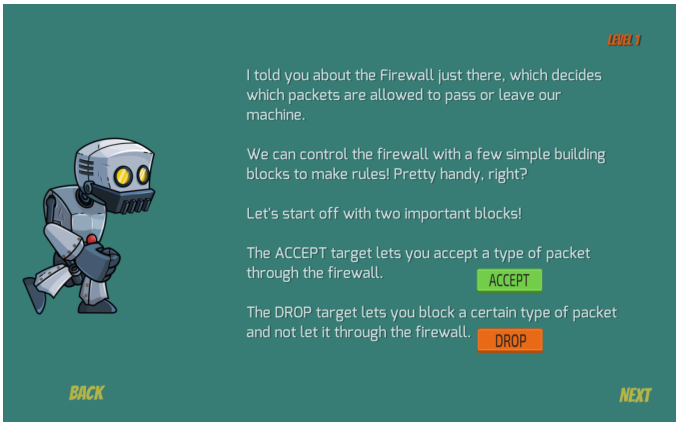

Fig. 2. A tutorial Screen for Level 1 that presents information about ACCEPT and DROP blocks.

Participant B also suggested we add predefined outlines for the rectangles/rule components, which would act as visual cues, making the decision process easier and less intimidating. Participant A verbally agreed that this is a good idea, and participant B also pointed out that this would make the game easier to play on mobile devices.

The packet animations were helpful to the participants in understanding the impact of the rules. But the pressure to do something about the falling packets was too stressful and violated our safe and welcoming goal. The simultaneous building of the firewall rule and packets moving around did not achieve the learning goal and led to more trial and error reactions because of short amount of time available, rather than actually reading and absorbing the instructions and thinking about the next movement.

The robot character was well received by participants who thought it looked good and gave them something to relate to. Its purpose, however, was less clear, particularly to participant A who initially thought it managed the firewall.

V. Final Game Design

The final version of Permission Impossible can be played online at:
https://groups.inf.ed.ac.uk/tulips/projects/1617/PermissionImpossible/

The final game design consists of two types of screens: instructional (Fig 2) and policy building (Fig 3).

A. Instructional Screens

Instruction screens provide the story line of the game as well as context, explanation of the terms, and the details of missions. Fig 2 is an example instructional screen from the initial introduction sequence explaining the idea of accepting and dropping packets.

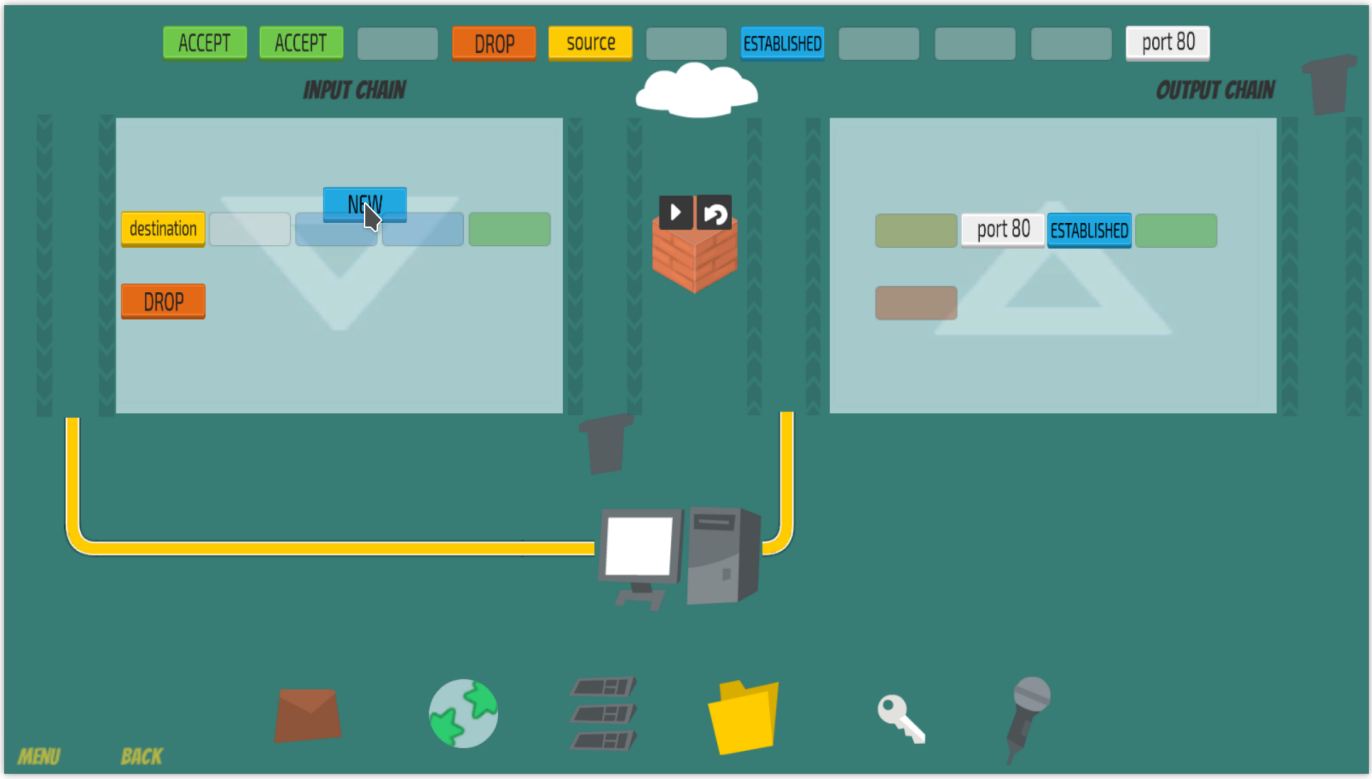

Fig. 3. Firewall policy building screen. Screenshot is of Level 2. Other levels are similar but have different sets of blocks along the top and different empty blocks for the input and output chains. The current mission is to allow web traffic inbound to port 80 and outbound to a port 80 destination, with a default policy of DROP. The player sets the policy by dragging the blocks at the top onto the boxes in the INPUT or OUTPUT chain areas where they snap into place.

The user is initially greeted by "Roboto" who wants help ensuring "that all the valuable data exchanged between different computers is delivered in the correct fashion." Roboto then proceeds to explain how data is transfered using "packets" and that the flow is managed by a firewall. He would like the user's help in setting up the firewall. All instructions and all missions are given by Roboto on instructional screens.

B. Firewall Policy Building Screens

The policy building interface is shown in Fig 3. This is the main screen the user interacts with in order to complete levels. Colored blocks representing rule elements are positioned across the top. To set a policy, the user drags the blocks onto either the *input* chain (left) or the *output* chain (right) where they snap into the provided lightly colored empty boxes. Color is used to help the user understand which blocks go where, but blocks can be dropped onto any available empty box.

Once done, the user clicks the play button in the middle of the screen. If they have the wrong answer, Roboto appears to tell them that it isn't quite right. If the user has the correct solution, an animation is played showing packets traveling across the network. One set of packets descends from a cloud on the top of the screen, progressing either to their corresponding service at the bottom of the screen (ACCEPT), or into the trash bin next to the input chain (DROP). Similarly, a set of packets is generated by the services at the bottom of the screen that then ascend to either the cloud or the trash bin next to the output chain. Each packet in the animation is either a generic packet block or is shown with one of the service icons from the bottom of the screen. For example, HTTP packets have a world icon which is commonly used to symbolize the Internet. After the animation, the user progresses via a "Next" button which appears.

C. Levels

Permission Impossible provides scaffolded learning through increasingly complex levels. Initial levels provide detailed instruction, with later levels progressively providing less details, less intuitive missions, and finally removing the color hints from the interface completely.

Level 1: introduces the policy building interface as well as the basic concepts of having different rules for the packets coming in and going out. The initial mission is designed to be easy to accomplish. The user is asked to drop all incoming packets and allow all outgoing ones. The building interface presents them with only two blocks: ACCEPT and DROP. The chains similarly, have only one empty box each.

Level 2: introduces the idea of having more than one rule in a chain and that rules are executed in order. The user is asked to setup the *input* chain to allow new and established connections with a destination of port 80, and the *output* chain to allow established connections with a source of port 80. Both chains should have a default drop rule. Fig 3 shows a player completing Level 2.

Roboto clearly describes the exact rules he would like enacted, including images of the blocks to be used. Roboto

ID	SUS	Age	Gender	CS knowledge	Security knowledge
P1	90	22	Male	Yes	Yes
P2	87.5	23	Female	No	No
P3	87.5	28	Male	Yes	No
P4	92.5	22	Male	Yes	Yes
P5	95	26	Female	Yes	Yes
P6	82.5	26	Male	No	No
P7	85	24	Male	No	No
P8	90	23	Male	No	No
P9	85	56	Female	No	No
P10	87.5	56	Male	No	No

TABLE I. FIRST FIVE PARTICIPANTS COMPLETED THE FULL LAB STUDY. LAST FIVE PARTICIPANTS PLAYED THE GAME REMOTELY AND ONLY PROVIDED DEMOGRAPHICS AND COMPLETED THE SUS SCALE.

explains that these rules will allow the use of the "web" but does not elaborate about services.

Levels 3-7: each introduce a new protocol with Roboto explaining what the protocol is and how it is used in networks. In order, introduced protocols are: SSH, FTP, Domain Name System (DNS), SMPT, and SIP. In these levels Roboto uses only words to explain what he wants the user to do.

Level 8: combines the knowledge of services and ports learned so far and introduces the concept of multiple complex rules. The user is asked to set rules for both web (80) and SSH (22) traffic as well as setting a default policy.

Level 9: takes a different approach from the previous levels and asks the user to implement a default policy of ACCEPT. The user is also instructed to block a specific IP address that Roboto describes as being malicious in the instructional screens.

Level 10: uses all the knowledge the player has gained. Roboto has an emergency and asks the player to "construct a sensible ruleset" without any detailed instruction. It also removes the color hints and includes a IP address to assess whether the player understands the concept of a default policy.

VI. METHODOLOGY

To evaluate the effectiveness of Permission Impossible we conducted a lab study with five participants and an additional online study with a further five participants. Lab participants completed a pre-test, played the game to completion in front of the researcher, and then completed a post-test to determine if the game had the intended learning outcomes.

A. Participants

Participants were recruited through personal connections of the authors'. They were selected to represent a range of potential users and included people with experience in computer science (CS) and computer security, as well as people who were novices. All of the participants were current students or recent graduates. Table I shows the demographics.

Five of the participants (P1-5) were able to complete a formal evaluation with the game in a lab-type environment including the pre/post questionnaires described below. A further five (P6-10) wanted to help with game evaluation but for various reasons were not able to participate in adequately controlled setting. Instead they played an online version of the game and only filled out the SUS scale.

B. Pre/Post Questionnaire

Demographics: (pre-test only) including participants' age, gender, and academic background.

System Usability Scale (SUS): (post-test only) is a "simple, ten-item scale giving a global view of subjective assessments of usability" [5]. It is a subjective Likert scale and covers areas such as the need for support, training, and complexity of a system. We included it as a commonly known measure of how acceptable a system is to users.

Knowledge of firewalls: Six questions asked about firewall familiarity starting with if they had ever heard the term "firewall" previously and ending by showing the participant a list of technical terms and asking them to mark which ones were associated with a firewall. The purpose of these questions was to gage before and after familiarity with firewall concepts.

Reading an iptables *command:* Participants were presented with the following *iptables* command, and asked what it means:

```
iptables -A INPUT -i eth0 -p tcp port 443 -m state
state NEW,ESTABLISHED -j ACCEPT
```

Since Permission Impossible is loosely based on the *iptables* syntax, we anticipated that some players completing the game would gain the ability read simple *iptables* commands.

Define terms: Participants were asked to provide a free-text description of each of the following concepts: "default policy", "protocol", "chain", and "INPUT vs OUTPUT rule".

Understand the rule building interface: Participants were shown a screenshot of the policy building screen similar to Fig 3. They were asked to indicate what different elements of the screenshot were in order to determine if the interface elements were readily recognizable (pre-test) and if participants had learned to correctly associate concepts after playing the game (post-test).

C. Protocol

Four participants completed the following protocol in-person and one completed it remotely using a combination of email, instant message, and local screen recording software.

Participants were first informed about the content of the study and asked to fill out a consent form followed by the pre-test. In-person participants were provided with paper forms and the remote participant was given a Google Doc version.

They were then provided with the game either preloaded into a web browser (in-person) or as a link (remote). In-person participants were also provided with a large monitor and a computer mouse. Interactions with the game were recorded using the video capturing software ShadowPlay [21] which records the interactions on the screen. The remote participant ran this software on her computer when playing the game and provided the resulting video file. Audio was not captured, but the researcher took detailed notes. All participants were able to play the game to completion and took 27 minutes on average.

After completion, they were provided with the post-test and asked to fill it out. On completion of the post-test they were asked verbally if they had any further questions or comments.

VII. Results

A. Pre-game questionnaire

Prior knowledge of firewalls: All participants indicated that they had heard the term "firewall" before. When asked to describe what a firewall does, they said that it prevents unwanted access (P3) and protect the computer from hackers (P2). Overall, P1, P2, and P3 indicated that they did not know how firewalls operate whereas P4 and P5 did.

Knowledge of firewall terminology: Familiarity with firewall terminology varied. P2 who had no knowledge of CS indicated that she did not know what a packet was. She also indicated limited familiarity with protocols and services and did not know what policy, rules, or protocols were. P1, P4 and P5 all indicated that they had knowledge of computer security but did not successfully describe all the terms related to firewalls prior to playing the game. They did however recognize a higher number of protocol names and services. None of the participants could explain what "chain" meant in terms of firewalls.

Reading an iptables rule: Apart from P4, every participant struggled to read an *iptables* command and could not identify what the words and numbers meant.

Input vs. output rules: It was not expected at this stage that participants would understand the difference between *input* and *output* but we included this question in the pre-test to check whether they would relate the terms to incoming and outgoing traffic. P1 and P4 (both having Computer Security knowledge) correctly identified the terms relating to connections. However, P5, despite having Computer Security knowledge, did not. P3 tried to relate it to something being inputted and outputted from the computer but did not relate the knowledge to firewalls.

Understanding of the rule building interface: A policy building interface screenshot was shown to participants to see if they could understand the interface of the game. We asked them to associate terms with certain game elements on the screen to see whether they could make connections between terms and the design elements. Results from this question varied greatly. Every participant apart from P1 could correctly identify the packets on the game screen. Every participant, apart from P4, only circled the brick block to show the firewall and did not identify the *input* chain and *output* chain as part of the firewall. Three participants could not locate the firewall rule or identify the policy. Three participants could not point out that the six symbols below the computer indicate the different services at this stage. The purpose of the bin was clear to three participants. The identification of the cloud to signify the Internet was not clear to two participants and they related it to data storage instead.

B. Game testing

Every participant carefully read through the instructions for the first level and tried to understand the tasks given to them. P1 and P3 needed to be prompted that the blocks at the top could be dragged around the screen, whereas P2, P4 and P5 seemed more familiar with the drag and drop style of the game. Once, P1 and P3 figured out that these elements could be dragged, none of the participants had visible problems with the first level and all participants successfully advanced to the next level. P1 also clicked on the moving packets during the animation to see whether he could interact with the packets and then realized that the dropped packet went to the bin to be destroyed.

The second level presented an increased challenge to the players because the user was required to construct rules on both the *input* and the *output* as well as set a default policy. The tutorial screens preceding the level gave the user information on chains, rules and how the rules are read. It also gave the user concise instructions on what exactly they had to implement in the game as to not overwhelm the user. P1 and P3 did not initially make the connection between the colors on the blocks and empty boxes. P2, P4, and P5 all matched the colors of the building blocks with the rule which helped them to accomplish the level faster than P1 and P3.

All participants had problems correctly placing the "destination" and "source" blocks into the *input* and *output* chains respectfully. This confusion makes some sense as these blocks are visually similar and hints like the colors are of no help. Also, for several levels, such as level 2, the *input* and *output* chain rules differed only in source versus destination elements. This problem occurred despite the tutorial screen specifically stating "destined for port 80" and "originating for port 80" in the different quests. This problem continued on later levels for all participants, although they recovered quicker from the error. By the time they reached level 6, all participants successfully managed to make the distinction.

Level 9 which presented the players with an ACCEPT default policy and the task of blocking a malicious IP address again highlighted that identifying source and destination presented problems for the players. It appeared that players were used by now to placing destination and source at *input* and *output* chain respectively, whereas Level 9 required them to put them the other way around. Level 9 contained explicit feedback for this error, as we had anticipated the problem, so participants easily recovered. P4 seemed to have thought that he was smarter than the game and had forgotten that the IP address belonged to a malicious sender. P2 was interested in how one can spot a malicious or suspicious IP address and the researcher explained to her that there are directories on the Internet summarizing which IP addresses should warrant special caution. Given that P2 had no previous knowledge, this interest was seen as a positive learning experience that extended beyond the game itself.

P4 also explained during game play that he could just color match and "win the game anyway". This was echoed by P1. Both participants had computer security and computer science knowledge, so it is not surprising that they found the repeating levels teaching about services easier than P2 and P3. However, the last level also presented great difficulty to them with both P1 and P4, verbally regretting, cursing and then laughing that they did not pay enough attention earlier thinking that they had outsmarted the system. P5 was the only player to successfully complete this level without making any mistakes and also completed it the fastest.

The completion time of the game also varied greatly with P1 taking around 23 minutes, P2 taking 33 minutes, P3 taking 38 minutes, P4 taking 26 minutes and P5 only taking 13

minutes. These different completion times correspond roughly with the users' experience levels.

C. Post-game questionnaire

Knowledge of firewalls: Participants that previously indicated that they did not know how a firewall operates now provided an answer. Answers included: "A firewall has rules and checks whether packets match these" by P2 who previously indicated that she did not know anything about Computer Science or Computer Security. P2 also learned about the term "packet" from playing the game and afterwards could accurately describe them.

Knowledge of firewall terminology: All participants were able to answer more terminology questions than in the pre-test, as well as correctly identify that firewalls use rules, policies, ports and read IP addresses. P2 could not explain what a chain is exactly and did not indicate that this word sounded familiar to her. P3, P4 and P5 stated that a chain is a "a set of rules" (P3) that are being made related to incoming and outgoing traffic.

Reading an iptables command: The ability to understand parts of the iptables command increased after playing the game for all of the participants. Even P2 who has no Computer Science or Security background could accurately describe what the command meant.

Input vs. output rules: Participants were able to explain the difference in the post-test. P5, who left the explanation blank in the pre-test, explained that the "Input rule specifies which packets are allowed to enter the system. Output rule specifies what is allowed to leave the system".

Understanding the rule building interface: P1 did still not identify packets correctly, despite his knowledge of computer science and security. P1, P2, P3 and P4 could now identify that the INPUT chain and OUTPUT chain areas were part of the firewall whereas P5 did not. Four participants correctly identified a rule after playing the game, with the exception of P1 who circled "Established". P2, P3 and P4 correctly identified the default policy. The identification of services was mixed with only P2, P3 and P4 correctly identifying them.

SUS outcome: The average score for the SUS for the ten participants was 88.25, which is generally considered a high degree of usability. Non-Informatics participants found the system slightly less usable at 86.25, whereas Informatics participants considered the system usable at 91.25. Moreover, female participants also considered the system more usable at 89.167 whereas male participants gave the system an average score of 87.857. The individual results can be found in Table I.

VIII. Discussion

While the area of security games already contains some good options, we feel that games like Permission Impossible fill an unique and important space. Existing games tend to be either overly simplified or highly competitive in nature, making entry challenging or unappealing for some populations. The issue of appealing to younger generations and more diverse groups is already a large issue being addressed by the National Cyber Security Center's Cyberists marketing push. Notably, their wording attempts to shift the language from attack style language towards a more community-service concept.

> We've chosen the term 'Cyberist' to describe – in a more positive light – the role of someone who works in the cyber security profession. Far from being a shadowy figure, a Cyberist is someone with a dynamic career who plays a vital role in the community and wider society, protecting the information and systems we care about and rely on in our daily lives. [13]

Permission Impossible tries to take a similar language approach. While we didn't ask participants directly about the language framing, our observation was that they responded well to it and the "help Roboto" style story line.

Firewalls were also more interesting to the general public than we considered at the beginning of the project. We initially expected that many people had heard of firewalls previously and associate them with some dull part of computer security. But the term is more recognizable and interesting than we anticipated. We had no issue finding students with no security experience, who despite not being paid, and not being able to participant in the full study, still really wanted to play the game. People are quite interested in this topic and want to know more about computer security concepts that they recognize.

Finally, one thing we realized is that it is easy to consider system administrators as some type of all knowing group when it comes to the systems part of computer security. However, most system administrators have a minimal computer science education and face many of the same conceptual-level learning issues as our audience [19]. Our experts expressed many of these elements such as a strong reliance on their tools to provide feedback and the experience of learning about firewalls over time and moving between different management systems. While our focus was on the younger population, we feel that building a training tool for system administrators focusing on many of the same aspects highlighted in Permission Impossible would be a potentially useful endeavor.

IX. Conclusion

Overall our evaluation found the game to be fun, engaging, and educational for people with and without prior computer science knowledge. Especially for participants without any prior computer security knowledge, the interest in firewalls was piqued and they could accurately explain concepts like an input chain rule or describe the overall purpose of a firewall after playing the game. We believe that this understanding on a conceptual level, especially for beginners, shows early signs of success of the game, that might make it particularly suited to use in secondary schools and therefore serve as another medium to interest girls and boys in computer security. The fact that we received questions beyond the learning objectives covered in the game further underlines that the game is suitable to motivate participants to explore topics like firewalls and security. The game also received a high usability score from participants showing good potential for it being an accessible and friendly game interface.

REFERENCES

[1] Android: Netrunner the card game. Available at https://www.fantasyflightgames.com/en/products/android-netrunner-the-card-game/.

[2] G Andreasson. Iptables tutorial 1.2.2, 2006. Available at https://www.frozentux.net/iptables-tutorial/iptables-tutorial.html (Accessed 05/06/2017).

[3] P Battistella and CG von Wangenheim. Games for teaching computing in higher education–a systematic review. *IEEE Technology and Engineering Education*, 9(1):8–30, 2016.

[4] Jordy Berson. *ZoneAlarm: Creating Usable Security Products for Consumers*, chapter 27, pages 563 – 575. O-Reilly Media, Inc, 2005.

[5] John Brooke et al. SUS-A quick and dirty usability scale. *Usability evaluation in industry*, 189(194):4–7, 1996.

[6] National Cyber Security Center. Cyber schools hubs. Available at https://www.ncsc.gov.uk/information/cyber-schools-hubs.

[7] William R Cheswick, Steven M Bellovin, and Aviel D Rubin. *Firewalls and Internet security: repelling the wily hacker*. Addison-Wesley Longman Publishing Co., Inc., 2003.

[8] Benjamin D Cone, Cynthia E Irvine, Michael F Thompson, and Thuy D Nguyen. A video game for cyber security training and awareness. *computers & security*, 26(1):63–72, 2007.

[9] Graham Cormode. The hardness of the lemmings game, or oh no, more np-completeness proofs. In *Proceedings of Third International Conference on Fun with Algorithms*, pages 65–76, 2004.

[10] Tamara Denning, Adam Lerner, Adam Shostack, and Tadayoshi Kohno. Control-Alt-Hack: the design and evaluation of a card game for computer security awareness and education. In *Proceedings of the 2013 ACM SIGSAC conference on Computer & communications security*, pages 915–928. ACM, 2013.

[11] Mary Jo Dondlinger. Educational video game design: A review of the literature. *Journal of applied educational technology*, 4(1):21–31, 2007.

[12] Allison Druin. The role of children in the design of new technology. *Behaviour and information technology*, 21(1):1–25, 2002.

[13] Chris Ensor. Who are the cyberists. Available at https://www.ncsc.gov.uk/blog-post/who-are-cyberists.

[14] Leonardo H. Iwaya; Artem Voronkov; Leonardo A. Martucci; Stefan Lindskog; Simone Fischer-Hbner. Firewall usability and visualization: A systematic literature review. Technical report, Karlstad University Studies, 2016.

[15] Mark Gondree and Zachary NJ Peterson. Valuing security by getting [d0x3d!]. In *Workshop on Cyber Security Experimentation and Test, Washington, DC*, 2013.

[16] Mark Gondree, Zachary NJ Peterson, and Tamara Denning. Security through play. *IEEE Security & Privacy*, 11(3):64–67, 2013.

[17] Mohamed G Gouda and Alex X Liu. Structured firewall design. *Computer networks*, 51(4):1106–1120, 2007.

[18] Iulia Ion, Rob Reeder, and Sunny Consolvo. "no one can hack my mind": Comparing expert and non-expert security practices. In *SOUPS*, volume 15, pages 1–20, 2015.

[19] Eser Kandogan, Eben M Maglio, Paul P amd Haber, and John Bailey. *Taming information technology: Lessons from studies of system administrators*. Oxford University Press, 2012.

[20] Microsoft. Elevation of Privilege card game, 2013. Available at https://www.microsoft.com/en-us/SDL/adopt/eop.aspx.

[21] Nvidia. Shadowplay. Available at https://www.nvidia.com/en-us/geforce/geforce-experience/shadowplay/.

[22] Marc Olano, Alan T Sherman, Linda Oliva, Ryan Cox, Deborah Firestone, Oliver Kubik, Milind Patil, John Seymour, Isaac S Kohane, and Donna Thomas. SecurityEmpire: Development and evaluation of a digital game to promote cybersecurity education. In *3GSE*, 2014.

[23] OWASP. Owasp security shepherd, 2018. Available at https://www.owasp.org/index.php/OWASP_Security_Shepherd.

[24] Fahimeh Raja, Kirstie Hawkey, Pooya Jaferian, Konstantin Beznosov, and Kellogg S Booth. It's too complicated, so i turned it off!: expectations, perceptions, and misconceptions of personal firewalls. In *Proceedings of the 3rd ACM workshop on Assurable and usable security configuration*, pages 53–62. ACM, 2010.

[25] Steve Sheng, Bryant Magnien, Ponnurangam Kumaraguru, Alessandro Acquisti, Lorrie Faith Cranor, Jason Hong, and Elizabeth Nunge. Anti-Phishing Phil: the design and evaluation of a game that teaches people not to fall for phish. In *Proceedings of the 3rd symposium on Usable privacy and security*, pages 88–99. ACM, 2007.

[26] Kristian Spoerer. *The Lemmings puzzle: computational complexity of an approach and identification of difficult instances*. PhD thesis, University of Nottingham, 2007.

[27] Unity Technologies. Unity game engine, 2017. Available at https://unity3d.com/.

[28] Carnegie Mellon University. Pico ctf. Available at https://picoctf.com/.

[29] Laurie Williams, Andrew Meneely, and Grant Shipley. Protection Poker: The new software security" game". *IEEE Security & Privacy*, 8(3):14–20, 2010.

[30] Avishai Wool. A quantitative study of firewall configuration errors. *Computer*, 37(6):62–67, 2004.

[31] Avishai Wool. Trends in firewall configuration errors: Measuring the holes in swiss cheese. *IEEE Internet Computing*, 14(4):58–65, 2010.