A Comparative Analysis of Hands-on Firewall Configuration Exercises for the Undergraduate Classroom

A. Evan Damon¹, B. Jens Mache¹, C. Miles Crabill¹, D. Kaleb Ganz¹, E. Claire Humbeutel¹
¹Department of Mathematical Sciences, Lewis & Clark College, Portland, OR, USA

Abstract - Teaching cybersecurity through hands-on, interactive exercises is a good way to engage students, especially undergraduates. In this paper, we compare three firewall configuration exercises: FireSim, DETERLab, and RAVE. We found that they are usable but could be improved upon. They each have strengths and weaknesses, and an exercise combining their strengths would be ideal. There were three main strengths: simplicity, extensibility, and competition. Each of these elements serve to make the exercises engaging and educational.

Keywords: security, firewalls, education, exercises, networks

1 Introduction

As computer security continues to develop as a field, one thing remains an important element in keeping networks and computers secure: firewalls. Firewalls are often one of the main lines of defense in a system. Despite this, most firewall education is done by employers, as different employers use different firewalls and/or syntax. We believe that teaching a conceptual understanding of firewalls to undergraduate students will help them to gain a more complete knowledge of computer security. While detailed knowledge of every element of security would be ideal, it is less than feasible. However, conceptual knowledge is easily achievable through interactive exercises.

Multiple interactive firewall exercises already exist. In an effort to find one suitable for teaching conceptual knowledge to undergraduates, we used and evaluated each of three exercises. The three exercises were: FireSim,¹ a Java applet-based simulation of a firewall environment that pits players against one another; DETERLab,² a remote environment, which involves the creation of firewall rules on real, working machines; and RAVE,³ a virtual environment complete with lab exercises, including the configuration of a firewall. We chose these exercises because they were readily available, and, as firewall education tools, fit the purpose of our evaluation well.

2 FireSim

One teaching exercise we examined was created by Professor Ken Williams, of North Carolina A&T State University. The firewall simulator (or FireSim for short) was created using Java applets supplemented with XML files. The setup involves downloading a group of necessary files to a networked machine and running the Java .jar file. This machine serves as the host machine. Players then connect to the host machine via web browser and choose a username. Once the administrator/instructor (on a separate computer) uses the GUI to start the game, players are able to conduct attacks against one another in a competitive setting. Through a GUI, players select an attack and direct it at a particular opponent. If the attack is successful, the attacker receives a point while the defender loses a point. After 60 seconds, the same attack can be launched against the same opponent.

This is where the firewall education takes place; in order to prevent successful attacks from other players, you must write firewall rules. These rules use the proprietary Cisco firewall syntax and are simply typed into a special applet window. The goal is to create a list of firewall rules that prevent other players from successfully attacking you. This can take many forms, but most rules block access to certain ports, depending on IP addresses.

To introduce complexity, FireSim allows the administrator/instructor to give players new “tasks”. Each task may require players to create additional firewall rules that mitigate the new situation or vulnerability. The task is announced to the players, and after 60 seconds, players may begin to attack each other using it.

FireSim does many of these things well. The firewall rules interpret correctly and are as intuitive as the syntax allows. FireSim’s network map includes many servers at specific IP addresses and does a good job of simulating a real network. This is one of FireSim’s greatest strengths. The underlying network simulation gives the exercise a great deal of potential for further development.
FireSim has weaknesses as well. It is, after all, a work in progress. The tasks are less clear than they could be. Some are “trick” tasks that do not require additional firewall rules. These are intended to show the user that, when creating a firewall, extending your whitelist too far is dangerous. However, this is not obvious to the user. If a player does nothing and is impervious to attack, they may not learn this lesson. On the other hand, players who incorrectly extend the whitelist may have difficulty understanding why they lose points. Essentially, the level of feedback provided by FireSim is too little for some of its exercises to be successful.

This is the only real shortcoming of FireSim: the scenarios get users to write firewall rules in a competitive setting, but they could explain more to promote a real understanding of what is going on under the hood. The scenarios are often about blocking access to a particular service, which is easy enough. However, the zero-sum nature of the point system may promote an offensive strategy, where players spend time to attack other players as much and as soon as possible. For some players, this may take emphasis away from the firewall configuration and places much of it on attacking.

Additionally, the setup was not entirely stable. Sometimes users could not connect to the game, or when connected, the game did not function. The administrator password is also non-functional, allowing players access to administrator privileges which usually breaks the game.

Overall, FireSim does many things well, but some things need improvement. The exercise is very interactive but does not provide much feedback. FireSim is also competitive, and the structure of the competition does well to engage students.

3 DETERlab

DETERlab is the second exercise we examined, and the first to use a remote-access setup. The DETERlab exercise that covers firewalls also covers Posix file permissions. The tasks for the two are independent of each other, but the reading and information for the two are intermixed. This can make it quite challenging to find relevant information for each of the tasks on the walkthrough webpage. This section of the paper will focus exclusively on the firewall aspects of the exercise.

The firewall portion of this exercise starts out with a description of stateless and stateful firewalls, what the difference between them is, and a little bit of history behind the development of the two. Next, there is a description of how a firewall policy should be designed and different ways to view the problem of designing a firewall. After that, the lab gives the user a crash course on iptables. In it, they tell the user what iptables is, briefly describe the syntax and then give a few example rules using the syntax. The lab then goes on to describe four different network tools. These are nmap, ifconfig, telnet and netcat. Each of these has about a sentence to a paragraph long description and an example of what it returns in the terminal when used.

The interactive portion of the lab is done on two remotely accessed nodes. Accessing these nodes requires the user to connect to their DETERlab account through SSH and then from there SSH into the two control nodes. Each of these two nodes serves a different function. One node is a server and is where the user implements their firewall rules. The other is the system on which the user tests the firewall rules they have implemented on their server node.

The firewall rules are easily established since the server node has a firewall script provided, in which the user only needs to write the rules and then run the script to get a firewall up and running. It is also easy for the user to turn off all rules and remove them by running another script that is provided on the server node. The tasks that the lab asks you to do are: to write a rule that prevents spoofing; allow access to OpenSSH, Apache and MySQL on their standard ports; allow UDP access to specified ports; allow ICMP ping requests; allow all established and related traffic; and lastly to drop all other traffic to any unspecified port. Since the user is remotely connected to the the server, it is possible for them to write firewall rules that lock them out of the server node, requiring the user to reboot the instance through the DETERlab website.

At any time while working on the firewall rules, the user can begin to follow the tests that are provided for each of the assigned tasks. These tests mostly involve using telnet to see if the ports are open and responding as they are expected to. However, there is no test provided that allows the user to determine if their anti-spoofing rules are functional. Also, the recommended test for seeing if all the expected ports are open (and all others are closed) is to write a script that tests every port. This could be difficult for some undergraduate students.

The best feature of using this DETERlab exercise is that it mirrors a real life firewall admin situation since it uses a commonly used firewall syntax, the student must remotely connect to the environment to make changes, and they have a broad spectrum of network testing tools at their command. It is also possible for the instructor to create new scenarios that require the students to modify their firewall to fit this new situation. Many of the advantages of using this exercise for learning/ teaching firewall rules have the potential to also be disadvantages, depending on the proficiency of the students and instructor. Iptables has a complicated syntax that can be hard to learn without a certain amount of experience. Also, the possibility that a student may lock themselves out of the environment is both an advantage and a disadvantage in that it teaches the student about the possibility of doing so in a real IT position, but it takes time to reboot the system and can be frustrating for the student. Another disadvantage is that some overhead is required in setting up the experiment and waiting for the systems to become available for the user to connect to them.
4 RAVE

The third exercise, the Rave lab, is a series of instructions and questions in the book *Principles of Computer Security* that (alongside an installed VMware vSphere client) teach students how to configure a firewall in Linux. The client allows the student to have access to different virtual machines that they can configure. Students follow the steps of the lab, enter commands and rules, and gauge how the different firewall configurations affect the system. The goal of this lab is to teach students how to use iptables (using the UFW syntax) and the effects of different rules through hands on work.

The advantages of this approach are mostly found in its simplicity, resistance to mistakes, and opportunity to practice what you are learning about in the related book. The instructions are easy to understand and follow, and progress in a logical manner. The pictures that are provided make it easy to check how your lab is progressing, and also to find any part of the GUI that the lab is requiring you to use.

It is resistant to mistakes because you are using virtual desktops that can be restored from snapshots. That way, if a student makes a relatively damaging mistake and changes any program for the worse, the problem can be corrected relatively simply, even if the whole desktop needs to be restored.

Finally, students can learn through trying an action, not just reading about it. By doing this, they gain a greater understanding of the subject they are currently working on.

The major disadvantage is the necessity of having to have a connection to the web in order to work. This can cause connection problems if a number of people are using the same account at the same time or the servers are under load. Also, if the web connection is not working, it is impossible to get to the virtual desktop, and thus impossible to do the lab exercise. Finally, if updates or any other changes are applied, the usernames and passwords will change as well, effectively making it impossible to do any work until you have gotten new ones.

5 Comparison

The conclusions made in the table below were informed by our experience with each exercise in an undergraduate classroom setting, and the subsequent application of the skills learned when the students participated in the Collegiate Cyber Defense Competition (CCDC). The comparison is based on each tool’s usefulness in learning network administration and security.

<table>
<thead>
<tr>
<th>Comparison of Firewall Exercises</th>
<th>FireSim</th>
<th>DETERlab 7</th>
<th>Rave/Nestler 7.3</th>
</tr>
</thead>
<tbody>
<tr>
<td>Syntax</td>
<td>Cisco</td>
<td>iptables</td>
<td>UFW</td>
</tr>
<tr>
<td>Scenarios</td>
<td>ftp, dns, http, snmp, ntp, instant messaging, netbios</td>
<td>ssh, http, sql, mail, ping, udp</td>
<td>ssh, http, ftp</td>
</tr>
<tr>
<td>Setup</td>
<td>LAN (need at least n+2 computers for n players)</td>
<td>Cloud (access through ssh)</td>
<td>Cloud (access through vSphere)</td>
</tr>
<tr>
<td>Complexity for students</td>
<td>Simple</td>
<td>Complex</td>
<td>Simple (step-by-step instructions)</td>
</tr>
<tr>
<td>Documentation</td>
<td>Present but not extensive</td>
<td>A long webpage (interleaved with Unix permissions exercise)</td>
<td>Lab book³</td>
</tr>
<tr>
<td>Extensibility</td>
<td>Possible (XML file supplements)</td>
<td>.ns files to build environments</td>
<td>Maybe possible (virtual machine images)</td>
</tr>
<tr>
<td>In-class vs. Homework</td>
<td>LAN setup (most likely requires classroom usage)</td>
<td>Possible as homework, but might require help from instructor</td>
<td>Easy steps make for a good homework assignment</td>
</tr>
<tr>
<td>Best feature</td>
<td>Competitive</td>
<td>Realism</td>
<td>Step-by-step</td>
</tr>
<tr>
<td>Disadvantage</td>
<td>Lack of feedback, bugs</td>
<td>Realism (places student in large, breakable environment)</td>
<td>Slow (needs bandwidth for remote desktop), limited support for Mac OS X</td>
</tr>
<tr>
<td>Syntax</td>
<td>Cisco</td>
<td>iptables</td>
<td>UFW</td>
</tr>
</tbody>
</table>
The EDURange project was inspired by a pressing need for hands-on security education in the undergraduate classroom. EDURange seeks to implement the core ideas of the “hacker curriculum” in conjunction with interactive and competitive exercises that challenge undergraduates to both compromise and secure systems. Designed with a focus on active learning and inquiry, EDURange aims to improve on the scenarios and exercises described above by providing an extensible framework for the development of interactive and competitive scenarios. EDURange is currently implemented in Amazon Web Services’ Elastic Compute Cloud, and is configured to quickly and painlessly create virtual machine instances that participants can connect to via SSH. EDURange players will be in direct control of a virtual instance, an approach that minimizes abstraction between participant and scenario. EDURange will implement systems that enable the simple configuration and design of scenarios, allowing for customized scenarios. EDURange looks to combine the best of each of the reviewed exercises, allowing for a competitive experience that accurately represents the responsibilities of a network administrator tasked with the configuration of firewalls.

7 Conclusion
Based on undergraduate students trying three different firewall configuration exercises, we are concluding that all three are workable firewall education tools, but they can all be improved upon. FireSim is competitive, interactive, and entertaining, but the educational aspects are dampened by the lack of feedback and guidance. A similarly engaging tool with additional guidance would make an excellent exercise for undergraduate security students. DETERlab was the most realistic of the exercises. The setup was also well thought-out. However, accessing and following the exercise was rather involved, and could be daunting for introductory-level students. Finally, the RAVE labs are efficient and accessible. The exercises are well-written, easy to follow, and educational. Running on virtual machines, students can make any number of catastrophic mistakes without preventing further use, since the system can be restored via snapshots. However, the virtual structure can cause service-related issues, and accessing user credentials is sometimes a problem. A web connection is also required, as well as a significant amount of setup to begin using RAVE.

An ideal exercise would combine the strengths of these three tools. FireSim’s competitive aspect engages students on a level that the other two cannot. DETERlab’s realism and breadth is also excellent, contributing to its extensibility. RAVE’s simple approach and resistance to mistakes make it a good platform for teaching introductory-level students. In constructing a new firewall education tool, educators should seek to include these elements. A competitive, engaging tool with an extensible structure will be an excellent tool for teaching undergraduates about firewalls. If it can be as simple and approachable as the RAVE exercise, as well, the exercise will be the most useful tool of those currently available.

8 Acknowledgements
Partial support for this work was provided by the National Science Foundation’s “Transforming Undergraduate Education in Science, Technology, Engineering and Mathematics (TUES)” program under Award No. 114314, by the John S. Rogers Science Research Program of Lewis & Clark College and by the James F. and Marion L. Miller Foundation. We would also like to thank Richard Weiss and Stefan Boesen.