AUTOMATED REMOTE SECURITY SCORING ENGINE (ARSSE): GAMIFICATION OF CYBER SECURITY EDUCATION

by

Arsh Chauhan

A Project Submitted in Partial Fulfillment of the Requirements

for the Degree of

Master of Science

in

Computer Science

University of Alaska Fairbanks

December 2020

APPROVED:

Dr. Orion S Lawlor, Committee Chair

Dr. Christopher M Hartman Committee Member

Dr. Jonathan B Metzgar, Committee Member

Dr. Christopher M Hartman, Department Chair
Department of Computer Science

Abstract

The goal of this project is to create an easy to use, extensible, and engaging method to compute scores interactively during a practical cyber security education. Gamification has been shown to be an effective teaching tool and has been used in the offensive cybersecurity education space (via Capture The Flag competitions and challenges such as hackthebox.eu) but there has not been an open-source effort to bring this idea to the defensive side (blue team) aspect of cybersecurity. The Automated Remote Security Scoring Engine (ARSSE, pronounced "Arsh") uses a combination of well maintained open-source tools and custom connectors to facilitate an easy to use, scalable, and secure system to check the state of a computer system against a desired state and award points based on passed checks. ARSSE has been released to the public with the hope that it will fill a gap in training the next generation of information security professionals.

Acknowledgments

This project was inspired by Walker Wheeler and Dayne Brodersons' independent suggestions to look into using Chef InSpec. I do not know where this project would have gone had it not been for that life-changing suggestion.

I would like to thank my advisor, Dr. Lawlor, for all his support and patience. This project would not be possible without his support and the support of the Computer Science Department.

Special shoutout to William Showalter and Brahm Lower for getting me interested in Computer Security by letting me get heavily involved in the UAF Cyber Security Club.

Table of Contents

Introduction

The number of devices connected to the Internet keeps increasing every day, Gartner estimates 5.8 billion devices by the end of 2020 which will be a 21% increase from 2019 (gartner.com). These devices range from smart speakers to crucial infrastructure such as smart meters and building automation (security devices, heat, lighting, etc). Gartner estimates that 1.3 billion of these devices are in the utilities section and estimates 38% of these will be smart electricity meters which shows that the Internet of Things (IoT) market is expanding into critical infrastructure. The increase in connected devices has outpaced the ability of the education system to fill roles that are crucial in supporting the security and management of these devices, with reports estimating about 3.5 million open cybersecurity jobs in 2020 (Herjavec Group) and filling this gap between supply and demand shows a need for more efficient and engaging education in the cybersecurity field.

Computer security education involves hours and hours of self-paced work to learn what is normal behavior and configuration. In addition, the various different possible configurations of a system make it hard to design one training solution that fits all situations. Many cybersecurity concepts are tedious and mentally menial tasks like user auditing or just writing secure configuration files, all of this leads to a curriculum that is not engaging. Gamification has been shown to improve engagement by increasing the feedback rate and identifying pain points early (Huang and Soman). With this in mind, gamification seems like a good idea to try and tackle this crucial shortage of trained cybersecurity workforce as the world moves to put more personal data and critical national infrastructure online.

Background

A prevalent method of gamifying cybersecurity education is via live competitions. The classic live competition that most people are familiar with is the Capture the Flag (CTF) style in which individuals or teams compete against each other to find the highest number of flags. Flags are usually long alphanumeric strings often starting with a special denotation (ex *flag_*) to signify the flag has been found.

Another form of live competition is simulation exercises such as The National Collegiate Cyber Defense Competition (NCCDC) organized by the University Of San Antonio, Texas (UTSA). These exercises simulate a scenario that generally has competitors playing the role of cybersecurity professionals in a company (blue team) while they are being attacked by a group of penetration testers

(red team). Other competitions like Panoply (cyberpanoply.com) combine the CTF aspect with the scenario competition to create king-of-the-hill style competitions.

While vastly different approaches live competitions, CTFs and the king of the hill style competitions engage participants by providing quick feedback and timely rewards for success and this is key in a gamification strategy. This ability to be able to provide quick feedback is key to a successful gamification strategy. Creating a good game is a balancing act between various factors that affect engageability and ease of deployment,

Prior Work

Live competitions are a useful teaching tool for cybersecurity due to their complex multidisciplinary nature. This significance has been stated in many studies, the paper "*Conceptual Analysis of Cyber Security Education based on Live Competitions*" (Katsantonis et al.) studied already existing literature on this topic to derive aspects of live competitions such as concepts, characteristics, problems, and challenges. Since a simple search for keywords returned a lot of literature, the research team used predefined criteria to find valid papers to include in their study. This filtering based on four criteria produced a final list of 34 papers. The paper produces a concept map (invented by Novak in the 1970s) of all characteristics of live competitions, these were then merged into groups such as contest form (Attack, Defend, Jeopardy, etc). Each group was crosslinked with other related groups. This concept map was used to construct a comparative analysis scheme that could be analyzed using three approaches for determining the educational impact of live competitions.

Cybersecurity training for computer users has been ongoing for a while now. "*Exploring Game Design for Cybersecurity Training*" (Nagarajan et al.) describes the traditional techniques of cybersecurity education and their shortcomings. The paper proposes using "resource management simulation games" and "first-person interaction games" to solve the shortcoming of traditional cybersecurity training methods such as Web-based sessions, computer-based sessions (labs or CD-ROMs), IT Security Days, etc. The paper talks about already existing games with a heavy focus on a simulation game called CyberNEXS. The paper also discusses how computer game design concepts can be used for enhancing cybersecurity training and again uses CyberNEXS as an example when applicable.

A major issue in cybersecurity education is keeping the trainees' attention. It has been shown that using games to support various business sectors has increased the level of interest and activity (Prensky). *"A video game for cybersecurity training and awareness"* (Cone et al.) describes a highly customizable

cybersecurity training videogame called CyberCiege developed by the Naval Postgraduate School in 2005. This paper created multiple custom CyberCIEGE scenarios for the U.S. Navy Individual Augmentee (IA) program: one for IT staff and a few scenarios for other users which emphasized various risks such as the distribution of worms and viruses.

The GenCyber Capture The Flag (CTF) was created with funding from the NSA's GenCyber program and is primarily focused on middle and high school students. GenCyber is designed for students who have no previous security knowledge and is structured like a tutorial with hints and all resources needed to solve the challenges packaged with the CTF itself (McDaniels et al.). A major challenge for gamification of education is set-up and repetition (Katsantonis et al.), GenCyber solved this issue by hosting the entire environment consisting of 24 servers on the Remotely Accessible Virtualization Environment (RAVE) Lab (McDaniels et al.). During summer 2015, this CTF was run in 12 different camps involving about 400 total participants with some camps using teachers from across the K-12 range as participants.

Another challenge with education is figuring out what to teach. Established subjects such as physics have identified their "core concepts" that students need to understand before they can be taught more advanced topics. Cybersecurity education is a new field with no established core concepts. Core concepts are timeless i.e. they do not change due to changes in technology, and also need to be hard topics that may prove to be the hardest barrier to mastery (Parekh et al.).

Camps and peer learning could be a useful tool for cybersecurity education. To test this, an observational study was designed in the form of a cybersecurity camp. There was no direct help provided by the researchers apart from three booklets containing headings and knowledge points associated with the topic for the day. Camp sponsors wanted to have general learning objectives and decided on secure systems administration, network security, and cryptography to provide for a large variation in participant background (Pittman and Pike).

Students at the US Air Force Academy (USAFA) would compete in CTF's and other cybersecurity competitions (e.g. NCCDC) for fun. Student motivation was significantly greater while learning for these competitions than for traditional class assignments (Carlisle et al.). Due to this observation, USAFA took their "cyber-related" curriculum and converted it into a CTF framework. USAFA offers a "Cyber Training elective" that is taken by about 17% of their sophomores, this class exposes the students to cyber topics through a CTF and combines it with peer learning by having juniors and seniors who have taken the course mentor the current students.

While most research discussed above has focused on using gamification to teach the technical skills required for cybersecurity, the University of Texas at San Antonio (UTSA) developed an active

learning curriculum to teach the management aspect of cybersecurity as part of its graduate-level program. This curriculum focused on managerial aspects such as security principles, incident response, digital forensics, and security assessments (Conklin). This focus on business led to the creation of a competition (NCCDC) that not only focused on the technical aspects of defending a network but also completing managerial tasks known as business injects. Started as an internal exercise, this competition has now grown to over 200 competing teams from across the US and Puerto Rico (Communications, Raytheon Corporate).

Findings from Prior Work

Katsantonis et al. found some inherent problems with live competitions as an educational tool. The paper found that setting up a competition as an educational tool has to balance various aspects encompassing various aspects such as the competition organizer's bias towards certain topics all the way to logistical challenges like choosing a time frame that simulates nuances in the real world to technical challenges (e.g. limited resources). Katsantonis et al. built a concept map of all the different aspect a live competition for cybersecurity education has to take into account

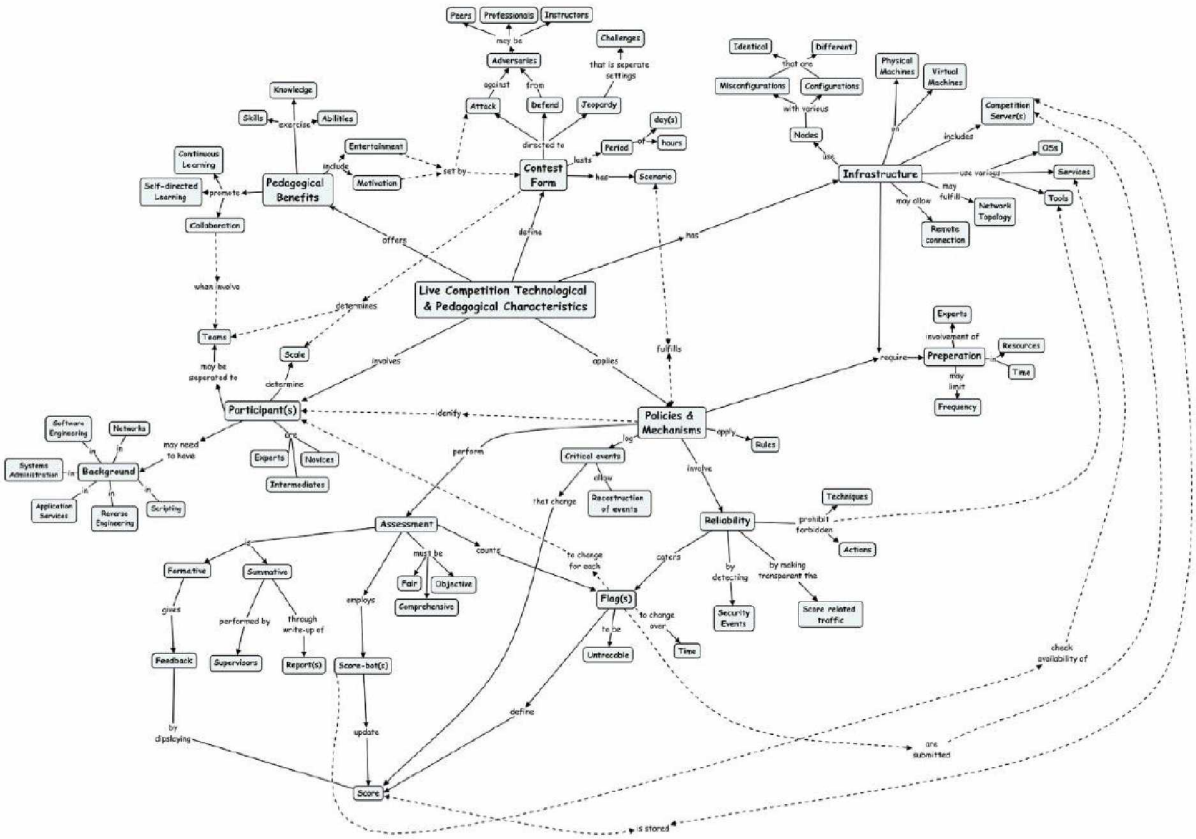

Figure 1: Concept map of live competition characteristics (from Katsantonis et al.)

Figure 1 shows how a successful live cybersecurity exercise has to balance many different aspects to make it engaging, challenging, and easy to set-up. This research showed that contest form, participants and infrastructure had the most links to other characteristics and thus shows that these aspects are often the hardest to balance. The NCCDC styled competitions focus on intermediate to expert participants, defending against professional adversaries, and reliability. These competitions also require all skills from the background characteristics and thus makes it a good template to use for all-round cybersecurity education. Where these competitions get hard is the scalability of their virtual or physical infrastructure along with the time required to prepare them. A weekend event like the CCDC also suffers from some of the issues the authors identify with short competitions such as students preparing for short term solutions like aggressive log storage without analyzing the stain it may have on limited storage space or CPU power. A weekend-long competition also does not focus on thinking about long-term security policies such as patching or supporting legacy applications but the CCDC competitions account for this by introducing the dynamic of injects (business tasks) that include tasks that range from updating software, creating backup infrastructure to creating a backup plan.

The CCDC encompasses more characteristics from Figure 1 than simpler, more focused competitions like CTFs or jeopardy styled competitions. While CTFs cover a wider range of participants (experts for the DefCon CTF to novices for the NYU CSAW CTF), they only tend to focus on the attack side of computer security and limit themselves to very specific topics that mostly center around scripting and reverse engineering. The jeopardy styled competitions like Panoply bridge this gap by scoring based on how well you can capture a flag and then defend that service to prevent another team from replacing your flag with theirs and provide a range of services that could satisfy novices and experts alike. Panoply and other jeopardy style games are a good combination of the pros and cons of CCDC and CTF styled competitions. These competitions still suffer from not training participants for long term goals like considering resource limitations or keeping services running for an extended amount of time.

It seems like the ideal exercise would take the pros from all these 3 different approaches and merge them into one exercise so a CCDC styled competition that also allows you to attack other teams and plant flags in their services while defending your network from other teams and handling injects. While great for participants of all levels, this would be a hard competition to pull off on the organization and infrastructure aspects but a solution that merges traditional class homework such as performing research into vulnerabilities or writing planning documents (e.g. upgrade policies) to simulate CCDC style injects would make a practical and engaging curriculum. This hybrid approach will add more work for the instructor so it is crucial for the success of the program to find ways to reduce this workload. This

need to create an easily deployable, extensible, and configurable system is what led to the idea of this project.

All the research into the idea of gamification of cybersecurity education points towards 3 big issues: identifying a curriculum, gauging and maintaining engagement, and developing an easy to deploy and modify system to run these exercises. The NCCDC method has been proven to work for maintaining engagement as can be seen from its growth from a pilot project with 5 participating teams in 2005 to more than 235 teams in 2020. The NCCDC is a good model but requires lots of manpower and machine power to set-up and run and is not feasible as a model to teach a typical college class with a single instructor, a varying number of students, and the potential need to run multiple exercises over a semester, McDaniels et al., solved this issue by packaging all aspects of their system into a single virtual machine image. This approach allowed them to rapidly deploy copies for their camp participants along with making it easier to share their system with others who may want to use it.

Scope

Taking inspiration from CTF competitions and the NCCDC CCS client, The ARSSE project is designed to provide a CCDC style experience that can be designed to train a single student or a team of students while providing them with live feedback and being easy for instructors to design and deploy. ARSSE is designed keeping the following goals in mind:

1) Easy to use
2) Engaging
3) Highly customizable and extensible
4) Scalability

The Oxford English Dictionary defines "Gamification" as "*the application of typical elements of game playing (e.g. point scoring, competition with others, rules of play) to other areas of activity, typically as an online marketing technique to encourage engagement with a product or service*". ARSSE borrows elements from a typical CTF and the NCCDC to provide point-scoring and competition with others by providing students with discrete challenges to compete for points and letting all students see their ranking and points. ARSSE uses gamification to teach computer security in an engaging manner.

System Architecture

Overview

As mentioned earlier, setting up an NCCDC styled exercise with a simulated network takes weeks and is not suitable for a university class where you may need multiple labs every week or the ability to change labs/homework rapidly. ARSSE solves this issue by decoupling the challenge creation system from the actual challenge machines themselves. This allows the instructor to set up machines that have more challenges than those being scored and add them to the scoring system if the initial lab is deemed too easy, or remove challenges that are found to be too hard for students. The live scoreboard and ability to see what challenges have been completed by students will empower the instructor to make informed decisions to dynamically change their exercise. ARSSE achieves this by separating the scoring from the student controlled machines as shown in the image below

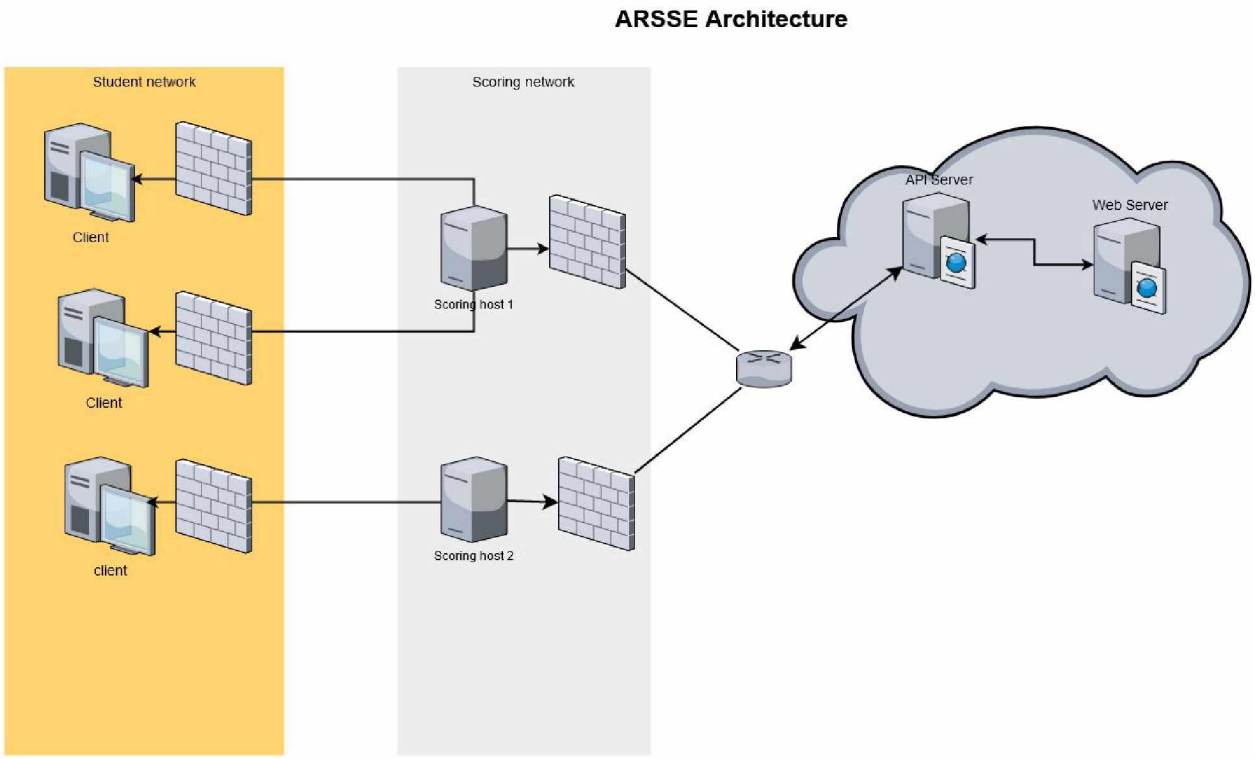

Figure 1: ARSSE Architecture

Figure 1 shows the three different networks that compose the ARSSE system: the student network (StuNet), scoring network (ScoreNet), and cloud network (CloudNet). For now, let us ignore the cloud network and focus on the "Student network" in the yellow background and the "Scoring network" in light gray because this is where the separation that allows for quick reconfiguration of challenges happens.

Design Rationale

Any system that involves grading students must be highly available and possess high data integrity.

Data integrity is hard to achieve since any process on a machine controlled by the students can be considered to be hostile. Trusting any data coming from these machines added a requirement that the ARSSE system must be able to prevent the students from manipulating scores but this is hard since the students have full access to the machines and can potentially modify the score data gathered by any scoring mechanism running on the machines in StuNet. The UTSA CCS client solves this issue by making their system closed source and adding hard to enforce regulations that disallow modifying the CCS program and any related files. The ARSSE architecture solves this data integrity issue by moving all scoring to machines that are not controlled by the students (i.e. machines in ScoreNet). This architecture ensures data integrity since students should never have access to the scoring machines and that scoring code is not run on StuNet machines. This does add the requirement that students must always allow some kind of remote access from ScoreNet into StuNet, this is a fair compromise since Inspec uses standard protocols like SSH and WinRM.

In today's world, most computer networks are behind Network Address Translation (NAT) and firewalls. Some firewalls could be very restrictive and only allow common traffic like web browsing. This limitation restricts the deployment of StuNet machines to a network that allows SSH or WinRM access from the ScoreNet machines. This often means that both StuNet and ScoreNet will have to either live in the same virtual environment or internal network. This is a fair compromise to provide data integrity and also provides the added benefit of reduced scoring latency.

Based on the research of prior work and the author's experience with the UTSA CCS system, this approach of moving the scoring to a network and hosts that are controlled by the students and only need very limited access into the competition network is a novel architecture. Competitions like CCDC and Panoply use external scoring engines to test services but are only restricted to services available externally (e.g. a website) and cannot run checks (e.g. checking if a user exists) that require access to the student machines. The ARSSE architecture combines the ease of use of an external scoring system along with the

flexibility of checking anything on the machine that would come with a system based on running the scoring on the students' machine.

Scoring Network

The scoring network consists of one or more scoring nodes. A scoring node is a machine (preferably Linux) running Chef InSpec (InSpec) and the ARSSE scoring module. InSpec was chosen since it is an actively maintained and well-documented tool that is designed to verify the current state of a machine against a predefined intended state of the machine. These properties meet the easy to use and highly customizable and extensible goals of ARSSE.

InSpec uses a Domain Specific Language (DSL) that is easy to read and write and does not require any programming knowledge. This DSL makes it easy to use but advanced users can extend its functionality by writing custom Ruby code. InSpec provides built-in resources that cover many commonly inspected parts of a machine (filesystem, firewalls, users, registry keys, etc) that can be used to check compliance.

```
describe port(22) do
  it { should be_listening }
  its('addresses') { should include '0.0.0.0' }
  its('protocols') { should cmp 'tcp' }
end
```

Listing 1: InSpec example for the port resource

Listing 1 shows a simple example to check if the target machine has TCP port 22 listening on all interfaces. As shown, the InSpec tests are very human-readable and easy to write. These tests are also operating system (OS) independent so can be reused for multiple machines running various versions of Linux and Windows.

Ensuring the integrity of the scores is important to make ARSSE viable in an academic setting or NCCDC styled competition, ARSSE achieves this by isolating the ScoreNet from the StuNet to the maximum extent possible and using InSpec's ability to check hosts remotely via SSH or WinRM. This allows architecting a very secure ScoreNet since each host only needs SSH or WinRM access to the StuNet and does not require any scoring services to be running on student controlled machines.

Scoring Node

A scoring node is a machine running within the ScoreNet that is set-up to run InSpec and the ARSSE scoring module. Each scoring host is designed to be as lightweight as possible with the idea being that they could be run within containers, VM's or dedicated hardware. This architecture is inherently scalable since each scoring host can be set-up to run its own InSpec profiles against its own specified targets. This can be set-up in a few different configurations but the two main ones are:

1) One scoring node per image: This setup will have one scoring per "image", an image is a single machine that is part of the StuNet. So if each student has 5 machines in their network and there are 10 students, this setup will have 5 scoring hosts with each scoring host responsible for 10 machines. This setup will run 1 InSpec profile against 10 machines per scoring run
2) One scoring node per student: Given the same number of students and images as above, this setup will have 10 scoring hosts with each scoring host being responsible for running 5 InSpec profiles per scoring run

Note: The calculation above assumes that each image will only have one InSpec profile running against but there is nothing stopping multiple profiles from being run against the same host. An example of this would be running a profile that checks for valid users against all images, in this case, you could be running 2 profiles per image since there is 1 profile for the common users and another one for everything else.

Student Network

The student network as shown in Figure 1 represents the network given to a single student so if we had 10 students then the overall system would comprise of 1 scoring network and 10 student networks (StuNet). We'll call each machine in the StuNet a client. A client has no ARSSE infrastructure running on it and is completely isolated from the ScoreNet apart from having to allow SSH or WinRM connections to a selectable port from the ScoreNet scoring hosts.

Cloud Network

The cloud network(CloudNet) is separate from ScoreNet and student. This is where the Web frontend for students to view current scores and the web frontend for the instructor to add new challenges will be set up. For ease of use, both these components are suggested to be placed on a public-facing

network so the scores can be viewed and the challenges managed from anywhere in the world. The minimum requirement for the ARSSE scoring to work is outbound TCP 80/443 access from ScoreNet to CloudNet and the same from StuNet so students can see their scores and what challenges they got scored for. The architecture diagram in Figure 1 shows the API and web server for the web frontends as running on different machines but there is no restriction that they cannot be on the same host. CloudNet does not need any access into StuNet or ScoreNet and thus both StuNet and ScoreNet can live behind NAT.

Communication Protocol

All communication between the scoring hosts and the API is done via JSON messages sent to HTTP endpoints. This approach allows ScoreNet to be behind NAT. The web frontends will also consume endpoints on the same API. The API uses the correct HTTP verbs to signify how an operation may affect the data, for example, a GET request will never change data and a POST request will always be expected to change data. Endpoints that are not supposed to change data will respond to POST requests with HTTP 405.

Results

As mentioned above, intensive performance testing was not performed due to resource limitations. But simple tests showed promising results, simple timing tests for small InSpec profiles showed that a majority of the time spent was due to network latency between the ScoreNet and StuNet.

```
control 'test-sshd-port' do
  title 'Server: Check SSH server config security'
  impact 0.6
  desc 'SSH server mst be running on port 22'
  describe sshd_config do
       its('Port') { should cmp 22 }
  end
end

control 'test-ssh-root-login' do
  title 'Server: Check if root is allowed to login over SSH'
  desc 'Root shoujld not be able to login over SSH'
  impact 0.7
  describe sshd_config do
       its ('PermitRootLogin') { should cmp 'no' }
  end
end

control 'Nginx should not be running' do
```

```
  title 'Nginx should not be running'
  desc 'Webserver Nginx should not be running'
  impact 0.3
  describe service(nginx.service) do
       it { should_not be_running }
  end
end

control 'Nginx should  be disabled' do
  title 'Nginx should be disabled'
  desc 'Webserver Nginx should be disabled'
  impact 0.3
  describe service(nginx.service) do
       it { should_not be_enabled }
  end
end
```

Listing 2: InSpec profile used for timing tests

Listing 2 shows the InSpec profile that was used for timing tests to show whether the approach used in this project is fast enough to provide feedback useful for a fast-paced learning environment. This profile was run against a DigitalOcean server located in San Francisco from my laptop located in Reston, Virginia. The experiment setup was a simple 2 machine set up with my laptop acting as a single scoring node that was responsible for a single StuNet machine.

Task	CPU Time (s)	Wall Clock including SSH (s)
Code 2 InSpec profile execution time	0.7	9.3
Dev-Sec Linux Package Baseline InSpec profile execution time(Dev-Sec)	2.2	11.7
SSH to server		1.11

Table 1: SSH connection and InSpec timing results

InSpec profiles return how long they took to run with the results. This built-in timing feature was used to measure the CPU time. Experimentation found that this timing did not include the setup time (Time taken to establish the SSH connection for this test) so we ran some rudimentary testing to measure the overall time taken. As seen from the results in Table 1, a comprehensive test developed to test a machine against industry-standard recommendations will take approximately 11.7 seconds to run in a scenario where the scoring node and StuNet machines are located on opposite sides of the United States. The difference between just adding the InSpec profile execution time and the SSH connection time and the actual time taken to run the InSpec tests over SSH may include time to startup the ruby interpreter,

parse the profile files, time taken to return the results (in JSON) over the network amongst other unknown tasks. Since we can safely assume that the SSH setup time will be lower when the scoring nodes and StuNet machines are within the same virtual environment or in the same building as is expected in a classroom or CCDC styled competition, these results are expected to be lower when run in an actual exercise. This shows that this remote scoring strategy of using InSpec over SSH is viable.

Conclusion

This project aimed to find out if creating an open-source, scalable, and easy to deploy system to score cybersecurity exercises was possible. The big hurdle towards creating an open-source project was ensuring data integrity since access to source code could make it a little easier for students to manipulate the scoring. This issue was solved by the ARSSE architecture which moved the scoring infrastructure away from the student machines and onto a secure network controlled by the instructor. Next was scalability, this challenge was met by designing a distributed architecture and the scoring software to be easy to set-up. Verifying scalability means running various tests in different configurations. However, due to limited resources, this was not possible but the scalability looks promising based on limited testing and the fact that InSpec is used by big organizations for IT compliance purposes.

InSpec depends upon profiles written as individual ruby files. This file-based approach was integrated into the ARSSE scoring module to provide a consistent configuration interface to the entire system. A scoring node is configured by placing the appropriate InSpec profiles and private SSH keys (for SSH access) into a predefined folder structure. The scoring module will create the required folders and the base directory can be configured easily. The InSpec results are returned as JSON files in the same directory structure and everything is grouped by the IP address of the machine being scored. This pure file-based approach removes the need for complex dependencies such as a database to store the configuration for the scoring node.

Future Work

The ARSSE project establishes a framework that can be used for scoring gamified cybersecurity exercises. This project shows that it is possible to build a scalable, easy to use, and secure system that relies heavily on well supported open-source tools to provide a fast feedback-based learning environment for defensive cybersecurity exercises. While all the individual pieces shown in the architecture diagram

are available and will soon be released on Github, these pieces could not be hooked together to make a production-ready system due to time limitations. Releasing the work as open source allows anyone who is interested in this project to continue this work and build the system into something that may one day be used to score cybersecurity training exercises.

While performance has been tested, future work could also involve more thorough testing and contributing recommendations on how to best distribute scoring nodes for best performance.

References

A. Nagarajan, J. M. Allbeck, A. Sood and T. L. Janssen, "Exploring game design for cybersecurity training," *2012 IEEE International Conference on Cyber Technology in Automation, Control, and Intelligent Systems (CYBER)*, Bangkok, 2012, pp. 256-262. http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=6392562&isnumber=6350542

Benjamin D. Cone et al "A video game for cyber security training and awareness" Computers and Security 26, Feb 2007. https://doi.org/10.1016/j.cose.2006.10.005

Communications, Raytheon Corporate. “Raytheon: Raytheon Sponsors Nation's Largest Cybersecurity Competition - Mar 9, 2017.” *Raytheon News Release Archive*, Raytheon, 18 Apr. 2015. http://raytheon.mediaroom.com/2017-03-09-Raytheon-sponsors-nations-largest-cybersecurity-competition

Conklin, A. Conceptual Analysis of Cyber Security Education Based on Live Competitions." 2006 39th Hawaii International Conference on System Sciences. http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=1579743

“Gartner Says 5.8 Billion Enterprise and Automotive IoT Endpoints Will Be in Use in 2020.” *Gartner*, Gartner, 29 Aug. 2019.

www.gartner.com/en/newsroom/press-releases/2019-08-29-gartner-says-5-8-billion-enterprise-and-automotive-io

Dev-Sec. “Dev-Sec/Linux-Baseline.” *DevSec Linux Baseline*, github.com/dev-sec/linux-baseline.

G.Parekh et al. "Identifying Core Concepts of Cybersecurity: Results of Two Delphi Processes." *IEEE Transactions on Education, Education, IEEE Transactions On, IEEE Trans. Educ*, no. 1, 2018, p. 11. EBSCO*host*, doi:10.1109/TE.2017.2715174. http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=7967715

HG and CV The Cybersecurity Jobs Report 2017, Herjavec Group, 2017. www.herjavecgroup.com/wp-content/uploads/2018/07/HG-and-CV-The-Cybersecurity-Jobs-Report-2017.pdf

Huang, Wendy Hsin-Yuan, and Dilip Soman. “A Practitioner’s Guide To Gamification Of Education.” *A Practitioner’s Guide ToGamification Of Education*, University of Toronto, 10 Dec. 2013. rotman.utoronto.ca/behaviouraleconomicsinaction/files/2013/09/GuideGamificationEducationDec2013.pdf

L. McDaniels, E. Talvi, B. Hay. “Capture the Flag As Cyber Security Introduction”, 2016 49th Hawaii International Conference on System Sciences. http://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=7427865

M. Carlisle, M. Chiaramonte, D. Caswell, "Using CTFs for an Undergraduate Cyber Education", *2015 USENIX Summit on Gaming Games and Gamification in Security Education (3GSE 15)*, 2015. https://www.usenix.org/node/191760

M. Katsantonis, P. Fouliras, and I. Mavridis, "Conceptual analysis of cyber security education based on live competitions," *2017 IEEE Global Engineering Education Conference (EDUCON)*, Athens, 2017, pp. 771-779. http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=7942934&isnumber=7942803

Pittman, Jason M. and Pike, Ronald E. "An Observational Study of Peer Learning for High School Students at a Cybersecurity Camp." *Information Systems Education Journal*, vol. 14, no. 3, 01 May 2016, pp. 4-13. EBSCO*host*. https://eric.ed.gov/?id=EJ1136076

Prensky, Marc. "Chapter 1." *Digital Game-Based Learning*, by Marc Prensky, McGraw Hill, Columbus, OH, 2001.

Welcome to Panoply. cyberpanoply.com/event.html