AUTHOR	Contreras Vargas Berioska, Virtanen Seppo
TITLE	A cloud-based systematic cyber security teaching and learning framework for high school students
YEAR	2023
DOI	https://doi.org/10.1504/IJKL.2023.132163
VERSION	Preprint
CITATION	Vargas, B. C. & Virtanen, S. (2023) A cloud-based systematic cyber security teaching and learning framework for high school students. *International Journal of Knowledge and Learning* **16**:316–339.

A Cloud-based Systematic Cyber Security Teaching and Learning Framework for High School Students*

Berioska Contreras Vargas†

Department of Electronics, Federico Santa María Technical University, Chile, berioska.contreras@usm.cl

Seppo Virtanen

Department of Computing, University of Turku, Finland, seppo.virtanen@utu.fi

ABSTRACT

We present a systematic framework to construct cybersecurity knowledge for high school students on cloud computing. The learning outcomes were analyzed based on time and completion rates to evaluate a ranking, learning curves, and clusters. The present cyber security education module consists of a series of adaptive assignments anchored on constructivist and scaffolding learning theories. The probability of success for active guided tasks was 0.65, in the case of constructive adaptive assignments was 0.55, and interactive adaptive assignment was 0.25. The completion rate dropped in the last adaptive assignment implying the need to reinforce interactions earlier. Students demonstrated that motivation is as much determinant as the background experience showing noticeable variations in the time needed to complete our adaptive assignments.

Keywords and Phrases: security and privacy; cybersecurity; penetration testing; teaching and learning; constructivist learning theory; scaffolding learning; active learning; high school; cloud computing.

Declarations of interest: none.

This research did not receive any specific grant from funding agencies in the public, commercial, or not-for-profit sectors.

*Article Title Footnote needs to be captured as Title Note.
† Author Footnote to be captured as Author Note.

1 INTRODUCTION

Cybercrime and malicious attacks against online systems have become an everyday threat. Many times, the attacks are made possible due to vulnerabilities in the systems resulting from out-of-date software and hardware and the lack of awareness thereof in the targeted organization. One reason for the lack of cyber security awareness lies in the amount and depth of information technology and cyber security education obtained. To raise cyber security awareness in young adults for the future workforce, education in this field should already be included in the curriculums of secondary level schooling.

Furthermore, in the 21st century the global social disruptions, climate change and migration are considered complex phenomena having multidimensional consequences in education. Certainly, a highly diverse social environment is reflected inside the classroom which defies contemporary school pedagogy (Shaker, 2016). Throughout this work is addressed the challenging diversity of knowledge in the cases where ICT is not an integral part of the general study of the participating students.

The learner interests differ geographically and are also influenced by distributed administrations and the school's community (Anonymized, 2016). To exemplify, recent international curriculum reforms have introduced Information and Communications Technology (ICT) to the general curriculum, and its implementation varies from school to school and takes distinct forms, such as ICT applied to education and ICT designed to education (Anonymized, 2017a). Using ICT and online learning platforms has been shown as an efficient way to increase the self-study effort of students and to improve the learning outcomes (Anonymized, 2009).

Continual reforming is gradually enhancing education, although the reforms barely tackle teaching and learning innovation as a continuous process. This work introduces a teaching and learning framework to accomplish both the interests of 20 students' participants and a given educational curriculum. The presented framework is put to practice in the form of a course implemented at a European an upper secondary school in Finland, where 85% of the students' participants were attending an ICT specialization track and 15% were attending a general curriculum.

In this article we present a cloud-based systematic cybersecurity framework that is designed, implemented, and assessed. The framework core is anchored in constructivist (Papert, 1980) and scaffolding learning theories (Anonymized, 2020) and it progresses cyclically, developing adaptive computing problems of gradually increasing difficulty. The hypothesis is that the combination of such learning methods can reconcile the level of diversity in the students' existing knowledge and their motivation. We evaluate how efficacious the combination of constructivism and scaffolding is to design adaptive assignments. The learning outcomes in our framework are measurable and prove effectiveness from different dimensions: a ranking, learning curves, and clusters. The assignments were released iteratively, populating a cloud surface for each student to enable access from any place without overloading the student's own machine. The adaptive assignments were composed of 16 computing problems assessed in terms of time and completion. The experimental initiative consisted of 20 sessions distributed across a 10-weeks period.

The educational interest in the subject of cyber security responds to the students' enthusiasm for learning skills to protect themselves. Discussion on ethical hacking is in a sense controversial in high schools because the composite statement expresses two opposing meanings; a hacker aims to circumvent restrictions while an ethical ICT specialist would not. Our framework confronts such discrepancy by reinforcing the constructive perspective of penetration testing instead of the disruptive perspective of hacking. Penetration testing is a process that simulates attacks against a computing system to evaluate and revise its security. A penetration tester has strong programming and computer networking skills.

The rest of this article is organized as follows. In section 2 a discussion of learning theory and related work is provided. In section 3, we define and discuss penetration testing in the context of the work presented in this article. Section 4 presents the details of our cyber security teaching and learning framework. Section 5 discusses the design of the infrastructure needed to test the framework in practice and provides a description of the deployment process. The results of applying the framework in practice to implement a cybersecurity course in an upper secondary school are analysed in section 6. Finally, in section 7 we provide concluding remarks of the presented work.

2 LEARNING THEORY AND RELATED WORK

From cognitive sciences and according to the learning theory of early childhood by Jean Piaget in 1970, knowledge is shaped through a process of formation of permanent objects in infancy (Papert, 1980). Such entities are assimilated from the surrounding environment, and then turn into adaptable schematics or frames that become inherited specific knowledge. Since the environment is mutable, then, the building blocks that sustain such a structure of knowledge also are.

Seymour Papert, an American mathematician, adhered to the Piaget's theory and extended it into the principle of cognitive development. As the collection of frames are reorganized to build new structures, they are also preserved due to a logical and emotional connection. The affective factor is expressed in the appropriation of knowledge by the student, which was proven in his new teaching and learning method of mathematics named LOGO Educational Programming Language (Papert, 1980).

On the other hand, the scaffolding method consists of understanding the degree of support and mentoring the students require to develop the skills needed to solve an assignment. Assuming that the learner needs are inside a zone of proximal development (ZPD), as defined in Lev Vygotsky's theory (Anonymized, 2020), the scaffolding also resides in such a zone to guide students towards independent problem-solving. The degree of guidance is a relevant discussion in game-based learning methods. A game-based instruction is a familiar concept in cybersecurity where an apprentice plays the role of a hacker and/or defender to sharpen his or her skills.

2.1 A Modern Case of Applied Constructivism

The inception of instructional constructivism occurred in the 1970s. A learner-centered approach became a common principle between constructivist educators (Matthews, 2003; Ahtee and Pehkonen, 1994). Constructivism is rooted in the interactions to construct either individual or collaborative knowledge. Throughout constructivist learning the students examine a phenomenon from multiple dimensions increasing their abstract and critical thinking. Students construct the meaning rather than merely reproduce content. The teacher becomes the builder of the learning environment and provides the student's opportunities to actively engage in self-regulated learning and collaborative learning practices.

In 2017, a constructivist method was evaluated in Israel for undergraduate students pursuing educational studies (Alt, 2017). The aim was to evaluate the effectiveness of constructivism on a diverse group of students mainly characterized by ethnicity and gender. The learning environment was combined by lecture-based courses and student-centered seminars. By means of a perceived evaluation, the social activity factors explained 52% of the contribution to openness to diversity variance, while teacher-interaction represented only 14%. The teacher-student interaction and social activity were found positively connected to the constructive activity. Complementary, the teacher-student interaction explained 60% of the constructive activity variance when 8% was explained by the social activity. The work also revealed that collective constructions require explicit conceptual information to lessen relativism.

2.2 A Modern Case of Applied Scaffolding

In 2013, a scaffolding method was tested to integrate learning content within a business game experience (Barzilai and Blau, 2014). The work of Barzilai and Blau in Israel was evaluated on how scaffolding supports the construction of relations between formal learning and game enjoyment. Following the game narrative, the scaffold is designed as an advance organizer to unfold the concepts used in the game. A pre and post problem-solving assessment is carried out under three conditions: the students who only play without scaffolds, the students who study first through scaffolds and play after, and the students who play first and then study the scaffolded content. The results revealed that the students performed better when the game was scaffolded before. In addition, the results showed a decrease in the perceived learning of the game which is an expected effect. Hence, the game was not as much determinant in the problem-solving performance as the enjoyment that remained practically uniform. The scaffolds are designed as online study units preserving the game context and extending the meanings by connecting concepts to formal knowledge representations, such as mathematical equations. In conclusion, a scaffold bridges the gap between tacit knowledge of the game and the formal knowledge of the school which is not intuitively constructed by the elementary students. So, the mathematical-based game created for education was scaffolded by harmonic instructional materials to fulfil the learner needs.

Similarly, a technology created in South Korea for computer science education was tested to engage elementary students in coding skills. In this experience, the creative problem-solving capability slightly increased by integrating App Inventor as a coding learning environment. Nevertheless, the students argued that limited time was assigned to build and review their applications. Another shortcoming was identified in the lack of instructional material. An explanation of the issue was found in the simultaneous study of the block-based programming subject and the App Inventor usage (Seong-Won and Youngjun, 2014).

2.3 Relevance to Presented Work

The scientific observations of Piaget over his own offspring alter the traditional conception of writing and reciting as a learning method for children. The entire educational community becomes the environment from where a child learns or constructs his or her knowledge. The teacher-centered approach is questioned, yet it was that approach which stimulated critical thinking to propose a learner-centered approach as an eventual successor. Consequently, we assume in this framework both principles coexist.

Educators should devote much of their time to inquiring and improving their teaching and learning methods to respond to dynamic educational requirements. For instance, the programming language introduced by Papert is a virtual environment that allows the students to interact with a computer to construct geometrical solutions individually or in teams. Thus, the behaviour of the virtual environment relies on the educator who plans the problems to be solved.

A constructivist education method without adherence to a given context and purpose becomes amusing rather than truly educative. To exemplify, it is proven that the game experience was not determinant in formal learning (Barzilai and Blau, 2014), but it was once scaffolds were used to guide the students. Scaffolding is produced by observing the participants, so the educator adapts the level of support required by the students, which is variable.

In the work presented in this article, a constructive adaptive assignment includes pertinent scaffolds, and an interactive adaptive assignment acts as a computing problem in which students have no instructions. The computing problems are influenced by the capture-the-flag (CFT) hacking approach (Anonymized, 2017b; Seong-Won and Youngjun, 2014), yet it differs in that our adaptive assignments aim to awake coding skills and the students' participants present multiple solutions.

A capture the flag (CTF) tournament (Cutas et al., 2019) is linked to gaming experiences. The ecosystem of vulnerable systems for training, cyber academies and cyber ranges has grown in the last decade (Garcia, 2019). Nonetheless, it is arguable if virtual tournaments enable knowledge construction due to the standardized nature of the set of contests. To exemplify, a CTF game tends to verify a matching condition per each problem solved, so a single predefined answer is possible (Garcia, 2019).

Since upper secondary level students are exposed to computing problems with a limited formal base in computer science, they have less bias for approaching issues in unexpected and creative ways. Throughout our framework the student is stimulated to reflect, explore, and inquire to get a realistic perception of the problem. At the end, problem solving, and critical thinking are the foremost competences to develop in which this framework is focused on.

3 PENETRATION TESTING

Penetration testing aims to simulate an intrusion on a system or network to evaluate and revise the efficacy of defensive mechanisms, as well as end-user adherence to security policies. By simulating a break-in, the intruder's methods are analysed to identify indicators of an anomaly (Farmer and Venema, 1993; 2001; Hutchins et al., 2011).

A pen-tester seizes the weaknesses that may exist in operating systems, services, databases, and applications due to flaws, improper configurations, or risky end-user behaviour by simulating an intrusion. Starting a penetration testing requires clear objectives, scope, and authorization to avoid adverse effects on the confidentiality, integrity, and availability of the information that we intend to protect. A penetration test is also referred to as pen-testing, ethical hacking, red teaming, and vulnerability testing (Gordon, 2015).

3.1 Origin and Purpose of Pen-testing

Penetration testing intends to prove a break-in condition on a computing system following a set of tactics, techniques, and procedures (TTP) (Strom et al., 2018). Having a myriad of attack vectors becomes difficult to agree upon one attack methodology. Nevertheless, the attackers tend to reuse tools and techniques to downturn the cost of executing an intrusion (Hutchins et al., 2011). Similarities are found in multiple attacks allowing a set of processes to be modelled. Let us say a compromised system contains the traces left by an attacker, and then such traces are factual data to construct a pattern.

The precursors in the prevention of intruders by using break-in examples were Farmer and Venema (Garcia, 2019; Farmer and Venema, 1993). In 1993, the developers examined the traces left by a set of intrusion techniques they performed on their own systems to demonstrate the impact. In 1999, Ross D. from Microsoft published the first paper demonstrating a script injection into web content. The publication extended the study about Cross Site Scripting (XSS) conducted over Internet Explorer, as the issue proved a successful exploitation from the server side (Grossman et al., 2007).

In 2007, the Department of Defense of the United States published an attack method called Kill Chain based on the following set of 6 stages; Find, Fix, Track, Target, Engage and Assess. Such kill chain method aimed to enforce intelligence and defensive efforts by thoroughly understanding an intrusion (Hutchins et al., 2011). By increasing the adversary's cost at a certain stage of the attack, the entire adversarial effort can be disrupted.

In 2011, the Lockheed Martin Corporation introduced a modern intelligence-driven computer network defense (Hutchins et al., 2011). The method embraces a set of 7 stages referred to as Reconnaissance, Weaponization, Delivery, Exploitation, Installation, Command and Control (C2) and Actions on Objectives. The Reconnaissance stage seeks, identifies, and selects a target. The Weaponization stage spans the embedding of a payload inside a trusted file that may carry a trojan and exploit. The Delivery stage conveys the weaponized file through email, websites, or USB devices. Throughout the Exploitation stage triggers the weapon that seizes an eventual flaw or weakness. The Installation stage implies a successful execution of a weapon that brings remote control to the intruder. In the Command and Control (C2) stage the compromised system reaches the intruder through the Internet establishing a persistent channel. Lastly, the Actions and Objectives stages occur when the information becomes exposed to disclosure, alteration or even destruction.

The intelligence-driven model points out the defender's reaction that tends to precipitate preventive actions cutting off the leverage of recognizable indicators of intrusion. By extending the analysis on earlier stages, the defensive controls are leveraged by factual indicators of compromise improving the detection rate onwards. The traces left during reconnaissance, weaponization and delivery are essential for defensive purposes, including the reconstruction of the attack (Ahtee and Pehkonen, 1994).

At present, the penetration testing methodologies are diverse as the testing is frequently tailored by the pen-tester according to the dynamic conditions of the environment. An example is the Open-Source Security Methodology Manual OSSTMM v3 introduced by ISECOM in 2010 (Pete Herzog, 2010), the Penetration test framework introduced and maintained by Kevin Orrey (Orrey et al., 2014). In addition, two other methodologies are the Penetration Testing Execution Standard (PTES) proposed by PTES Group in 2009 (PTES Group, 2009), the ISO/IEC/IEEE 29119 Software and System Engineering - Software Testing released in 2013 (ISO, 2019), the methodology described in the CISSP common body of knowledge (Adam Gordon, 2015), and the Technical Guide to Information Security Testing and Assessment SP-800-115 published by NIST in 2008 (Scarfone et al., 2008).

A suitable degree of abstraction is given by the framework published by the NIST guide which consists of a set of 4 stages: Planning, Discovery, Attack and Reporting. The Planning stage focuses on defining the goals and scope. The Discovery stage covers the information gathering and scanning to identify and analyse vulnerabilities. The next Attack stage addresses the exploitation of selected targets or the alteration of the system's state to induce persistency (Foster, 2005). Lastly, the Reporting stage documents not only the findings of the previous phases but also the recommendations to mitigate the risk exposed. Consequently, the methodology adapted to the presented work considers a set of four stages as follows:

1. **Reconnaissance and Discovery**: Searching for any available information on the target avoiding intrusive methods.
2. **Enumeration and Vulnerability Analysis**: Intrusive data gathering. Mapping of known vulnerabilities.
3. **Execution and Appropriation**: Attempt to gain temporary or permanent users and privileged access based on an attack plan and access strategy.
4. **Document Findings**: Document the security state of the environment. It provides the test results by means of empirical evidence and remediations.

3.2 Penetration Testing Reflections

At present, the Privacy Rights Clearinghouse claims that more than 11 billion of records has been breached since 2005 (PRC, 2005). The global reality of hacking has evolved alarmingly, shaping an undeniable black market (Ferbrache, 2016) where information is traded and where talented teenagers are not exempt to partake.

Usually, software follows a development life cycle, although high-quality software is not built from the unique dimension of functionality. A functional increment of software could barely withstand an attack when countermeasures are omitted during the design and development, becoming expensive any amendment once in production. Despite elaborated attacks against cryptographic controls, usually an attacker exploits code´s deficiencies, interface issues, lack of verification, weak settings, deprecated libraries and so for. A designing error could evolve as a vulnerability once is deployed.

Through development, the static and dynamic code analysis are well known techniques collaborating in the loop of refactoring and improvement. In the testing stage the recurrent security techniques are manual or automatized penetration testing, vulnerability scanning and fuzzing tests. A pen-testing simulates an attack sending data to the application and inspecting the behaviour and impact.

An intrinsic factor in any attack strategy is the exploitation or the factual validation of a vulnerability. Such successful exploitation allows us to recognize that every outsider attacker becomes an insider at the break-in point of the first layers of security. Penetration testing contributes to revising periodically the existing security measures that become a contribution when the mitigations are diligently implemented. If so, the enhanced countermeasures increase the adversarial time effort to drop off the attack rate.

4 THE TEACHING AND LEARNING FRAMEWORK

The next section embraces the research process of the cloud-based systematic cybersecurity framework for high school students.

4.1 The Student Participants

The selected number of students for our experiment is 20: where 85% were attending an ICT track and 15% were enrolled in the general curriculum track. By breaking down the number of student participants, the proportion of female students reached 15% and male students 85%. In terms of grade levels, the students enrolled in 2015 were 20%, those from 2017 were also 20%, and those from 2018 represented 45%, making it a diverse group.

4.2 Testable Adaptive Assignments

The adaptive assignments are scaffolded by formal learning and practical tasks to state a computing problem. Throughout experimentation, a student reasons about a security problem and a plausible solution. Table 1 summarizes the teaching and learning outcomes designed and developed by the framework.

Subject Stage	Lecture and Guided Tasks	Adaptive Assignment
General	Lectures 1 and 2	No Assignment
	Tasks 1 and 2	
Reconnaissance and Discovery	Lecture 3 and 9	Assignment 1
	Tasks 3 and 9	Assignment 10.1
Enumeration and Vulnerability Analysis	Lecture 4 and 8	Assignment 2
	Tasks 4 and 8	
Execution and Appropriation	Lectures 5-7 and 10-14	Assignment 3-5
	Tasks 5-7 and 10-14	Assignment 6-9
Document and Findings	Lecture 15 and 16	Assignment 10.2
	Tasks 15 and 16	

Table 1: Teaching and Learning Outcomes

4.3 Teaching and Learning Classification

The learners explore the subject of study in a hybrid teaching and learning environment where they initially interact with the instructor and then they continue an individual series of guided tasks to finally construct a solution collectively. A cloud-based learning environment is developed where learners' interactions occur cyclically. The teaching and learning outcomes are classify in terms of interactions (Moore, 1989), as the next table illustrates:

Teaching & Learning	Student-Content	Student-Instructor	Student-Student
Lectures	Promotes the interaction between the student and the subject of the study directed by the educator.		
Guided Tasks		Educator encourages and maintain students' motivation through experimental tasks bringing feedback.	
Adaptive Assignments			Promotes the interaction between students to solve problems either individually or collectively.

Table 2: Teaching and learning classification based on interactions

A cloud computing extends the face-to-face learning environment and the technology applied on it towards a smart environment accessible for students located in distant areas that have Internet access. The framework core methods can cyclically occur per each session to approach a multidiscipline, such as cybersecurity. According to the conceptual framework introduced by Chi [30], we can differentiate activities from the students' perspective for further analysis as the next table unfolds:

Teaching & Learning	Active	Constructive	Interactive
Guided Tasks	Activate existing knowledge, Assimilate, encode, or store new information. Encourage searching of existing knowledge.		
Adaptive Assignments		Infer new knowledge, Integrate new information with existing knowledge. Organize own knowledge for coherence. Repair own faulty knowledge. Restructure own knowledge.	Creating processes that incorporate a partner's contributions (joint dialogues).

Table 3: Teaching and learning classification based on ICAP concepts

From this perspective, a content-based lecture is regarded to passive learning that directs the subject of study and context, and the dialogue is individual or inclined to a one-way communication which is evidently surpassed by active learning. Considering the notion that an interactive method surpasses a constructive one, and the former is greater than the active method, then it is conceivable the logical statement: passive < active < constructive < interactive, as in Chi [30]. Nonetheless, a pure interactive method for upper secondary students, would likely obviate the prior concepts and knowledge required to interact effectively. Therefore, this work tested a cyclical combination of methods to encourage both critical thinking and problem-solving. The learning outcomes are rated according to the following table:

Framework Core	Active rate	Constructive rate	Interactive rate
Guided Tasks	5 points per student [Tasks 1.1-5.1		
Adaptive Assignments		9 points per student [Assig.1.2-5.2], [Assig.6-9]	7 points per student [Assig.10.1-10.2]

Table 4: Teaching and learning rates

5 INFRASTRUCTURE DESIGN AND DEPLOYMENT

The selected infrastructure is based on a public cloud computing service. A set of requirements are defined to reduce the risk of an eventual misuse of the technology and unintentional information disclosure. Moreover, as the initial number of students was

unknown, the size of the virtual system was estimated per single student rather than multiple students accessing a shared environment at the time. The case of simultaneous sessions was already experimented (Anonymized, 2017b), where students performed penetration testing techniques concurrently and experienced reiterative interruptions over the planned infrastructure. Therefore, the capacity and flexibility of the infrastructure became a crucial success factor.

The learning environment is implemented over the cloud, and it consists of two elements: the multiple instances of virtual systems, and a replica of a productive system which is one instance. The number of multiple instances is equivalent to the number of students, so it is one instance per student. One student's instance embraces five virtual systems or testing targets where only one of them is published on the Internet to receive Secure Shell (SSH) requests remotely. The replica instance is a web application reachable from any student instance to be tested.

5.1 Tasks and Assignments Preparedness

The preparedness is a process that progressively enables the tools and components required. The process starts by narrowing the scope for the guided tasks and assignments, as in Figure 1. The next step is the building of the scripts that are categorized as shell scripts and powershell scripts (Robbins, 2018). The script's codes are tested in one instance. Once the result is functional, the scripts are finally deployed to all the student's instances, otherwise are refactored.

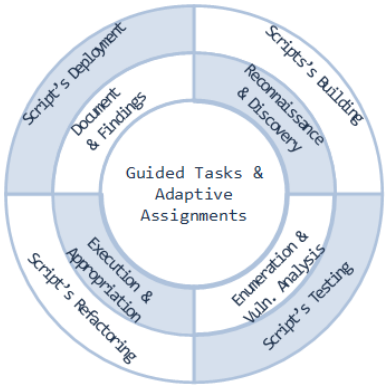

Figure 1: Adaptive Cybersecurity Teaching and Learning Framework

5.1.1Adaptive Framework Deployment

The learning environment is mutating through the continuous deployment of the scripts. An instance over the cloud is a group of virtual resources that can be replicated and associated. Some scripts are organized per guided task and assignment separately although they are frequently blended for efficiency. A more accurate categorization is to group them as shell scripts running on a Linux operating system, and a complementary group of powershell scripts running on Microsoft Windows system. The powershell scripts act as triggers to run the instructions in a finite loop for remote participants. Table 2 presents a shell script sample that covers the deployment of the Web Application Attack and Audit Framework W3af (Riancho, 2008). To exemplify, the student discovers a condition prone to injection and demonstrates the case of exploitation applying certain tool and/or code.

Line	Snippet Code
1	export SSHPASS='secret'
2	cat <<_EOT_ | sudo -s
3	/usr/bin/sed -i 's/# deb-src/deb-src/' /etc/apt/sources.list
4	/usr/bin/git clone https://github.com/andresriancho/w3af.git $home/workspace/w3af
5	/tmp/w3af_dependency_install.sh
6	/usr/bin/setfacl -m g:team:rwx -R /home/hack/workspace/w3af
7	_EOT_

Table 5: Script sample w3af deployment

Table 3 shows a snippet code that brings some familiarity with Scapy library over Python (van Rossum, 1990). The threats emphasized are eavesdropping (Combs, 1998) and man in the middle attack through Address Resolution Protocol (ARP) poisoning technique (Marlinspike, 2011; Plummer, 1982). The following code is an adaptation written in Python language (Biondi et al., 2008) that builds a forged ARP frame.

Line	Snippet Code
1	def send_arpspoof(IFACE_MAC,IFACE_IP,V_IP,GW_IP):
2	V_MAC = get_MAC(IFACE_MAC,V_IP)
3	GW_MAC = get_MAC(IFACE_MAC,GW_IP)
4	#<<< Complete the Missing Parts denoted by dashes>>>
5	send(ARP(hwdst=---,pdst=---,hwsrc=--_MAC,psrc=--_IP,op=2),iface='eth1')
6	send(ARP(hwdst=---,pdst=---,hwsrc=--_MAC,psrc=--_IP,op=2),iface='eth1')
7	#<<< End of Missing Parts >>>
8	IFACE_MAC, IFACE_IP = get_if_hwaddr('eth1'), get_if_addr('eth1')
9	V_IP, GW_IP = "10.1.1.29", "10.1.1.25"

Table 6: Snippet code sample ARP frame

Table 4 presents a snippet code about a Failure to Preserve SQL Query Structure according to the Common Weakness Enumeration community (CWE, 2006; Howard et al., 2010). An attacker can insert a malformed data into parameters controlled by the user, and by processing such insertion the application can change the semantics of a coded SQL statement (OWASP, 2017).

Line	Snippet Code
1	$mystatement = "SELECT * FROM users WHERE userid = $myuid ";
2	$myquery = mysqli_query($mysqli, $mystatement);
3	while ($mydata = mysqli_fetch_array($myquery))

Table 7: Snippet code sample SQL weakness

Deployment is based on iterations pushing the coded outcomes to a set of student's instances. Table 5 shows a powershell script sample that covers a deployment step:

Line	Snippet Code
1	$myArray = ("op1", "op2", "op3",…, "opN")
2	New-Variable -Name "PW" -Visibility Public -Value "secret"
3	Foreach ($op in $myArray){
4	.\scp -i id_rsa.rsa -pw $PW .\script* mentor@std-instance-$op-geo.azure.com:/home/mentor
5	.\ssh -i id_rsa.rsa -pw $PW mentor@std-instance-$op-geo.azure.com 'sudo <command>'}
6	Remove-Variable -Name "PW" – Force

Table 8: Snippet code sample iterative script

5.2 Infrastructure Reflections

Constructivist learning implies highly dynamic testing environment, and the nature of penetration testing involves numerous tactics, techniques, and tools so both can exhaust the computational resources of a student's personal computer.

It is acknowledged that by building systems over a virtualized platform costs are saved related to hardware, cooling, and electricity. However, such virtualization is still limited to the hardware of the host system, which is likely not sized to have frequent

peaks of computer power consumption. Then, a cloud computing environment becomes a highly scalable option providing interfaces to build dynamic settings, which is implemented in this work.

Inside the cloud we built multiple individual instances per student, protecting his or her learning environment by means of cryptographic keys (Barker and Roginsky, 2018). An appropriate public key verification occurs between their personal computer and the learning environment providing confidentiality. Isolated learning environments allow the students to practice at his or her own pace from home with minimal impact on their own computers. In addition, individual environments on the cloud are deployed from prepared templates, in this way a faulty environment is just discarded and a fresh one is spawned in order of minutes.

Certainly, an individual surface implies responsibilities, so the students understand that they are liable citizens and they agreed upon the usage of the technology. Thus, any attack or analysis outside the scope defined is considered unauthorized.

Multiple attack tools and techniques exist, and the most relevant ones were chosen as a complementary element, since the purpose is to awaken a maker mindset. For such endeavours, the open-source communities are preferred in this work as they are the main booster in making free software.

6 ANALYSIS OF RESULTS

Throughout the assignment assessment it is possible to improve the teaching and learning methods. Complementary to the student's feedback, his or her solution's timestamp is tracked. The timing is computed using modular arithmetic to identify statistical properties.

The timing computation allows to solve an eventual tie in the scored ranking, to project meaningful learning curves, and to visualize insightful clusters of comparative patterns.

6.1 Quantitative Method

Assignments are announced progressively, so we can measure a period $\boldsymbol{\delta}\mathtt{t}$ from the moment of announcement $\mathtt{t}_{\mathbf{0}}$ and the student's solution $\mathbf{t}_1$ is notified.

$$\boldsymbol{\delta}\mathtt{t} = \mathtt{t}_1 - \mathtt{t}_0 \quad (1)$$

A modular reduction of the timestamp is a convenient way to accelerate computation. Let $\boldsymbol{\alpha} = \mathbf{mq} + \mathbf{r}$, and $\boldsymbol{\alpha_i} \boldsymbol{\epsilon} \mathbf{Z}_n$. We adopt the timestamp format '$\mathbf{YYYYMMDD}\boldsymbol{HH}\mathbf{mm}$', according to the RFC-3339 conventions (Klyne and Newman, 2002) and treat it as a series of integers. Then, each timestamp is reduced in a modular way: $\boldsymbol{\delta}\mathtt{t} \rightarrow \mathtt{t}_{\mathbf{i}}$ and $\mathtt{t}_{\mathbf{i}} \boldsymbol{\epsilon} \mathbf{Z}_n$, where $\boldsymbol{i}=\{1,2,3,\ldots,\boldsymbol{n}-1\}$. For instance, given $\mathtt{t}_1 = 201903141532$ and $\mathtt{t}_0 = 201903121400$, then $\mathtt{t}_0, \mathtt{t}_1 \boldsymbol{\epsilon} \mathbf{Z}_{12}$, and we derive:

$$\begin{aligned} YYYY &= (t_i \, mod \, 10^{n} - t_i \, mod \, 10^{n-4}) \, / \, 10^{n-4} \\ MMDD &= (t_i \, mod \, 10^{n-4} - t_i \, mod \, 10^{n-8}) \, / \, 10^{n-8} \\ HHmm &= (t_i \, mod \, 10^{n-8} - t_i \, mod \, 10^{n-12}) \, / \, 10^{n-12} \quad (2) \end{aligned}$$

It is known that the solutions for assignments 1 to 5 tend to be solved in less than a day, then $\mathtt{t'}_1 - \mathtt{t'}_{\mathbf{0}}$ must be zero as the timing is not exceeding a month according to the following expression in (3). If $\mathtt{t'}_1 = (\mathtt{t}_1 \, mod \, 10^{n-4} - \mathtt{t}_1 \, mod \, 10^{n-6}) \, / \, 10^{n-6})$ and $\mathtt{t'}_0 = (\mathtt{t}_0 \, mod \, 10^{n-4} - \mathtt{t}_0 \, mod \, 10^{n-6}) \, / \, 10^{n-6})$, therefore:

$$t'_1 - t'_0 = 0 \quad (3)$$

Let us say a day is equivalent to 2400 units, then we formulate (4) based on (3) as follows:

$$\begin{aligned} \mathtt{t''}_1 &= [(t_1 \, mod 10^{N-6} - t_1 \, mod 10^{N-8}) \, / \, 10^{N-8}] * 2400 + HHmm_1 \\ \mathtt{t''}_0 &= [(t_0 \, mod 10^{N-6} - t_0 \, mod 10^{N-8}) \, / \, 10^{N-8}] * 2400 + HHmm_0 \\ t''_1 - t''_0 &= \delta t \quad (4) \end{aligned}$$

Otherwise, if $\mathtt{t'}_1 - \mathtt{t'}_{\mathbf{0}} > 0$, then we assume the solution exceeded a month of 31 days and it is derived as follows:

$$\begin{aligned} \mathtt{t''}_1 &= [(t_1 \, mod 10^{N-6} - t_1 \, mod 10^{N-8}) \, / \, 10^{N-8}] * 2400 + HHmm_1 \\ \mathtt{t''}_0 &= [31 - (t_0 \, mod 10^{N-6} - t_0 \, mod 10^{N-8}) \, / \, 10^{N-8}] * 2400 + HHmm_0 \end{aligned}$$

All unsolved assignments received 0 points without penalties, yet for further clustering analysis all those zero values are treated as a -1 constant value.

6.2 Preliminary Hack Rate Analysis

Table 6 displays the results in the same order of the adaptive assignments. The hack rate represents the percentage of assignments solved, and the other complementary indicators are based on a $\boldsymbol{\delta}\mathtt{t}$ defined in (1) in order of days:

Assign.id	Min.	Max.	Avg.	Sdv.	HackRate
1.1	0.132	9.840	1.396	2.00	0.8
1.2	0.169	9.184	1.706	2.00	0.65
2.1	0.034	29.32	3.690	8.00	0.55
2.2	0.015	33.48	5.190	9.00	0.65
3.1	0.013	28.48	5.749	10.0	0.55
3.2	0.013	22.21	3.755	7.00	0.45
4.1	0.002	30.27	10.02	11.0	0.4
4.2	0.265	15.02	10.45	6.00	0.2
5.1	0.065	38.48	6.630	13.0	0.35
5.2	0.065	3.809	1.328	2.00	0.3
6	4.965	11.22	7.053	3.00	0.15
7	1.384	2.394	2.129	0	0.2
8	3.093	9.130	6.111	3.00	0.1
9	0.209	0.431	0.329	0	0.15
10.1	3.088	15.32	8.721	5.00	0.2
10.2	5.758	7.798	6.617	1.00	0.25

Table 9: Table of Hack Rate Results

The hack rate results steadily decrease while the assignments shift from system and network security towards web application security. The complexity increases due to two factors: the penetration testing methodology expansion as a process and the shifting in the scope's subject. Accordingly, 80% of the students were able to execute a brute force attack, whilst 10% of the participants proposed an attack vector for the SQL injection case. The fastest solution for the brute force attack arrived about 3 hours after the assignment's announcement, and the average performance was about one day.

Assignment 5 denoted a transition between scopes, and it focuses on the study of Hypertext Transfer Protocol (HTTP) message exchanging. At that moment, there was a whole view of the penetration testing methodology, and the students started to combine techniques. The fastest answer arrived in about an hour, and on average the answers were solved the next day.

Multiple answers arrived in less than 60 minutes in which the fastest solution was achieved in just 4 minutes.

A relevant insight is exhibited in the adaptive assignment 7 where students proposed different alternatives to prove an injection flaw. The results arrived between one and two days after the announcement. The performance was ratified in assignment 9 where a Cross Site Scripting injection (XSS) (Grossman et al., 2007) was solved in roughly 5 hours, and the average response period remained less than one day.

The final assignment started by traversing directories to analyse the composition of the targeted web site. The students selected different techniques to discover and enumerate the components and provided their answers between 3 days and up to 2 weeks after the announcement. In this experience, the students spontaneously started to collaborate and share insights among them since the case was emulating a blind scenario with limited information. The students were able to understand the authentication logic. The backend provided debugging data intentionally which is avoided in a productive environment, yet a student necessarily requires critical thinking and problem-solving capability to prove a break-in. The students proposed their own attack strategy, altering the existing conditions of the website to induce a forged token from the client side. The findings were provided on average one week after the announcement and according to the reporting requirements.

6.3 Learning Curve Analysis

In cognitive psychology, a learning curve is understood as a continual improvement produced by extensive practice (Ritter and Schooler, 2002). The time effect due to the numerous trials can be modelled mathematically based on a power law formula that is simplified and given in (5)

$$y = \alpha_n x^b, \quad n \varepsilon N = \{1, 2, \ldots, 10\} \qquad (5)$$

Where $\boldsymbol{\gamma}$ is the cumulative average time per attempt, and $\boldsymbol{\alpha}$ is equal to the average time of the whole class to solve the assignment $\boldsymbol{n}$, and $\boldsymbol{b}$ represents a logarithmic slope of a given learning effect rate, and $\boldsymbol{x}$ is the total number of attempts. We assumed a learning effect rate of 80% that is distributed in log of 2 to fix a constant of activity (6).

$$b = log_{10}(0.8)/log_{10}(2) \qquad (6)$$

Therefore, the projected learning curves were graphed for the representative framework core methods.

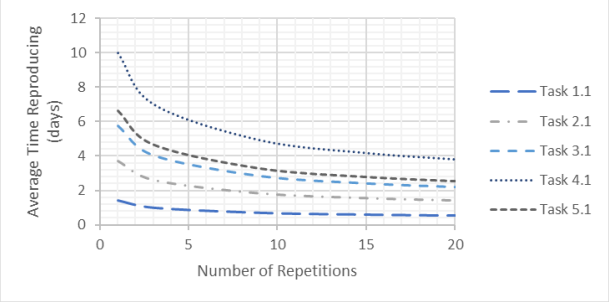

Figure 2: Learning curve: active guided tasks.

The Figure 2 shows the learning curves of active guided tasks (scaffolding) where solutions arrive in less than a week while the penetration testing methodology is approached per single stage with focus on systems and network security. A notable acceleration is shown in the curves through the first 10 repetitions, then remain at steady pace.

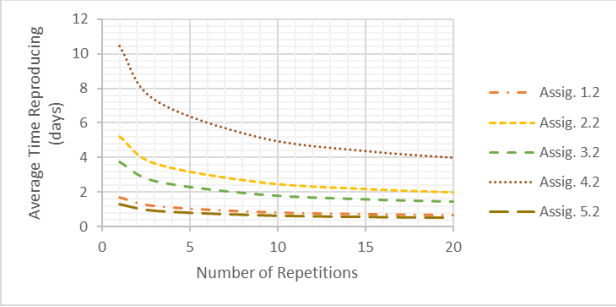

Figure 3: Learning curve: constructive adaptive assignments.

Figure 3 displays the learning curves of the first five adaptive assignments complementing the guided tasks to enhance construction of knowledge. The trend exhibits a slightly faster accomplishment from the beginning, and the projected repetitions revealed a moderated acceleration in the first 5 attempts that remain quite constant.

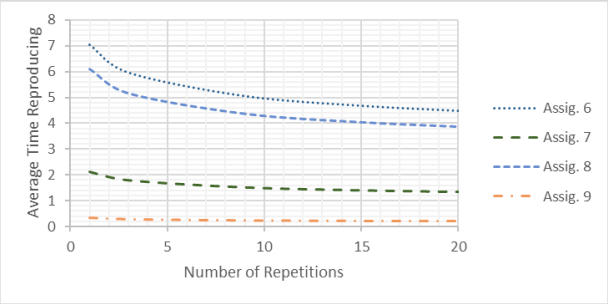

Figure 4: Learning curve: constructive adaptive assignments.

Figure 4 displays the learning curves of the last five adaptive assignments at a moment when the penetration testing methodology and the subjects of the system, network and application security are fully integrated. A regular curve closer to zero represents the lowest completion rate, immediately followed by the opposite case where the solution arrived in a couple of days. The degree of dispersion in the second half of assignments is notorious, and the projected repetitions reveals a steadily acceleration in the first 10 attempts, that decreases and remain regular subsequently.

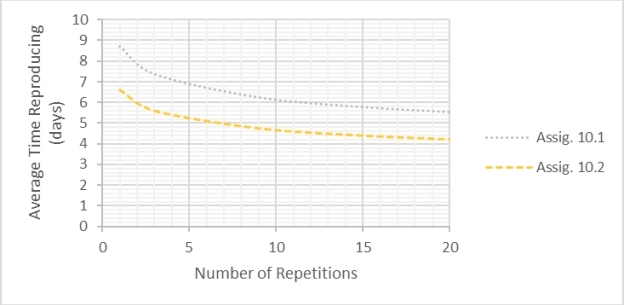

Figure 5: Learning curve: interactive adaptive assignment 10.

The Figure 5 shows the learning curves of the adaptive assignment 10 where the pen-testing methodology is seen as a full process and the subject turned to application security. This is considered as the most difficult stage, and a joint dialogue is expected to conceive multiple perspectives of the problem.

The learning curves exhibit an optimistic gain in performance as they are based on the whole class average instead individual repetitions. Constructive assignments in parallel to guided tasks were able to reduce scattering trends, yet without such scaffolds the constructive methods uncover dispersion.

6.4 Student Ranking and Assignment Classification

The assessment on time basis proved to be useful to break a tie of any exact match of points earned by the students or teams. We recall that formula (1) defined an average period $\boldsymbol{\delta}\mathtt{t}$ that was calculated per student. The tiebreak in the student's ranking represents the arithmetic mean of $\boldsymbol{\delta}\mathtt{t}_{\mathtt{i}}$ where $\boldsymbol{i}=\{1,2,3,\ldots,\boldsymbol{n}\}$, and n=16 owing to the combination of guided tasks and adaptive assignments.

The unsolved assignments received 0 points without penalties, and the arithmetic mean considered those zero values without further adjustments as some students were less interested in competition. Table 7 exhibits the students ranking sorted by points and tiebreak values, and it is extended by the classification of methods: guided tasks (scaffolds), constructive adaptive assignments and interactive adaptive assignments.

Student Id	Points	Tiebreak	Year Enrolled	Guided Tasks	Constructive A.A.	Interactive A.A.
student 8	20	75117	2017	5	8	7
student 11	18	107946	2018	5	9	4
student 17	18	111500	2018	5	9	4

Student Id	Points	Tiebreak	Year Enrolled	Guided Tasks	Constructive A.A.	Interactive A.A.
student 5	17	21176	2016	5	5	7
student 6	16	91795	2018	5	5	6
student 4	9	121421	2018	5	4	0
student 1	6	23563	2016	3	3	0
student 7	6	314998	2018	3	3	0
student 16	6	1483994	2017	5	1	0
student 9	5	31995	2018	2	3	0
student 14	4	10396	2018	2	2	0
student 12	4	50700	2018	2	2	0
student 18	3	320109	guest	3	0	0
student 13	2	500	2017	1	1	0
student 3	2	10849	2016	1	1	0
student 2	2	19630	2016	1	1	0
student 10	0	0	2018	0	0	0
student 15	0	0	2017	0	0	0
student 19	0	0	guest	0	0	0
student 20	0	0	guest	0	0	0

Table 10: Table of Final Ranking and Assignment Classification

The probability to accomplish from 2 up to 5 active guided tasks is in the interval between 0.35 and 0.65, so the probability interval is 0.30. Similarly, in the case of constructive assignments, the probability of accomplishment is in the range from 0.45 up to 0.55, thus the probability interval is narrowed to 0.10. The interactive constructive assignment reached a probability of success p=0.25. Hence, the completion rate of active guided tasks is greater than the constructive assignments, and constructive is greater than the interactive one, in contrast to the assertion that entirely interactive learning is the greatest method (Chi, 2009). The last assignment requires a strong collaboration between participants that occurs only in the case of the top five students who also maintained a consistent rate of completion for all assignments.

In terms of attendance, there were 3 formal requests for dropping out of the course. The main reason expressed was the time involved, an aspect that is variable considering the different course load of each student in the period. Also, some students decided to join the course to understand how to protect themselves from hacking, and they were highly committed with their attendance yet less interested in the accomplishment of tasks and assignments.

6.5 Student Cluster Analysis

Throughout the assessment, the best student is subject of individual evaluation (student 8) who from the ranking perspective achieved 20 points by solving everything except assignment 8. In fact, the team which student 8 was member of completed assignment 10 creatively, and teammates of student 8 credited him with the successful strategy. On average, student 8 was also faster than the three closest finalists who solved all the assignments, and assignment 10 partially. Hence, the best student patterns can be analysed to form clusters based on number theory.

The reduction of $\boldsymbol{\delta t}$ was already introduced in formulas (4) and (5), and $\boldsymbol{\delta t}$ is a series of integers where i={1, 2,…,n}, and n=20 represents the number of participants. If $\boldsymbol{\delta t} < 0$ the assignment was unsolved.

If $\boldsymbol{\delta t_i} > 0$, then $\boldsymbol{\delta t_8}$ is the completion time of student 8. By observing $\boldsymbol{\delta t_8}$, we note that assignments 1 to 5 were solved in less than a day each, and assignment 10 in about 7 days. Thus, the analysis occurs in $\boldsymbol{k}$ clusters, and $\boldsymbol{k}$ =3. If the clusters form a vector $\boldsymbol{c}$= {1,2,3}, then every element in $\mathbf{c}$ is mapped to all elements in $\boldsymbol{\delta t_i}$ iteratively.

Thus, we calculate $\boldsymbol{\delta t'_i}$ = average($\boldsymbol{\delta t_i}$) for all values of **i** in the vector $\mathbf{c_i}$. Then, an interpolation is computed using formula (8), the sum of squares of two vectors $\boldsymbol{x}$ and $\boldsymbol{\gamma}$:

$$\boldsymbol{\Sigma}(\boldsymbol{x} - \boldsymbol{y})^2 \quad (8)$$

The clusters were computed and visualized, and for each of them a finalist student in the ranking was chosen for evaluation: student 11, student 8 and student 5.

Figure 6 shows a comparison between the student ranked in position 2 (student 11), and the best student (student 8). So, $\boldsymbol{\delta t_{11}}$, $\boldsymbol{\delta t_8}$ $\boldsymbol{\varepsilon}$ $\boldsymbol{x}$= {$\boldsymbol{\alpha}$, $\boldsymbol{b}$}, and $\boldsymbol{\delta t'_{11}}$, $\boldsymbol{\delta t'_8}$ $\boldsymbol{\varepsilon}$ $\boldsymbol{\gamma}$= {$\boldsymbol{c}$, $\boldsymbol{d}$}. It is noticeable that student 11 was highly effective despite completion time variable with respect to the best student.

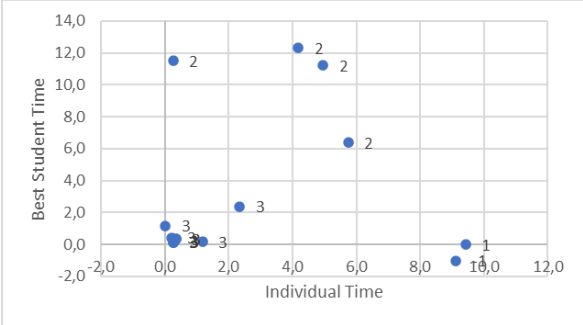

Figure 6: Cluster analysis: student 11.

Figure 7 shows a comparison between the best student (student 8) and the completion time average of all participants. So, from $\boldsymbol{\delta t_8}$, $\boldsymbol{\delta t_i}$ $\boldsymbol{\varepsilon}$ $\boldsymbol{x}$= {$\boldsymbol{\alpha}$, $\boldsymbol{b}$}, and $\boldsymbol{\delta t'_8}$, $\boldsymbol{\delta t'_i}$ $\boldsymbol{\varepsilon}$ $\boldsymbol{\gamma}$= {$\boldsymbol{c}$, $\boldsymbol{d}$}, it is noticeable that student 8 was highly effective and efficient with respect to the whole class.

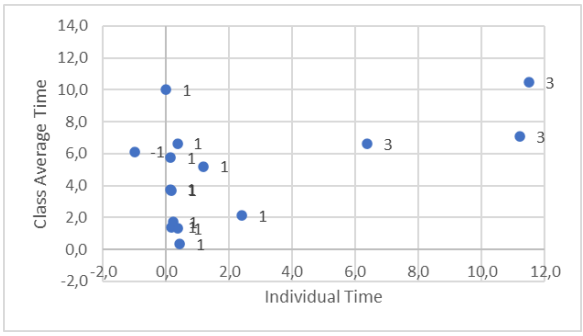

Figure 7: Cluster analysis: student 8.

Figure 8 shows a comparison between the student ranked in position 4 (student 5) and the minimum completion time of the whole class. So, from $\boldsymbol{\delta t_5}$, $\boldsymbol{\delta t_m}$ $\boldsymbol{\varepsilon}$ $\boldsymbol{x}$= {$\boldsymbol{\alpha}$, $\boldsymbol{b}$}, and $\boldsymbol{\delta t'_5}$, $\boldsymbol{\delta t'_m}$ $\boldsymbol{\varepsilon}$ $\boldsymbol{\gamma}$= {$\boldsymbol{c}$, $\boldsymbol{d}$}, it is noticeable that student 5 was highly efficient with respect to the whole class, this is clear in the marginal scattered solutions.

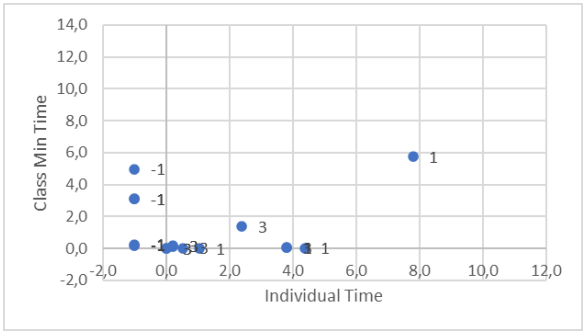

Figure 8: Cluster analysis: student 5.

Consequently, we discard that the most experience students follow a perfect linearity considering the sample of Students 5 and 6 who lost traction from assignment 6 onwards. Also, some students were highly effective, having variations; for example, Student 11 and 17 being enrolled to upper secondary school just one year before the course.

7 CONCLUSIONS

The assessment demonstrated to be objective to state that a motivated high school student has the potential to learn and solve computing problems regarded to cyber security. By being immersed in an adaptive digital environment, the students have a more realistic view of the problems that we face in cyber security, and how they can help to solve them creatively, and without overloading or exposing their own machines or personal data.

Through the application of our framework, it was proven that the highest probability of success for active guided tasks (scaffolds) was 0.65, and in the case of constructive adaptive assignments was 0.55, whilst interactive adaptive assignment was 0.25. The last interactive assignment required a strong collaboration between participants that occurs only in the case of the top five students who also maintained a consistent rate of completion for all assignments.

The completion rate dropped in the last adaptive assignment implying the need to reinforce interactions on earlier cycles. For instance, introduce joint dialogues in the scaffolding or guided tasks which are focus on instructional dialogue of the educator and individual feedback.

The completion time is computed using modular arithmetic to identify statistical properties and produce clusters. The results allow educators to discover the degree of diversity of participating students early to calibrate the guided tasks and adaptive assignments. We assessed that an overall average hardly explains how demanding an assignment was for a given student or the cadence used by the best student. Thus, the multiple dimensions from a ranking, the learning curves, and clusters were found to provide valuable insights to confirm the effectiveness of our framework.

Overall, the assignments 1-3 showed a significant concentration of solutions in less than one day. Throughout assignments 4 and 5, the figures dropped, and the completion time increased to up to 10 days on average. The last assignments, numbers 6-10, showed a slight decrease in completion time and ranged from 2 to 8 days of workload. By comparison, we noticed that the best student was both highly effective and efficient with respect to the whole class. The comparison also identifies that the most effective and efficient students are not necessarily the most experience ones in the class. The results demonstrated that the student's motivation is as much determinant as their background experience.

As future work, it should be of great interest to explore learning methods to enhance collaborative interactions, and to find out mechanisms to accelerate coding assessment to bring timely feedback. Regarding to the time-based assessment, it is also left to further study the data minimization aspects.

ACKNOWLEDGMENTS

[Third parties in collaboration anonymized].

REFERENCES

Genevieve G. Shaker (2016), The Global Common Good and the Future of Academic Professionals. Higher Learning Research Communications, vol.5, num. 2. [Online] http://files.eric.ed.gov/fulltext/EJ1132845.pdf (Accessed on: Dec. 14, 2018).

[author and publication details removed for anonymization], (2016), Book.

[author and publication details removed for anonymization], (2017), Bachelor's Thesis.

[author and publication details removed for anonymization], (2009). International Journal of Knowledge and Learning vol. 4, n. 6, p. 527-538.

Seymour Papert (1980), Mindstorms: Children, Computers and Powerful Ideas. Basic Books, Inc. Publishers, New York, United States of America.

[author and publication details removed for anonymization], (2020), Master's Thesis.

William J. Matthews (2003), Constructivism in the classroom: Epistemology, history, and empirical evidence. Teacher Education Quarterly Vol.30, n. 3 p.51-64. [Online] https://eric.ed.gov (Accessed on: Feb. 18, 2019).

Maija Ahtee and Erkki Pehkonen Eds. (1994), Constructivist Viewpoints for School Teaching and Learning in Mathematics and Science, Research Report 131. Department of Teacher Education, University of Helsinki. ERIC, Helsinki.

Dorit Alt (2017), Constructivist Learning and Openness to Diversity and Challenge in Higher Education Environments. Learning Environments Research Vol.20, n. 1, p. 99-119. [Online]https://eric.ed.gov (Accessed on Feb. 18, 2019).

[author and publication details removed for anonymization], (2017), Building Virtual Penetration Testing Environment. Master's Thesis.

Kim Seong-Won and Lee Youngjun (2014), A Study of Educational Method Using APP Inventor for Elementary Computing Education. Journal of Theoretical and Applied Information Technology, JATIT vol.95, n. 18.

Sarit Barzilai and Ina Blau (2014), Scaffolding game-based learning: Impact on learning achievements, perceived learning, and game experiences. Science Direct, Computers & Education Journal, vol. 70, p.65.

Felicia Cutas, Claudio Ardagna, Kostas Lampropoulos, Mattijs Jonker, and Neeraj Suri (2019), Assessing the courses for Cybersecurity professionals already developed by CONCORDIA partners. Cyber Security Competence for Research and Innovation, CONCORDIA. European Union.

Gorka Abad Garcia (2019), Final Degree Project: Online Penetration testing Laboratory. Computer Management and Information System, Universidad del País Vasco. Bilbao, España.

Dan Farmer and Wietse Venema (1993), Improving the Security of Your Site by Breaking into It. Usenet posting. [Online] http://www.porcupine.org/satan/admin-guide-to-cracking.html (Accessed on: Jan. 14, 2019).

Dan Farmer and Wietse Venema (2001), Being Prepared for Intrusion. Dr. Dobb's Journal, vol.26, issue 4, pp.78-85. [Online] http://www.drdobbs.com/being-prepared-for-intrusion/184404565 (Accessed on: Jan. 14, 2019).

Eric M. Hutchins, Michael J. Cloppert and Rohan M. Amin (2011). Intelligence-Driven Computer Network Defense Informed by Analysis of Adversary Campaigns and Intrusion Kill Chains. Lockheed Martin Corporation.

Adam Gordon (2015), Official (ISC)2 Guide to the CISSP CBK, 4th edition. Boca Raton, Florida, United States of America: CRC Press, Taylor & Francis Group.

Blake E. Strom, Andy Applebaum, Doug P. Miller, Kathryn C. Nickels, Adam G. Pennington, and Cody B. Thomas (2018). Mitre ATT&CK: Design and Philosophy. The Mitre Corporation, McLean, VA.

Jeremiah Grossman, Robert Hansen, Petko D. Petkov, Anton Rager, and Seth Fogie (2007). XSS Attacks: Cross Site Scripting Exploits and Defense. Massachusetts, United States of America: Syngress, Elsevier.

Pete Herzog (2010), The Open Source Security Testing Methodology Manual: OSSTMM 3. Institute for Security and Open Methodologies, ISECOM. [Online] http://www.isecom.org/mirror/OSSTMM.3.pdf.(Accessed on Feb. 11, 2019)

Kevin Orrey, M. Byrne, A. Doraiswamy, L. Lawson, and N. Ouchn (2014), Penetration Test Framework (PTF) v0.59. [Online] http://www.vulnerabilityassessment.co.uk/Penetration%20Test.html (Accessed on Feb. 11, 2019)

PTES Group (2009), Penetration Testing Execution Standard. [Online] http://www.pentest-standard.org/index.php/FAQ (Accessed on Feb. 11, 2019).

ISO (2019), Software and Systems Engineering - Software Testing. International Organization for Standardization ISO/IEC/IEEE 29119.

Karen Scarfone, Murugiah Souppaya, Amanda Cody, and Angela Orebaugh (2008), Technical Guide to Information Security Testing and Assessment. The National Institute of Standards and Technology, Special Publication 800-115. Maryland, United States of America: NIST.

James C. Foster (2005). Sockets, Shellcode, Porting, and Coding: Reverse Engineering Exploits and Tool Coding for Security Professionals. Syngress, Elsevier.

Privacy Rights Clearing House - PRC (2005), Records Breached Since 2005. [Online] https://privacyrights.org/ (Accessed on Feb. 28, 2019).

David J. Ferbrache (2016), Ruthless and Rational Cyber Criminal Entrepreneurs. The Implications for Our Cyber Security, KPMG UK.

Michael Grahame Moore (1989), Three Types of Interaction. Pennsylvania State University. American Journal of Distance Education.

Michelene T.H. Chi (2009), Active-Constructive-Interactive: A Conceptual Framework for Differentiating Learning Activities. Psychology in Education, Arizona State University. Cognitive Sciences Society.

Mike F. Robbins (2018), Powershell 101, Microsoft Documentation. [Online] https://docs.microsoft.com/en-us/powershell/scripting/learn/ps101/00-introduction (Accessed on Jul. 1, 2020).

Andrés Riancho (2008), W3af: Web Application Attack and Audit Framework. [Online] https://github.com/andresriancho/w3af (Accessed on Feb. 18, 2019).

Guido van Rossum (1990), Python Language, Python Software Foundation. [Online] https://docs.python.org/ (Accessed on Jan. 14, 2019).

Gerald Combs (1998), Wireshark: Open-Source Network Protocol Analyzer. [Online] https://www.wireshark.org (Accessed on: Feb. 18, 2019)

Moxie Marlinspike (2011), SSLStrip. [Online] https://github.com/moxie0/sslstrip (Accessed on Feb. 18, 2019).

David C. Plummer (1982), An Ethernet Address Resolution Protocol, RFC 826. Internet Engineering Task Force IETF Network Working Group. [Online] https://tools.ietf.org/html/rfc826 (Accessed on Feb. 18, 2019).

Philippe Biondi and the Scapy Community (2008), Scapy Python Library Documentation, Scapy Project. [Online] https://scapy.readthedocs.io/en/latest/index.html (Accessed on Feb. 18, 2019).

Common Weakness Enumeration CWE (2006), Improper Neutralization of Special Elements used in an SQL Command CWE-89. [Online] https://cwe.mitre.org (Accessed on: Feb. 18, 2019)

Michael Howard, Daniel LeBlanc, and John Viega (2010), The 24 Deadly Sins of Software Security: Programming Flaws and How to Fix Them. McGraw-Hill.

OWASP Foundation (2017), The Official OWASP Top 10 Document Repository. [Online] https://github.com/OWASP/Top10 (Accessed on Feb. 18, 2019).

Elaine Barker and Allen Roginsky (2018), Transitioning the Use of Cryptographic Algorithms and Key Lengths. The National Institute of Standards and Technology, Special Publication SP 800-131A Rev.2. Maryland, United States of America: NIST.

G. Klyne and C. Newman (2002), Date and Time on the Internet: Timestamps. RFC: 3339. Internet Engineering Task Force IETF Network Working Group. [Online] https://tools.ietf.org/html/rfc3339 (Accessed on Feb. 18, 2019).

Frank E. Ritter and Lael J. Schooler (2002), The Learning Curve. Penn State University. Pennsylvania, United States.