EDURange: Meeting the Pedagogical Challenges of Student Participation in Cybertraining Environments

Stefan Boesen *Richard Weiss* *James Sullivan* *Michael E. Locasto* *Jens Mache* *Erik Nilsen*

Summary

This paper reflects on the challenges that arose and the lessons learned when we used hands-on cyberoperations exercises in our courses. After exploring a range of exercises and platforms (and having discovered their limitations), we designed and built an environment for hosting such exercises called EDURange.

These limitations fall into two categories: technical and pedagogical. One of the main pedagogical issues was that most existing exercises were not aimed at teaching analysis skills, (i.e. a set of practices that support the ability to achieve understanding of complex systems). On the other hand, one of the main practical issues with existing cyber-training environments involves scalability limitations imposed by the inherent resource constraints of existing testbeds. A third techno-pedagogical issue was that scenarios were not dynamic. An exercise that is always the same has limited utility in that there is little incentive for students to repeat it, and with time, the solutions can be found on the Internet. EDURange allows instructors to configure aspects of the scenarios to repeatedly create new variations of the exercises. EDURange is designed especially for the needs of teaching faculty. The scenarios we have implemented each are designed specifically to nurture the development of analysis skills in students as a complement to both theoretical security concepts and specific software tools.

1 Introduction

According to published reports by the SANS Institute and other groups [2], The US faces a major shortage of cybersecurity workers to defend our information infrastructure from attack. In recognition of this need, security has been included as a core topic in the new ACM/IEEE Computer Science 2013 Curricula. Cybersecurity is also mentioned in more than half of the other knowledge areas in this report. At educational conferences such as SIGCSE and regional CCSC conferences, we are also seeing a growing interest in cybersecurity among faculty who do not have expertise in this area. Given the tight constraints of the Computer Science curriculum, most schools do not have the luxury of offering a separate class in cybersecurity. Thus, the first step is to integrate it into other classes both at the upper and lower division levels.

One of the major obstacles to integrating cybersecurity into the curriculum is the amount of work required to create and set up new hands-on exercises that can be easily adapted to any specific course. We found as a practical matter that most deployed exercises and hosted environments have several shortcomings that make them difficult to leverage in our classrooms. Few two-and four-year colleges have the facilities to set up their own hardware cluster dedicated to a security lab. In addition, we wanted hands-on exercises that teach analysis skills. For us, there was a gap between what we wanted and what we could access, so we decided to build our own tool. The criteria we used were:

1. *Flexibility* to specify exercises at a high level and create variations. DETER [3], The RAVE, and SEED [1] provide sets of exercises. Many of them were good but were not easily modified, and they have significant limitations with respect to elasticity and scalability, particularly during busy times of the semester. A challenge to these approaches to creating *long-term teaching tools* is that exercises become stale and answers become easy to Google.
2. *Ease-of-use for faculty*, which includes providing easy access to exercises, making them easy to create (not requiring configuration of VMs manually), and parameterizing exercises so that faculty can se-

lect the level of complexity that matches the level of their class.

3. *Educational goals*: we wanted to implement scenarios that would teach analysis skills, the *security mindset,* and address the CS2013 guidelines. The security mindset is the ability to think about how systems can fail, and be made to fail in different ways. This also extends to questioning assumptions and think analytically about their implications. This could be something simple such as, "What can I assume about this web page that is asking for my password?" to "What is the set of invalid strings that an attacker could send to this application that would make it execute arbitrary computation?" The analysis of the implications often leads to an exploration of the subtleties of failure cases and requires detailed knowledge of the underlying concept. Another analytical skill which is often neglected is the ability to *understand complex, opaque datasets.*

We designed EDURange as a flexible complement to existing facilities. EDURange provides a framework to support exercises in an elastic cloud environment. With EDURange, it is easy to modify an existing exercise so that students can repeat it multiple times, and the instructor does not need to worry about solutions being posted. It is easy to vary the difficulty of an existing exercise and update software versions by making small changes, and variations can be created for different courses.

1.1 Analysis Skills

One of our primary motivations is to create exercises that would nurture analysis skills. When speaking of analysis skills, we mean the ability to reason about large, complex, and opaque data and systems. Strong analytical skills enable people to impose structure and meaning on such artifacts, reason about these relationships, and draw meaningful conclusions or inferences. These are precisely the kinds of skills that we believe are useful in many cybersecurity scenarios from security policy design to reverse engineering to vulnerability analysis. In designing EDURange exercises, we focus on the following list of analysis skills.

- **Verify assumptions** by checking network messages, protocols, file formats and other input data constraints to see if layers of abstraction are coherent and correct. Enumerating and checking if failure modes, exceptions, errors are controlled, caught or anticipated.
- **Gaining understanding** of program, network, or system behavior and semantics, network topology or organization, or a defense posture.
- **Extracting Information** from large data collections, such as analyzing a raw dump of network traffic, intrusion alerts, or firewall logs. Observing and enumerating how software components or network elements are actually composed.
- **Creating Emergent Resilience** Understanding a system well enough to design and propose enhancements to reliability, fault tolerance, or availability.
- **Create Deception** Applying probability and randomization to selectively increase complexity for the adversary to exploit failure modes.

1.2 Discussion, Lessons Learned, and Future Work

We learned that it is hard to build good exercises, especially when the goal is teaching analysis skills. We also learned that the EDURange exercises are an excellent basis for classroom discussion where students' joint experience of undertaking the exercise becomes the focus of discussion. EDURange provides starting points for discussing important topics. For example, Recon 1 can be an opportunity to discuss the OSI model, subnet masking, broadcast addresses, using the command line, and even observing how network scanning tools actually implement certain types of scans.

We learned that when faculty are exposed to EDURange, they want to participate and contribute. Over the past year and a half, more than 130 students and 30 faculty have participated in an EDURange scenario.

References

[1] Wenliang Du and Ronghua Wang. Seed: A suite of instructional laboratories for computer security education. *J. Educ. Resour. Comput.*, 8:3:1–3:24, March 2008.

[2] Z. FRYER-BIGGS. Dod faces cyber expert talent shortage. *Computer*, 33(12):52–59, 2000.

[3] J. Mirkovic, T.V. Benzel, T. Faber, R. Braden, J.T. Wroclawski, and S. Schwab. The deter project: Advancing the science of cyber security experimentation and test. In *Technologies for Homeland Security (HST), 2010 IEEE International Conference on*, pages 1–7, 2010.