UNIVERSITY OF MACEDONIA

SCHOOL OF INFORMATION SCIENCES

DEPARTMENT OF APPLIED INFORMATICS

Cyber Security Game Based Training

Ph.D. Dissertation

Menelaos N. Katsantonis

Bachelor's Degree (BSc) in Computer Science, University of Reading, UK, 2000

Master's Degree (MSc) in Distributed Systems and Networks, University of Kent at Canterbury, 2002

Thessaloniki Greece

2021

Supervisor

Mavridis Ioannis

Professor, Department of Applied Informatics, University of Macedonia

Advisory Committee

Mavridis Ioannis

Professor, Department of Applied Informatics, University of Macedonia

Manitsaris Athanasios

Professor, Department of Applied Informatics, University of Macedonia

Xinogalos Stelios

Associate Professor, Department of Applied Informatics, University of Macedonia

Examination Committee

Mavridis Ioannis

Professor, Department of Applied Informatics, University of Macedonia

Manitsaris Athanasios

Professor, Department of Applied Informatics, University of Macedonia

Xinogalos Stelios

Associate Professor, Department of Applied Informatics, University of Macedonia

Satratzemi Maria

Professor, Department of Applied Informatics, University of Macedonia

Fouliras Panagiotis

Assistant Professor, Department of Applied Informatics, University of Macedonia

Gritzalis Dimitris

Professor, Department of Informatics, Athens University of Economics and Business (AUEB)

Rantos Konstantinos

Associate Professor, Department of Computer Science, International Hellenic University

ABSTRACT

Cyber security education is gaining more attention in the past years as cyber security incidents grow in number and fierceness. Nevertheless, there is a deficit in the cyber security workforce as the demands for skilled cyber security personnel continually rise and cyber-criminals persistently demonstrate new distinctive skills, which in many cases overwhelm the knowledge and skills of cyber security professionals. One of the key priorities to cover this deficit is to improve the effectiveness of cyber security education. Though cyber security is a complex and multidomain field and cyber security education faces many problems and challenges that downgrade its effectiveness. Specifically, current cyber-security learning and training generally use traditional teaching methods and they often utilize live competitions (e.g., capture the flag competitions), in which participants compete on their knowledge and skills. Although live competitions incorporate several pedagogical benefits when adopted in educational contexts, they are also associated with many obstacles that reduce their pedagogical value and effectiveness. Game-based approaches seem to have the potential to improve the effectiveness of cyber security education, as serious games continually gain increasing recognition and acceptance and they have already been utilized successfully in multiple fields.

In this study, the domains of cyber security education and live competitions are analyzed, and their strengths and weaknesses are identified; serious games design models and frameworks are exploited, and based on the performed analysis, the game-based approach is utilized as a vehicle to confront the cyber security education weaknesses and improve its impact. To this end, the Conceptual Framework for eLearning and Training (COFELET) is proposed, a framework for the design and implementation of cyber security serious games. The COFELET framework is based on modern learning theories and it envisages cyber security serious games as highly organized and parameterized learning environments that monitor learners' actions, evaluate their activities, adapt to their needs and scaffold their efforts. COFELET compliant games provide learners the possibilities to practice their knowledge and skills, unleash cyber-attacks and experiment in a safe environment. COFELET foresees the adoption of well-known models and strategies (e.g., MITRE's CAPEC, Lockheed Martin's Cyber Kill Chain model) generally used in threat analysis and modeling approaches that verify the validity, applicability and sustainability of the COFELET approaches. COFELET employs the COFELET ontology, which aims at constituting a universal knowledge model for cyber security e-learning and training. The COFELET ontology provides coherent descriptions of the key elements COFELET compliant games need to embrace to model the actions attackers perform to unleash cyber-attacks. Additionally, COFELET applies the COFELET game life-cycle, a roadmap that exhibits how the game's major components and the COFELET ontology elements can be organized in the structure of a COFELET game and the main actors to be involved in the development process of a COFELET compliant game.

Based on the COFELET framework a prototype hacking simulator COFELET game was developed, called HackLearn. HackLearn is a web-based serious game, which can deliver learning sessions through any web browser, anytime and anywhere without the presence of an instructor. HackLearn is a scenario-based hacking simulation game for teaching cybersecurity concepts while providing hands-on hacking experiences to the learners. HackLearn's design and implementation details are methodically presented along with a prototype scenario and a set of COFELET ontology elements.

HackLearn was evaluated twice. It was preliminary evaluated during its design phase and subsequently, it was evaluated in a class of the University of Macedonia post to its implementation. During the second evaluation, the game's perceived UX was assessed along with its effectiveness to teach penetration testing concepts and cyber-attack techniques and strategies. The results of HackLearn's evaluations suggest that it embraces several features which promote the deliverance of effective learning and training approaches. In practice it was a beneficiary addition in a real educational environment, as learners were engaged, motivated and satisfied. Conclusively, the results suggest that COFELET compliant approaches promise to enhance the impact of cyber security learning and training. Serious games can be part of a formal educational system, as students are motivated in learning in more active, creative and entertaining ways. The evaluation also revealed a few shortcomings, which suggest avenues for future research, such as the lack of multiplayer support, the need for more sophisticated scaffolding, and the lack of multi-mode operation (e.g., certification mode, competition mode etc.).

Keywords: Cybersecurity education, Game-based learning, eLearning, Training, Serious games, Cyber security, COFELET, Design, Framework, Evaluation, User experience, Ontology, Didactic framework, Concept map, ATMSG, CAPEC, Cyber Kill Chain.

ΠΕΡΙΛΗΨΗ

Η εκπαίδευση στην κυβερνοασφάλεια κερδίζει ολοένα και περισσότερη προσοχή τα τελευταία χρόνια, καθώς τα περιστατικά κυβερνοεπιθέσεων αυξάνονται συνεχώς σε αριθμό και σφοδρότητα. Παράλληλα, παρατηρείται έλλειμμα στο εργατικό δυναμικό κυβερνοασφάλειας, καθώς οι απαιτήσεις σε εξειδικευμένο προσωπικό συνεχώς αυξάνονται ενώ οι εγκληματίες στον κυβερνοχώρο επιδεικνύουν νέες δεξιότητες, οι οποίες σε πολλές περιπτώσεις ξεπερνούν τις γνώσεις και δεξιότητες των επαγγελματιών του χώρου. Μία από τις βασικές προτεραιότητες για την κάλυψη αυτού του προβλήματος είναι η βελτίωση της αποτελεσματικότητας της εκπαίδευσης στην κυβερνοασφάλεια. Ωστόσο, ο τομέας της κυβερνοασφάλειας είναι ένας πολύπλοκος τομέας και η εκπαίδευση στην κυβερνοασφάλεια περιλαμβάνει πολλά προβλήματα και προκλήσεις που υποβαθμίζουν την αποτελεσματικότητά της. Συγκεκριμένα, η εκπαίδευση στην κυβερνοασφάλεια αξιοποιεί κυρίως παραδοσιακές μεθόδους διδασκαλίας και συχνά χρησιμοποιεί τους διαγωνισμούς κυβερνοασφάλειας (συνήθως της μορφής Capture-the-flag ή CtF), στους οποίους οι συμμετέχοντες ανταγωνίζονται με βάση τις γνώσεις και τις δεξιότητές τους. Παρόλο που οι διαγωνισμοί κυβερνοασφάλειας ενσωματώνουν πολλά παιδαγωγικά οφέλη, όταν υιοθετούνται σε εκπαιδευτικά πλαίσια, συνδέονται επίσης με πολλά προβλήματα που μειώνουν την παιδαγωγική τους αξία και την αποτελεσματικότητά τους. Οι προσεγγίσεις που βασίζονται σε παιχνίδια φαίνεται να έχουν τη δυνατότητα να βελτιώνουν την αποτελεσματικότητα της εκπαίδευσης στον τομέα της κυβερνοασφάλειας, καθώς τα σοβαρά παιχνίδια κερδίζουν συνεχώς αναγνώριση, ενώ ήδη χρησιμοποιούνται με επιτυχία σε άλλους τομείς.

Σε αυτήν την διατριβή, αρχικά αναλύεται ο τομέας της εκπαίδευσης στην κυβερνοασφάλεια γενικότερα και των διαγωνισμών κυβερνοασφάλειας ειδικότερα, ενώ εντοπίζονται τα πλεονεκτήματα και τα μειονεκτήματά τους. Αξιοποιούνται τα μοντέλα ανάπτυξης σοβαρών παιχνιδιών, και με βάση την ανάλυση του κάθε τομέα, χρησιμοποιείται ως όχημα η εκπαίδευση με αξιοποίηση ψηφιακών παιχνιδιών για την αντιμετώπιση των αδυναμιών εκπαίδευσης στην κυβερνοασφάλεια και την βελτίωση της αποτελεσματικότητάς της. Για το σκοπό αυτό, προτείνεται το εννοιολογικό πλαίσιο για την ηλεκτρονική μάθηση και κατάρτιση (Conceptual Framework for eLearning and Training) με την ονομασία COFELET, ως ένας οδηγός για το σχεδιασμό και την ανάπτυξη σοβαρών παιχνιδιών για την ασφάλεια στον κυβερνοχώρο. Το πλαίσιο COFELET βασίζεται σε σύγχρονες θεωρίες μάθησης και βλέπει τα σοβαρά παιχνίδια στην κυβερνοασφάλεια ως οργανωμένα και παραμετροποιήσιμα περιβάλλοντα μάθησης που καταγράφουν τις ενέργειες των εκπαιδευομένων, αξιολογούν τις δραστηριότητές τους, προσαρμόζονται στις ανάγκες τους και στηρίζουν τις προσπάθειές τους. Τα παιχνίδια που είναι συμβατά με το COFELET παρέχουν στους μαθητές τις δυνατότητες να εξασκήσουν τις γνώσεις και τις δεξιότητές τους, να εξαπολύσουν κυβερνοεπιθέσεις και να πειραματιστούν σε ένα ασφαλές περιβάλλον μάθησης. Το COFELET προτείνει την υιοθέτηση γνωστών μοντέλων και στρατηγικών (π.χ., το CAPEC της MITRE, το Cyber Kill Chain της Lockheed Martin) που χρησιμοποιούνται γενικά σε προσεγγίσεις ανάλυσης και μοντελοποίησης κυβερνοαπειλών που επαληθεύουν την εγκυρότητα και τη βιωσιμότητα των

προσεγγίσεων COFELET. Προς αυτό το σκοπό, προτείνεται η οντολογία COFELET, της οποίας η συνδυασμένη εφαρμογή στοχεύει στη δημιουργία ενός μοντέλου γνώσης για την ηλεκτρονική μάθηση και κατάρτιση στον κυβερνοχώρο ασφάλειας. Η οντολογία COFELET παρέχει λεπτομερείς περιγραφές των βασικών στοιχείων που πρέπει να περιλαμβάνουν τα παιχνίδια που είναι συμβατά με το πλαίσιο COFELET ώστε να μοντελοποιούν τις ενέργειες που κάνουν οι επιτιθέμενοι στον κυβερνοχώρο. Επιπλέον, το πλαίσιο COFELET εφαρμόζει τον κύκλο ζωής του παιχνιδιού COFELET (COFELET game life-cycle), για να καθορίζει τον τρόπο οργάνωσης των βασικών συστατικών και των στοιχείων της οντολογίας COFELET στη δομή ενός παιχνιδιού COFELET, καθώς επίσης και τους κύριους ρόλους του προσωπικού που συμμετέχει στη διαδικασία ανάπτυξης του παιχνιδιού.

Με βάση το πλαίσιο COFELET αναπτύχθηκε ένα πρωτότυπο παιχνίδι προσομοίωσης κυβερνοεπιθέσεων, που ονομάζεται HackLearn. Το HackLearn είναι ένα σοβαρό παιχνίδι, το οποίο είναι σχεδιασμένο για να προσφέρει διδακτικές συνεδρίες μέσω προγράμματος φυλλομετρητή (browser), χωρίς να είναι απαραίτητη η παρουσία εκπαιδευτή. Το HackLearn υιοθετεί τη χρήση παραμετροποιήσιμων δυναμικών σεναρίων για τη διδασκαλία βασικών εννοιών κυβερνοασφάλειας, ενώ παρέχει στους εκπαιδευόμενους ευκαιρίες για πρακτική άσκηση και απόκτηση εμπειριών στο ethical hacking. Οι λεπτομέρειες σχεδίασης και εφαρμογής του HackLearn παρουσιάζονται λεπτομερειακά στη διατριβή μαζί με ένα πρωτότυπο σενάριο και το σύνολο στοιχείων από την οντολογία COFELET.

Το HackLearn αξιολογήθηκε δύο φορές: αρχικά κατά τη φάση σχεδιασμού του και στη συνέχεια στα πλαίσια μαθημάτων κυβερνοασφάλειας στο Πανεπιστήμιο Μακεδονίας όπου αξιολογήθηκε η αποτελεσματικότητά του στη διδασκαλία βασικών εννοιών της κυβερνοασφάλειας και τεχνικών και στρατηγικών κυβερνοεπιθέσεων, καθώς και στην απόκτηση χρήσιμων εμπειριών από τον χρήστη. Τα αποτελέσματα των αξιολογήσεων δείχνουν ότι το HackLearn περιλαμβάνει δυνατότητες που διευκολύνουν τη δημιουργία αποτελεσματικών διδακτικών προσεγγίσεων στην κυβερνοασφάλεια. Οι εκπαιδευόμενοι ήταν αφοσιωμένοι στην επίτευξη του εκπαιδευτικού σεναρίου και δήλωσαν ικανοποιημένοι από τις εμπειρίες που αποκόμισαν. Συνεπώς, οι προσεγγίσεις COFELET μπορούν να ενισχύσουν τον αντίκτυπο της μάθησης και της κατάρτισης στον τομέα της εκπαίδευσης στην κυβερνοασφάλεια. Ακόμη, φάνηκε ότι τα σοβαρά παιχνίδια μπορούν να αποτελούν μέρος ενός εκπαιδευτικού προγράμματος, καθώς οι μαθητές έχουν κίνητρο στη μάθηση με πιο ενεργούς, δημιουργικούς και διασκεδαστικούς τρόπους. Παρόλα αυτά, Η αξιολόγηση αποκάλυψε ορισμένες ελλείψεις, οι οποίες δείχνουν τις προοπτικές για μελλοντική έρευνα, όπως η έλλειψη υποστήριξης για πολλούς παίκτες (multi-player), η ανάγκη για πιο εξελιγμένη βοήθεια και υποστήριξη των προσπαθειών του χρήστη και η έλλειψη πρόσθετων τρόπων λειτουργίας (π.χ. λειτουργίες πιστοποίησης των γνώσεων του εκπαιδευόμενου και λειτουργία διαγωνισμού εκπαιδευόμενων).

ACKNOWLEDGEMENTS

Undertaking a Ph.D. at the University of Macedonia has been a life-changing adventure for me and a fulfilment of a goal that I had since my MSc degree in 2002. My Ph.D. is the outcome of tons of hard work merged with creativity, persistence, and inquisitive spirit. However, it wouldn't have been possible without the support of several people to whom I am extremely grateful. Primarily, I would like to express my deepest gratitude to my supervisor Prof. Ioannis Mavridis for giving me the chance, for his guidance and support, and for providing to me the privilege to be an active member of the MSN Lab of the University of Macedonia. Additionally, I would like to thank the staff and colleagues of the MSN Lab (especially Assistant Prof. Panagiotis Fouliras) and the members of my advisory committee Associate Professor Stelios Xinogalos and Professor Athanasios Manitsaris for their support.

I would also like to thank my wife Depi and my daughter Nikoleta that were always supportive despite my early morning studies and my stressful moods, my parents Nikos and Smaragda who as a researcher and a teacher hinted the path to this point, and my brother Dimitris for his advice and support.

TABLE OF CONTENTS

Contents

LIST OF FIGURES

LIST OF TABLES

1. INTRODUCTION

1.1. Motivation and Research Questions

The digitization of our world provokes serious implications for public security. All the aspects of peoples' lives are tightly coupled with networks and the internet. People use mobile devices and computers to become more connected to access information and services. Revolutionary technologies like cloud computing, web 2.0 and social networks bring new opportunities for attackers to exploit systems' weaknesses. As the dependence on technology grows, cyber security risks and cyber-attack casualties increase. Cyber security incidents in critical infrastructures and the private sector are continually on the rise in both numbers and fierceness. Cyber-attacks become more sophisticated as cyber criminals continually exhibit deep technical knowledge and new special skills in exploiting technological and social means. Although cyber-security is not a new field, it is continuously emerging and growing and it is becoming a bigger issue.

Cyber security personnel in companies, agencies and organizations is the frontline of defense against theft, destruction and manipulation of resources. Nevertheless, there is a deficit in skilled cyber security personnel that is expected to grow in the years to come. Moreover, cyber security professionals do not get the proper level of training or they do not keep the same pace in updating and reinforcing their knowledge and skills as cyber criminals (Katsantonis et. al, 2019). To this end, cyber security education needs to face new and ongoing challenges in its effort to satisfy the required needs of the field. Such challenges are primarily driven by the necessity for more cyber security personnel capable of facing the emerging threats and fighting the cyber criminals in terms of knowledge and competencies. According to the International Information System Security Certification Consortium (ISC, 2019), cyber security workforce needs to grow by 145% to meet the market demands. At the same time, the cyber security incidents continually rise in numbers and fierceness (Risk Based Security, 2020), affecting global economy and national security (ENISA, 2019). However, cyber-security is a difficult topic to be taught efficiently. Traditional teaching items like lectures, seminars, events and discussions might help to the deliverance of knowledge but they are not appropriate for this topic. Learners are presented with a large amount of information in a short period of time. Thus, it is not easy for the educators to keep them motivated. They do not participate in the learning process as they are passive receivers of information disconnected from the real aim of the cyber-security. Learning does not result permanent change of learner's behavior. In other words, cyber-security knowledge and skills taught, fail to be utilized when they have to be put into effect. In this context, the critical issue which motivated this study is formulated to the following research question (RQ):

RQ1: *What prevents the delivery of effective cyber security learning and training?*

Provided the context mentioned above, game-based learning approaches provide a new anchor for cyber security education, as serious games have been proven effective educational tools, already

successfully applied in other fields (e.g., healthcare (Wang, 2016)). However, game-based cyber security learning and training is a new approach. There are very few studies in the field (Hendrix et al., 2016), there is no empirical evidence on the utilization of game-based learning and serious games in cyber security education and there is lack of design standards and common methodologies (Katsantonis et al., 2017b). Thus, under this perspective two research questions were formed in this study:

> **RQ2**: *Can game-based learning and training improve the effectiveness of cyber security education?*
>
> **RQ3**: *Can cyber security game-based learning and training be supported by any means such as frameworks, methodologies, and tools?*

1.2. Aim and Objectives

The main aim of this thesis is to answer the research questions by investigating how and to what extent the utilization of game-based learning and serious games can improve the effectiveness of cyber security education. To achieve this, three objectives were formulated. Objective 1 addresses RQ1, whereas Objective 2 and Objective 3 tackle RQ2 and RQ3. The following objectives are correlated, and they informed each other during the research process:

i. **Objective 1**: *Identify the pros and cons of the provided cyber security education*.
Cybersecurity education is a highly evolving field characterized by complexity arising from its interdisciplinary nature and the need to meet special requirements for the training of cybersecurity personnel (e.g., the military, cybercrime police, etc.) so that they can cope with their duties and be ready to face new threats. Cyber security educational programs usually apply curricula and deliver learning and training programs specifically designed to satisfy the demands in cyber security personnel. Current programs usually apply traditional teaching methods (e.g., lectures, workshops, lab sessions) and they often utilize live competitions (e.g., capture the flag) in which participants compete on their knowledge and skills. Such contests have been organized for many years since the nineties. Live competitions have many pedagogical benefits when incorporated in educational contexts, but they are also associated with many obstacles that limit their pedagogical value and effectiveness. On the other hand, very few efforts have been made in utilizing cyber security serious games and game-based approaches in cyber security education. Nevertheless, the existing approaches are analyzed, and the key characteristics are identified, along with the pros and cons of live competitions.

ii. **Objective 2**: *Formulate the appropriate means for the development and support of cyber security game-based learning and training*.
Methodological frameworks provide guidelines for developing educational serious games and game-based learning approaches. Methodological frameworks are based on learning

theories and instructional strategies and support the detailed representation of educational serious games, describing the manner game elements are in the structure of the game. Objective 2 involves the elaboration of an innovative conceptual framework specifically targeting cyber security educational serious games. Moreover, as there is a lack of methodologies and standards in developing cyber security serious games, this objective requires the systematic review of the domain-independent educational serious games design frameworks discussed in the literature and the identification of the key concepts that should be taken into consideration, when designing such approaches. The proposed framework needs to reconcile learning theories, serious games development frameworks that fuse learning the game's mechanics and remedies to the problems of cyber security education identified in Objective 1. The work done in the context of this objective needs to be validated with the evaluation of the proposed framework through its practical application in a real educational context.

iii. **Objective 3**: *Provide support for the reuse of cyber security serious games' elements*.
The elements of the conceptual framework, elaborated for the fulfillment of Objective 2, should be incorporated in a modular fashion in cyber security serious games and be able to be reused. In such way, the development of cyber security serious games will be facilitated, and on-demand solutions can be created efficiently, cost-effectively and fast. Besides, these elements may encapsulate pedagogical aspects and particular models and methodologies of the cyber security field that are difficult to be developed from scratch. Thus, game developers will focus on the game's features and mechanics, but they will need support on how to incorporate them into their games.

1.3. Contributions

The study presented in this thesis is multidisciplinary, as it combines knowledge from the fields of game-based learning and serious games, cyber security, and pedagogy and learning theories. The central contribution of this research is the Conceptual Framework for eLearning and Training (COFELET) framework, a framework for the design and implementation of serious games and game-based approaches. The COFELET framework promotes game-based approaches that envisage the improvement of cyber security education pedagogical effectiveness by embracing modern learning theories and innovative teaching approaches. Although the thesis focuses on cyber security serious games, the COFELET framework can also be utilized in other research areas outside the field of cyber security. The knowledge contributions of this thesis are presented below:

1. A thorough review of the state-of-the-art of cyber security education, particularly pertaining to the utilized methods and learning theories, and the pros and cons. The literature review also included the review of the domain of live competitions, which is an important part of cyber security education. The domain of live competitions was analyzed, and the key

characteristics were recorded and categorized along with related problems. Based on the conducted literature review, the concept map of live competition technological and pedagogical characteristics was formed along with a categorization of live competitions problems.

2. A review of the studies of cyber security game-based learning and training was conducted. The structure of identified approaches was analyzed and they were decomposed into their respective elements, which are related to cyber security and pedagogical aspects. The elements were grouped, and their relations were examined. Based on the conducted literature review, the concept map of cyber security game-based approaches key elements was formed.
3. Two analysis schemes were formed. The first one is based on the concept map of live competition technological and pedagogical characteristics and the categorization of live competitions' problems, whereas the second analysis scheme is based on the concept map of cyber security game-based approaches key elements. The former can be utilized for the evaluation of new live competition approaches, whereas the latter aims at evaluating new cyber security serious games during the design phase.
4. The COFELET ontology aims at providing an analytical description of the key elements of COFELET's compliant serious games along with the appropriate classes and their properties. These elements include the cyber security domain elements that model the actions attackers perform to unleash cyber security attacks (i.e., the tasks) and the strategies they employ to achieve their malicious objectives (e.g., CAPEC's attack patterns and the CKC model). The cyber security domain elements are associated with the learning and the instructional aspects (e.g., hints, utilized knowledge, exercised skills), which provide the means to infuse the didactics in the COFELET compliant approaches.
5. The COFELET game life cycle, a blueprint illustrating how the game's major components and the elements of the COFELET ontology are organized in the structure of a COFELET game and the course of phases for the development of COFELET compliant games. Besides, the COFELET game life cycle describes the main actors involved in the life-cycle of a COFELET game and the manner they have to cooperate.

1.4. Overall Research Approach

The methodology of the COFELET framework research study is based on the engineering method (Adrion, 1993) (Glass, 1995) and it is depicted in *Figure 1-1*. The applied methodology consists of four phases which are repeated three times in an iterative manner.

- **Iteration 1**: In *phase 1*, a literature review has taken place on the cyber security game-based approaches, the educational games' frameworks, and the problems and challenges of cyber security education. In *phase 2*, the game-based learning concept map and the capture the flag competitions' concept map were proposed that led to the elaboration of the analysis schemes of current approaches in live competitions field and in cyber security game-based learning

and training field (*phase 3*). In *phase 4* the elaborated evaluation schemes were utilized to assess current approaches in the field.

- **Iteration 2**: Based on the review process of phase 1 of iteration 1, the analytical evaluations of live competitions and cyber security game-based approaches, the COFELET framework was proposed in *phase 2*. Subsequently, a new literature review was carried out on the cyber security modeling techniques and methodologies (*phase 1* of iteration 2). The literature review of this iteration revealed the manner which cyber security attacks and models can be integrated into highly organized and parameterized learning environments (e.g., serious games). Next, the COFELET ontology was proposed in *phase 2* to formally exhibit a proposition for modeling the actions of attackers to unleash cyber-attacks and the strategies they employ to achieve their goals. Finally, the COFELET game life-cycle has been proposed, to illustrate the design aspects and the course of phases for the development of COFELET compliant games. In *phase 3* the design of the HackLearn game was produced followed by its preliminary evaluation in *phase 4*. HackLearn's preliminary evaluation employed the evaluation schemes elaborated in the first iteration.
- **Iteration 3**: Based on the design of the prototype cyber security COFELET compliant game and the COFELET game life-cycle, the HackLearn hacking simulation game has been implemented (*phase 2*). In parallel, a new literature review was conducted on the quality characteristics of serious games used in the evaluation of serious games (*phase 1*) that led to the analysis of the HackLearn's quality characteristics (phase 3). Finally, in *phase 4* the evaluation of HackLearn was conducted in an authentic educational environment in terms of the quality characteristics analyzed in phase 3.

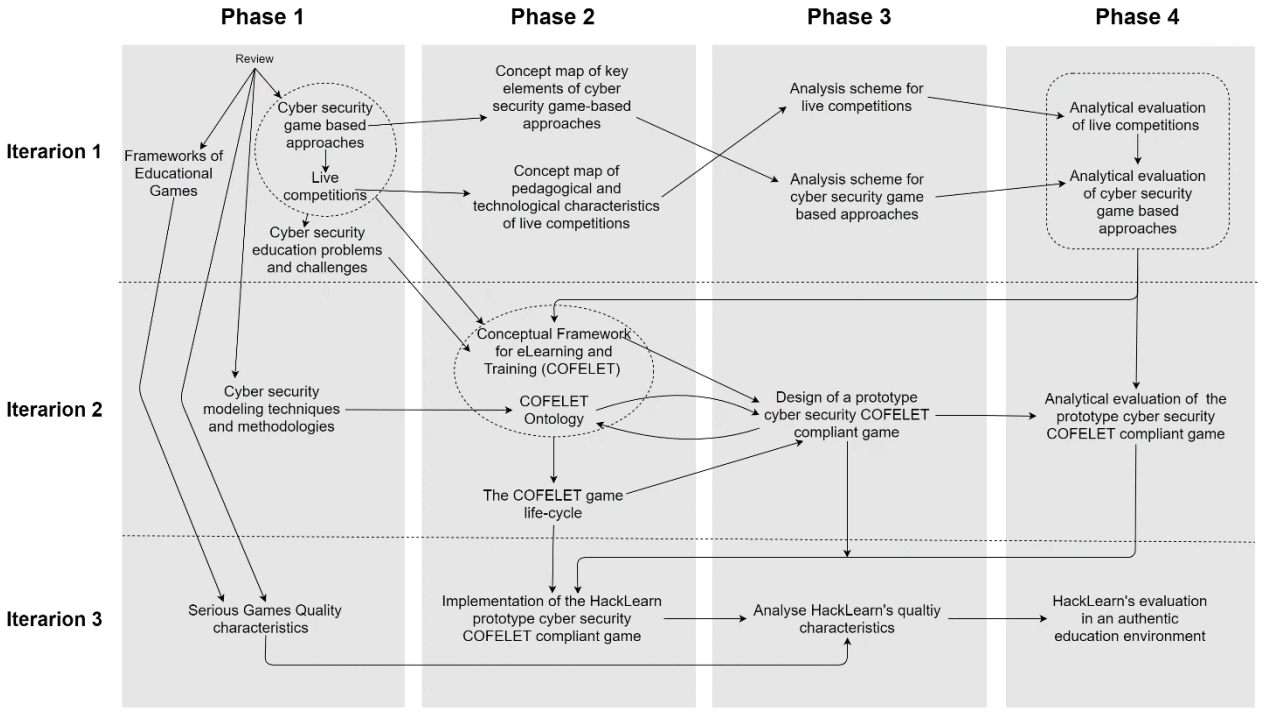


***Figure 1-1**. The methodology applied in the dissertation*

1.5. Thesis Structure

The thesis consists of eleven (11) chapters. The first chapter (the *Introduction*) introduces the thesis by presenting a brief overview of the problem, the research questions, the objectives and the contributions. In the second chapter, the *Theoretical Background*, the critical concepts important for comprehending the theoretical context of the thesis are described and clarified. The chapter also presents the existing frameworks for the development of serious games proposed in the literature along with the features that they indicate. Chapters 3 and 4 analyze the domains of cyber security game-based learning and training and live competitions respectively and provide some important considerations. Chapter 5 presents the COFELET framework, whereas chapters 6 and 7 present two important facades of COFELET, the COFELET ontology and the COFELET games life-cycle. Chapter 8 presents the HackLearn COFELET-compliant serious game and chapter 9 provides implementation details of HackLearn. Chapter 10 describes the HackLearn evaluations and presents the results, and chapter 11 concludes this thesis and provides considerations for future work.

2. THEORETICAL BACKGROUND

2.1. Introduction

The analysis of the research context, questions and objectives of this study leads to the framing of the presented research around three pillars. The first pillar regards game-based learning and serious games as well as the proposed serious game frameworks offering different approaches for game design and development; the second pillar contains the cyber security standards, methodologies and models used in nowadays in various fields of cyber security including the cyber security education; and the third pillar contains the cyber security education and the challenges it faces. This chapter focuses on the first two research pillars. Initially, the main concepts of the domains of serious games and game-based learning are analyzed and clarified and then the serious games design frameworks proposed in the literature are presented and discussed. Subsequently, the cyber security methodologies and models adopted in this study are presented (second pillar). The background related to the third research pillar is provided in chapters 3 and 4 along with the performed theoretical analysis (i.e., rational, reflections and concept maps).

2.2. Game-Based Learning, Serious Games and Relative concepts

2.2.1. Games

Playing games was always interesting, exciting, and stimulating. Games played on electronic devices that produce images presented on various displays are called video games. The video games industry has been exploded in the last decades. The games' market has been split into categories according to the gaming technology and the means used e.g., personal computers, consoles and mobiles. They are used for fun and entertainment purposes, but their potential applications are extended in various domains like education, military, health, safety, training, etc.

The growth of the video game industry has vividly increased the academic interest in the concept of games. Since 1980's games have been broadly studied in various contexts like anthropology, sociology, education and computer science. Consequently, researchers presented various definitions and characteristics of the game concept, trying to shape a formal statement that will help clearing the ambiguity of what truly a game is and accordingly how to use it (Schell, 2008). On this track, Salen and Zimmerman in their book '*Rules of Play: Game design fundamentals*' (Salen & Zimmerman, 2004) analyze and compare various definitions and finally come out with their own notable definition:

> "*A game is a system in which players engage in an artificial conflict, defined by rules, that results in a quantifiable outcome*." (Salen & Zimmerman, 2004; pp 80)

where:

- *System*: Is a set of parts that interrelate to form a complex whole and includes *Objects* (i.e., parts, elements, or variables within the system), *Attributes* of the system and the Objects, *Internal relationships* among the objects and the *Environment* that surrounds the system.
- *Players*: One or more participants that actively play the game. Players interact with the system of a game to experience the play of the game.
- *Artificial*: Games maintain a boundary from so-called 'real life' in both time and space. Although games occur within the real world, artificiality is one of their defining features.
- *Conflict*: All games embody a contest of powers. The contest can take many forms, from cooperation to competition, from solo conflict with a game system to multiplayer social conflict. Conflict is central to games.
- *Rules*: They are a crucial part of games. Rules provide the structure out of which play emerges, by delimiting what the player can and cannot do.
- *Quantifiable outcome*: Games have a quantifiable goal or outcome. At the conclusion of a game, a player has either won, lost or received some kind of numerical score. A quantifiable outcome is what usually distinguishes a game from less formal play activities.

Furthermore, Salen and Zimmerman claim that there is a peculiar relationship between game and play because games are a subset of play whereas play is a component of games. Jesse Schell in his definition (Schell, 2008) highlights the notion of a game as a problem-solving activity that has a particular resonance in the learning context. Moreover in (Kapp, 2012), Kapp proposed a modification of Salen and Zimmerman's definition to better fit it in the education domain:

> "*A game is a system in which players engage in an abstract challenge, defined by rules interactivity, and feedback, that returns in a quantifiable outcome often eliciting an emotional reaction*." (Kapp, 2012; pp 7)

Kapp's modification points out that a game player has to compete and correlate with the system or other players. Additionally, players will be informed on their progression and they will be subjected to the emotions caused by the actual play and the outcome they bring forward.

2.2.2. Game-Based Learning

Game-Based Learning (GBL) describes the learning process enhanced with the use of games. GBL considers the use of digital and non-digital games such as board games, and card games. On the contrary, the term Digital Game-Based Learning (DGBL) refers to learning through digital games. DGBL has been initially stated by Marc Prensky in his equally titled book (Prensky, 2005). Prensky

states that DGBL is the fusion of digital video games or game mechanics with learning, and he points out that games have a great potential for helping people to learn more effectively. Today's learners, who Prensky calls '*digital natives*', have changed dramatically. Digital natives have grown up using digital technology of which video games are a major part (Prensky, 2005). Moreover, today's learners use computers, gaming consoles, and mobile technologies to play, socialize and keep up to date. At the same time, teachers are not quite familiar with the use and comprehension of digital technology and especially the use of computer games. Prensky also claims that games provide a new means to deal with the problem of motivating the learners to stick with the learning process for the fulfillment of the learning objectives, as the learning materials are provided to the learners in the context of a story (i.e., the game's scenario and narrative) and the learning activities include problem-solving activities, analysis of the in-game components and the components' properties, behaviors and relationships (Tang et al., 2009). Besides, the GBL includes several beneficial characteristics (Tang et al., 2009) such as:

- It is engaging and motivating,
- It involves active participation from learners through a set of defined actions,
- It has clear learning objectives,
- It provides feedback to the learners,
- It assimilates the teaching and assessment aspects in the game-play,
- It is highly scalable as it can associate concurrently many participants.

2.2.3. Serious Games

The term Serious Games (SG) was first introduced in (Abt, 1970) by Clark Abt. SGs are games that have an explicit and carefully thought-out educational purpose and are not intended to be played primarily for amusement. According to Abt, this proposition does not claim that serious games are not, or should not be, entertaining (Abt, 1970). SGs and GBL approaches have a common ground, but they also differ in two fundamental ways:

a) SGs have a broader application scope that goes beyond the education domain,
b) GBL focuses on describing a game-based learning process and its characteristics (e.g., occurring activities, the learners' interactions with the games), whereas SGs, even when used in the educational domain, focus on the game tools developed and utilized for educational purposes. Moreover, in GBL the game mechanics infusion is not critical to the essence of the whole process, as the environment can be a non-game real-life setting (Kapp, 2012).

Zyda in (Zyda, 2005; pp 8) proposed the following definition for SGs and he also stated that serious games result from '*applying games and simulations technology to non-entertainment domains*':

> "*A mental contest, played with a computer in accordance with specific rules, that uses entertainment to further government or corporate training, education, health, public policy, and strategic communication objectives*."

A lot of debate and research has been carried out in the SGs domain, as there is no generally accepted definition of the term, and there are sparse opinions on the SGs' objectives, approaches, and methodologies (Martens & Müller, 2017). Susi et al., 2007 state that the term 'serious games' contains a contradiction as the notions of 'serious' and 'game' appear to be mutually exclusive. On the other hand, Zyda states that it is the addition of 'pedagogy' i.e., 'activities that educate or instruct' that make games serious. Some researchers argue that the term 'serious games' is not clearly defined and some even argue that game-based learning and serious games are more or less the same thing (Susi et al., 2007). To this end, many researchers tried to classify and clarify several concepts associated with SGs and GBL (e.g., edutainment, e-learning) presented in the subsequent section. In any case, it is commonly accepted that SGs are games that are used for other purposes than entertainment.

2.2.4. Relative concepts

There are several concepts related and partly overlapping with the notions of GBL and SGs such as edutainment, e-learning, and gamification. As these notions mean slightly different things and they often cause a form of confusion, many studies tried to classify, clarify them and schematically represent them (*Figure 2-1*, *Figure 2-2*, and *Figure 2-3*). Thus, *educational games* are games developed explicitly for educational purposes or games appropriated for educational use, even though they were not designated for this purpose. Tang et al. (2009) state that educational games are a subset (i.e., a special form) of SGs (*Figure 2-1*). *Edutainment* refers to education, which utilizes various media (e.g., video games, films, computer software, multimedia) to promote fun, entertainment, and learners' engagement. Edutainment usually targets the development of cognitive skills in young children mainly in a behaviorist manner (Tang et al., 2009).

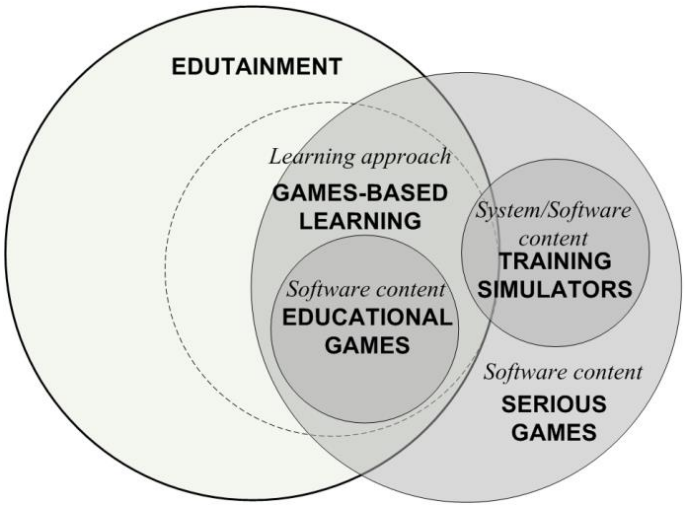


***Figure 2-1**. Relationships of SGs, GBL, and relative concepts according to (Tang et al., 2009)*

Since, edutainment aims at fusing entertainment and education or fun and learning it is associated with GBL and SGs, but it is not an identical notion to GBL and SGs and it is not a superset of SGs

or GBL (*Figure 2-2*). Most researchers and game designers claim that SGs differ from edutainment, because they employ different learning approaches, they have more diverse target groups (e.g., school children to adults), and they have purposes that go beyond traditional teaching and learning approaches usually utilized in *edutainment*.

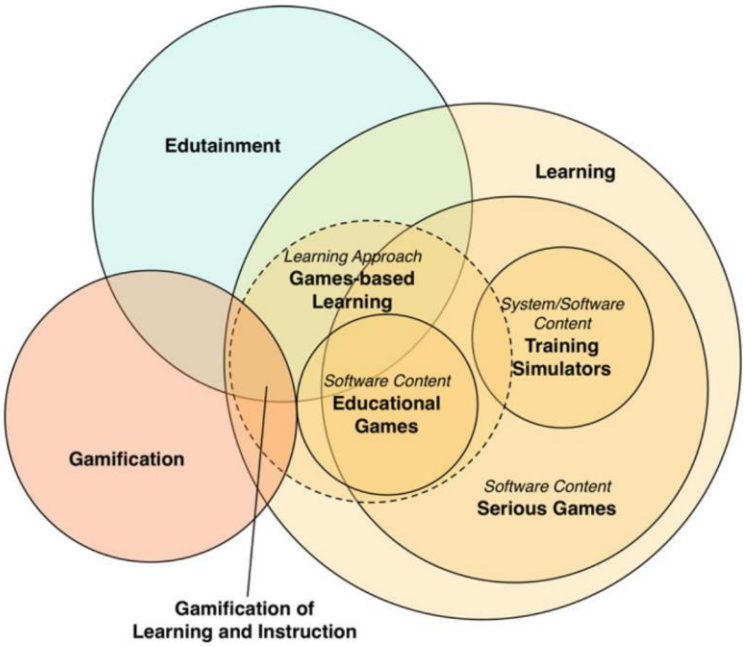


***Figure 2-2**. Relation between edutainment, gamification, and learning (Martens & Müller, 2017) after (Tang et al., 2009)*

Gamification is the use of game-based mechanics, aesthetics, and game thinking to solve problems and encourage learning in a non-game context. Mechanics include levels, badges, scores, and time constraints perceived through a well-designed user interface. Gamification and serious games share the same directions towards motivating people, trying to solve problems, and promoting the game-based thinking and techniques (Kapp, 2012). When gamification is applied in the learning domain it intersects with edutainment, with which they share the objective of motivating learners by making the learning more entertaining (*Figure 2-2*).

E-learning refers to any type of computer-based learning. E-learning is a very popular approach, as it is highly flexible in terms of the time and the space at which learning happens. E-learning can be synchronous or asynchronous and it can be distributed at any location. Unlike games and game-based approaches, e-learning does not imply any need for entertainment. All forms of learning, which use digital games are subsets of *e-learning*, as they electronically deliver materials to the learners (Zhang et al., 2004) (Breuer & Bente, 2010).

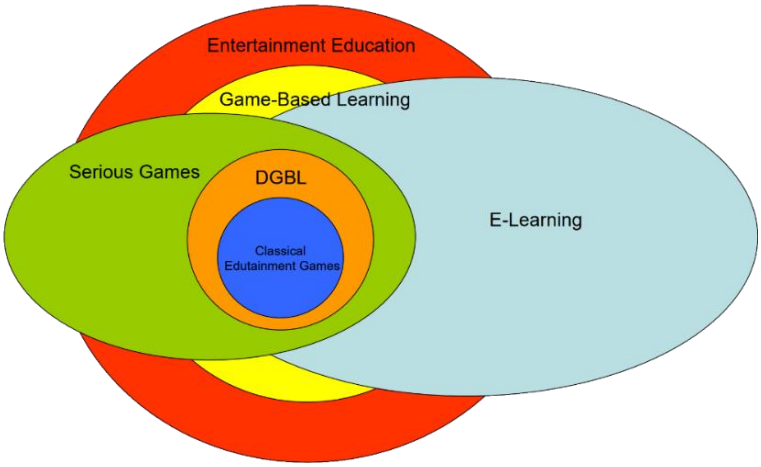


***Figure 2-3**. The relations between serious games and similar educational concepts (Breuer & Bente, 2010)*

Training Simulators are computer simulation programs intended for the development of trainee's skills (Tang et al., 2009). Computer simulation programs (or simulators) are software systems that represent real-world systems at a specified level of detail, and they model the behavior of these systems. Trainees interact with simulators by performing actions and by observing the feedback they get as output from the simulators. The output of a simulator corresponds to the output that the real-world system will generate when the user performs the same actions as the input (Parker & Becker, 2013). Training with simulators is not a substitute for physical training, but an intermediate between the theory and the real practical experience that is cheaper and safer.

As with SGs definition, there is a lot of debate on the relationship between simulators and games, as some researchers argue that all games are simulations, some that they are not (Taylor, 2014) and some that games are a special subclass of simulations (Parker & Becker, 2013). However, by definition, a simulation is an '*artificial representation of real conditions*' (Rothwell & Kazanas, 1997), whereas SGs do not have as primary objective to represent any real-word system or part of a real-word system, and so they are 'real-world' systems in their own right (Taylor, 2014).

2.2.5. Learning vs. Training

Training is considered more specific than learning and it is concerned with more practical topics that focus on particular jobs. Training is a planned and systematic learning activity, or a set of activities aimed at developing or modifying knowledge, skill, and abilities to perform a given task or job and realize its potential. On the other hand, learning can be formal or informal, but it is also related to the processes and activities that enable individuals to develop or assimilate knowledge, skills, and abilities. Learning has a continuous nature that causes permanent changes in learners' cognitive structures and determines the manners and the perspectives under which one thinks and behaves. Although, the same cognitive processes are likely involved in learning and training, the latter is utilized in more practical and job-oriented contexts aiming at achieving specific objectives. Training

is considered more task-oriented whereas learning, (especially formal learning) emphasizes on adapting towards the learners. Besides, training can be realized as a subset of formal learning (e.g., education) as the latter is a general concept that also foresees planned and systematic activities (Buckley & Caple, 2009).

Nevertheless, training can be part of any educational program. Consequently, Game-Based Training (GBT) has clearly stated performance objectives that aim to the development or modification and exercise of explicit knowledge and competencies. Normally GBT is expected to provide some sort of support including ordered guidelines' presentation or the presence of an instructor (Buckley & Caple, 2009).

2.2.6. Foundational Knowledge

Foundational knowledge is the set of critical information and skills professionals use frequently. It is recalled instantly and thus it is considered reflexive. Reflexive knowledge is considered very important in activities, professions, and environments that time is a critical factor. More specifically, it improves the time factor of a process, as it helps professionals achieve motion efficiency by removing the need to search for things. Subsequently, a cyber-security experts' training program should identify the foundational knowledge and verify that it becomes reflexive. Additionally, a cyber-security training program should consider the fact that quite often reflective knowledge decays, because it might not be practiced quite often for a number of reasons. For example, cyber security personnel of an organization very often work on tasks that are not related to the activities required to respond to a cyber security incident. Therefore, a program needs to help trainees practice foundational knowledge by providing 'continuous' and 'always on' training and motivating them to carry on. The knowledge, skills, and abilities (KSA) that form the foundational knowledge of cyber-security have not been defined yet. However, some typical information and procedures like operating system commands, protocol to port mappings, standard practices in specific cyber security areas (e.g., forensics) can be considered part of the field's foundational knowledge (Allen & Straub, 2015).

2.2.7. Importance of Games in Learning and Training

Learning and teaching have been moving to constructionism and sociocultural theories. Moreover, the technology that supports learning is constantly changing and new approaches are created. On the other side, learners themselves have changed. New generations acquire the appropriate skills to use digital technology to communicate, be informed and learn new things, whereas gaming has become part of their culture (Breuer & Bente, 2010).

All these revolutions constitute critical the need for alternation in learning. The integration of games in learning provides the means to embrace all these changes in learning and training. Besides, the inclusion of games in the educational process has been linked to the increase in students' motivation, which aids students to increase their efforts and performance by soliciting new challenges and trying

new experiences (Garris et al., 2002). Games use visual, textual, and auditory channels to provide feedback and progression indicators to the users (Greitzer et al., 2007). Moreover, games have various characteristics that correspond to valuable features of learning. Wendy Bedwell (Bedwell et al., 2012) classifies nineteen game attributes that have been defined in the bibliography and can be linked to learning. The most important are the attributes of *assessment*, *adaptation*, *challenge*, *control*, *interaction*, and *rules*/*goals*. The *control* feature refers to the player's capacity for power or influence over elements of the game. The *interaction* with equipment in the user interface indicates the game's response to the controls and actions of the player. Yet, *interaction* refers to a face-to-face relationship between players of the game for communication, cooperation, and challenge. Likewise, *interaction* considers the participation of the player in the game's communities that produce a sense of belonging. *Adaptation* adjusts the game's features, e.g., level of difficulty, to the skills and abilities of the player by matching challenges and possible solutions whereas assessment measures the achievements within the game, e.g., scoring. *Challenge* is the ideal amount of difficulty and improbability with the aim of obtaining the specified goals. A challenging game creates barriers between the current and goal state by adding progressive difficulty and informational ambiguity. Tutorials, sometimes under the form of an intelligent tutor, teach players how to follow certain rules to achieve the goals of the game and provide feedback through score notification and adjustment of the system (Bedwell et al., 2012).

The features stated above can provide the means of creating games that endorse effective notions like situative learning theories focusing on social learning and social interactions (De Freitas & Liarokapis, 2011). According to Piaget's theory, the learner is not simply a passive receiver of information. The number of passive activities, like reading, hearing, and watching, is reduced. Instead, learners control the game, interact with it, and are encouraged to work immediately on meaningful and realistic tasks to solve problems. As Bruner suggests, the learner uses prior knowledge to advantage, experiments with the system, make assumptions and errors that are not traumatic but pedagogically productive. The game assesses data that is coming from the user and informs the '*play*' within the learning environment in order to adapt according to his/her actions. Consequently, learners consider, organize, and use the new information in ways that encourage active construction of meaning, help build lasting memories, and deepen understanding of the material (Greitzer et al., 2007). The sense of control and freedom that is provided to the user is pleasurable and motivates further interaction (Breuer & Bente, 2010). The challenge increases the game's competition, adds excitement, and motivates the player even more. Additionally, a player does not manufacture his/her knowledge in a cultural and communicative '*gap*' (Vygotsky, 1978). In a serious game, learners have the means to interact, cooperate and compete with other learners and thus, they can be a member of a community which involves cultural and communicative aspects.

2.3. Design Frameworks

The success of entertainment games in the last decades among the new generations has prompted the interest of the community to utilize them in other fields such as the education field. As the serious game domain is complex and multidisciplinary, a lot of research has been conducted in several fields (e.g., development, evaluation) and under different perspectives (Zimmerman, 2004), (De Castell & Jenson, 2003). Several SGs frameworks have been proposed as guides for the analysis and design of serious games, involving different mechanics and elements, and relying on different educational approaches. However, the SGs field is still considered an emerging field that faces many challenges both from the game design perspective and more frequently for applying effective games-based learning. The SGs development field lacks common methodologies and design standards in particular areas (e.g., cyber security education). In the remainder of this chapter the most prominent frameworks proposed in the literature are described, which assisted in the elaboration of the presented study.

2.3.1. The Mechanics, Dynamics, and Aesthetics Framework

The Mechanics, Dynamics, and Aesthetics (MDA) framework (Hunicke et al., 2004) divides the relation of the designer with the player into three distinctive counterparts (or stages) presented within its name (*Figure 2-4*). The *Mechanics* counterpart or rules of the game is elaborated by the game designer and it describes the game's components and the manner they are represented in the game. The *Dynamics* counterpart is instantiated at run-time and it describes the system of the game that is formed by the *Mechanics* influenced by the player's in-game actions or inputs. The *Aesthetics* counterpart focuses on the analysis of the player's emotional responses when playing the game.

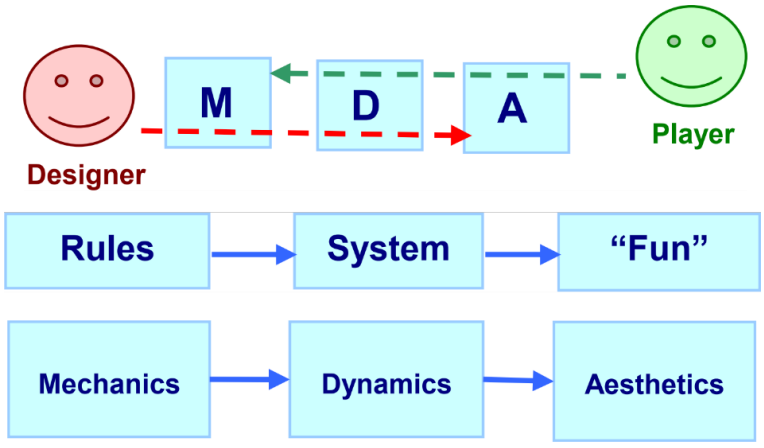


***Figure 2-4**. The MDA Framework (Winn, 2008)*

The MDA framework is a well-known approach with over 2500 citations. However, it has never been utilized as a blueprint for the development of serious games, but only for the elaboration of SG design frameworks.

2.3.2. The Design, Play, Experience Framework

The Design, Play, and Experience (DPE) framework (Winn, 2008) (*Figure 2-5*) relies on the MDA framework, and similarly, it divides the relation of the designer with the player into three counterparts or stages presented within its name. In *Design*, the designer designs the game, in *Play* the player plays the game, and in *Experience* the results of the player's experience are captured. Moreover, the DPE framework decomposes the design stages into layers, i.e., game's subcomponents of serious games design. The DPE layers include the *Learning*, *Storytelling*, *Gameplay*, and *User Experience* presented in the remainder of this section.

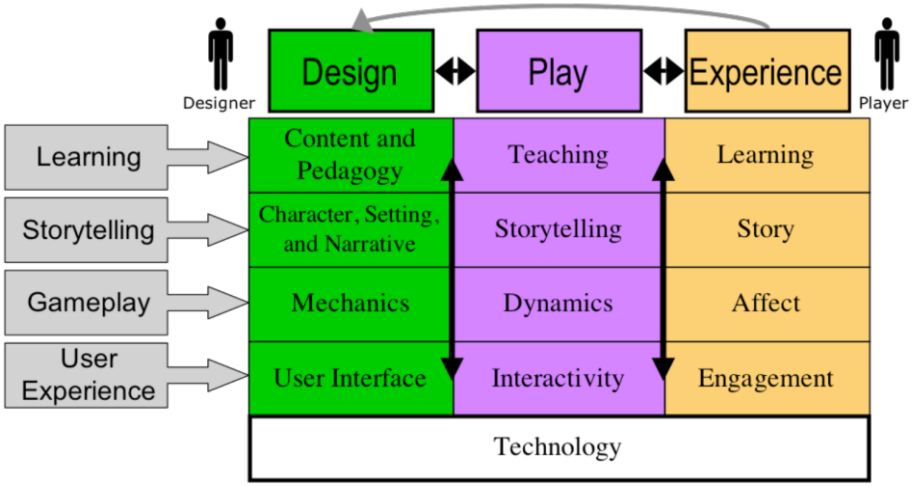


***Figure 2-5**. The DPE Framework (Winn, 2008)*

In the *Learning* layer, initially in the *Design* stage, the game designer sets the learning outcomes, the teaching content, and the learning strategy that will be applied. During the gameplay (i.e., the *Play* stage), the actual teaching takes place, whereas the *Experience* stage represents the learning that is the fulfilment of the learning objectives.

In the *Storytelling* layer, the game designer forms the games' environment or the context, the game characters, and the game's narrative. In the *Play* stage, the player interacts with the game's environment and characters, follows the game's narrative and s/he generates her/his own story (i.e., the storytelling). The players' generated storytelling can be thought in analogy to the part that an actor performs by following the script of a play that is an analogy of the game's narrative. The resulting experience of the player's story forms the story of the *Experience* stage.

The *Gameplay* layer defines the player's action in the game. In the *Mechanics* stage, the game designer defines the actions that can take place in the game's context, the game's challenges and goals. At runtime, the player performs actions and interacts with the game. The results of the player's actions are captured in the *Dynamics* stage, whereas the experiences that the player has at runtime are captured in the *Affect* stage.

The *User Experience* layer represents the surface of the game (i.e., the most visible part), in which the game designer designs the game interface. The game interface has to be entertaining and accessible by the player to provide him/her the means to interact with the game (*Interactivity* stage) and engage (*Engagement* stage).

2.3.3. Four-Dimensional Framework

The Four-Dimensional Framework (FDF) (De Freitas & Oliver, 2006) was designed to explicitly consider the use of serious games and simulations in education. The FDF framework proposes four dimensions depicted in *Figure 2-6*. The FDF dimensions form an iterative process the educator has to undertake before implementing and utilizing a serious game. The FDF aims at helping educators evaluate the feasibility of using educational games in their practice, appreciate the manner a serious game can be integrated into the curriculum and the learning process. The FDF also assists researchers and game designers in analyzing educational games.

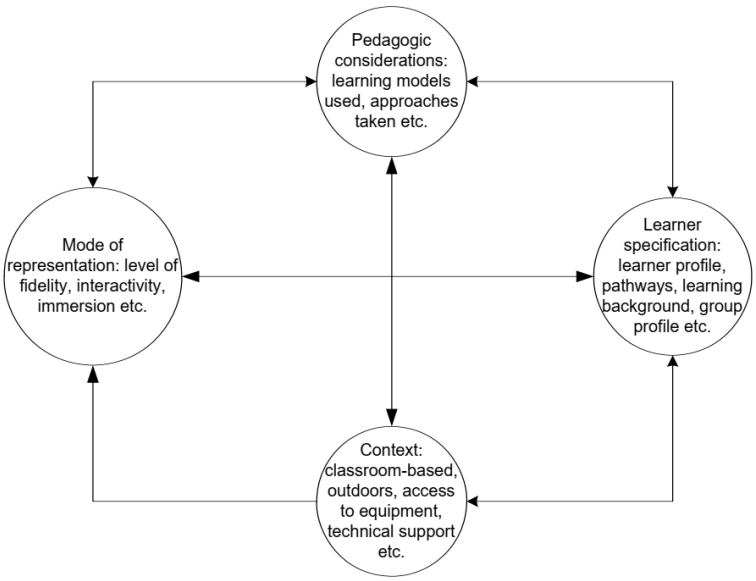


***Figure 2-6**. The Four-Dimensional Framework (De Freitas & M. Oliver, 2006)*

The *Context* dimension of the FDF framework focuses on the environment in which the gameplay and learning session takes place. This dimension examines both the macro-level factors (e.g., historical, political, and economic factors) and the micro-level factors (e.g., availability of specific resources and tools, educator's background, and technical support). The game context is a determining factor that influences the game scenario and the manner the educator and the instructor act in the game.

The *Learner specification* dimension focuses on the learner's attributes or the group of learners' attributes. The Learner specification dimension includes the process of analyzing and modeling the learner's attributes and needs and forming the learner's profile. The formation of the learner's profiles facilitates the design of proper learning activities that will ensure the fulfillment of the

required learning outcomes. The learner's attributes considered include the characteristics of age, background, learning style and preferences. The Learner specification dimension is a determining factor in the successful development of game-based learning approaches. For example, a learner's experience in digital games has to be taken into account in this dimension, as it can influence the manner the learner interacts with the game and thus, s/he is involved in the game's activities.

The *Mode of representation* dimension is concerned with the game's representations (e.g., the game's world and the game's narration), and it involves the aspects of the mode of presentation, the interactivity, the levels of immersion and the fidelity. For example, the game's world needs to comprise attractive scenes and be consistent with the game's purpose and genre to create effective and immersive environments.

The *Pedagogic considerations* dimension focuses on the formation of the learning process, both in formal and in informal learning. The Pedagogic considerations dimension analyzes the pedagogical side of the learning approach and it considers the methods, the theories, the models and the frameworks that will support the learning practice. The pedagogic considerations are a very critical factor because it determines the manner that the game will be assimilated to the learning process.

The FDF framework also includes a checklist on specific issues of each dimension of the framework which is used to evaluate the adoption and utilization of educational games in the learning process.

2.3.4. The Learning Mechanic - Game Mechanic

The Learning Mechanic - Game Mechanic (LM-GM) model is a framework for the analysis, evaluation and design of serious games developed by Arnab et al., 2015. The LM-GM model foresees the identification of the learning mechanics (LMs) and the game mechanics (GMs) and the mapping of the GM components of a serious game to its LM components. The LMs refer to the integrated components of a serious game that allow the learner to achieve his/her cognitive goals. The GMs describe the manner learners interact in the game, the game rules, the learner's goals, and the performable in-game actions. Along with the LM-GM, Arnab et al. proposed a non-exhaustive set of LMs and GMs elements (listed in *Figure 2-7*).

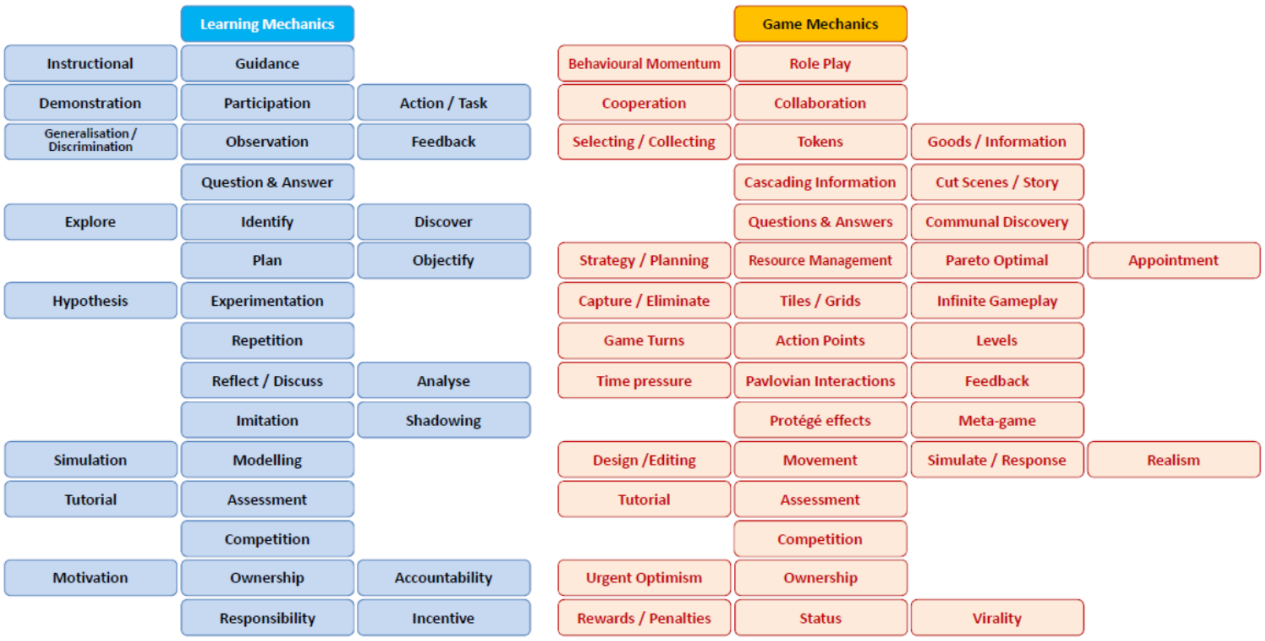


Figure 2-7*. Learning and game mechanics used as the basis to construct the LM-GM map for a game (Arnab et al., 2015)*

Although, the LM-GM model considers a variety of learning theories (e.g., constructivism, behaviorism, personalism), it focuses on supporting the intrinsic experiential learning of games involving knowledge acquisition and skill training through the game mechanics (e.g., missions, leaderboards, goals, levels, badges, role-play, etc.). The LM-GM model most important features are:

- The design of a serious game is facilitated through the combination of LMs and GMs to describe what the learner experiences, through which the learning takes place.
- The game designers are allowed to correlate LMs and GMs in any manner it is convenient for the game development process.
- The classification of LMs and GMs components based on Bloom's ordered thinking skills (*Figure 2-8*).

GAME MECHANICS	THINKING SKILLS	LEARNING MECHANICS	
○ Design/Editing ○ Infinite Game play ○ Ownership ○ Protégé Effect ○ Status ○ Strategy/Planning ○ Tiles/Grids	CREATING	○ Accountability ○ Ownership ○ Planning ○ Responsibility	LOTS to HOTS
○ Action Points ○ Assessment ○ Collaboration ○ Communal Discovery ○ Resource Management ○ Game Turns ○ Pareto Optimal ○ Rewards/Penalties ○ Urgent Optimism	EVALUATING	○ Assessment ○ Collaboration ○ Hypothesis ○ Incentive ○ Motivation ○ Reflect/Discuss	
○ Feedback ○ Meta-game ○ Realism	ANALYSING	○ Analyse ○ Experimentation ○ Feedback ○ Identify ○ Observation ○ Shadowing	
○ Capture/Elimination ○ Competition ○ Cooperation ○ Movement ○ Progression ○ Selecting/Collecting ○ Simulate/Response ○ Time Pressure	APPLYING	○ Action/Task ○ Competition ○ Cooperation ○ Demonstration ○ Imitation ○ Simulation	
○ Appointment ○ Cascading Information ○ Questions And Answers ○ Role-play ○ Tutorial	UNDERSTANDING	○ Objectify ○ Participation ○ Question And Answers ○ Tutorial	
○ Cut scenes/Story ○ Tokens ○ Virality ○ Behavioural Momentum ○ Pavlovian Interactions ○ Goods/Information	RETENTION	○ Discover ○ Explore ○ Generalisation ○ Guidance ○ Instruction ○ Repetition	

***Figure 2-8**. The LM-GM model: Classifications based on Bloom's ORDERED Thinking Skills (Arnab et al., 2015)*

2.3.5. Activity Theory Model for Serious Games

Activity Theory Model for Serious Games (ATMSG) (Carvalho et al., 2015) is an extension of the Learning Mechanic - Game Mechanic (LM-GM) model (Arnab et al., 2015). The ATMSG model (*Figure 2-9*) is based on the activity theory, a social constructivism learning theory, considering the educational games as interactive, composite, and dynamic systems analyzed under gaming, learning and instructional perspectives (Jonassen & Rohrer-Murphy, 1999).

Activity, as the central unit of analysis in the activity theory, involves subjects that interact with objects directed at motives. In general, activities can be identified by answering the question 'who is doing what for what purpose and how". To identify activities in a game one should answer questions like: "who is the learner?", "why is the learner engaging with the game?", "why is the game produced?", "how is the game used to teach?". An activity is decomposed into a series of actions. Actions are directed at a goal and they can be further decomposed to lower-level units, called operations (e.g., reading text, changing focus, drawing cards, clicking buttons). Operations are performed according to given conditions. *Figure 2-9* depicts the three main activities and the relationships between people and artifacts in this system.

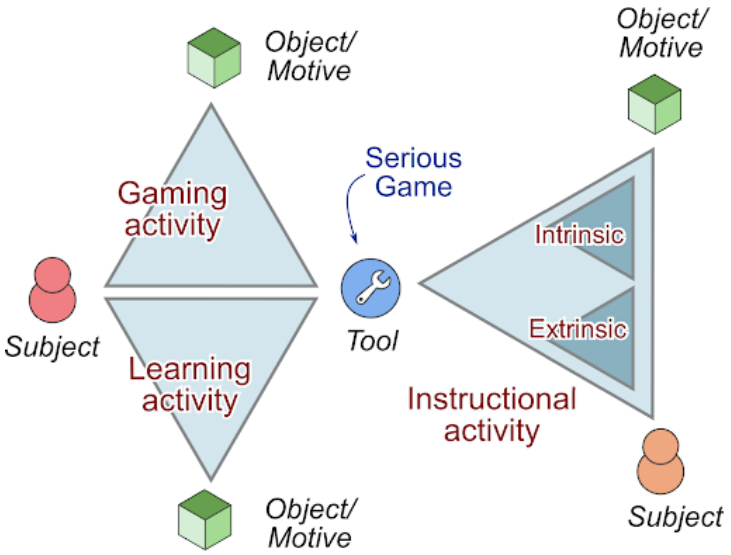


***Figure 2-9**. The Activity Theory Model for Serious Games*

The ATMSG model highly supports the design and development of educational games, as it facilitates their analysis, organization and detailed representation, by offering a precise approach for the realization of the games' components. While other models provide a high-level understanding of serious games aspects, the ATMSG model facilitates the systematic analysis and illustration of the gaming and learning elements by proposing a taxonomy of components and an approach for the creation of tables (*Table 8-1. HackLearn's Serious Game Components*) and diagrams (*Figure 8-1. HackLearn's Sequence Diagram*).

Besides, the model follows activity theory principles by taking into account the serious games' main subjects (i.e., player/learner and instructors) and their motives that drive the activities performed in the game. The instructor's motives initiate the intrinsic and extrinsic instructional activities whereas the learner's and the player's motives initiate the learning and gaming activities. The ATMSG model proposes a taxonomy for the identification of the serious games' primary elements (e.g., characters, hints, buttons) and their classification as gaming, learning and instructional components according to the activity they embrace. The ATMSG model also follows a hierarchical approach to decompose the activities taking place in a serious game into a sequence of actions. Actions are mediated by game components, called tools and they are identified and classified as gaming, learning and instructional. For example, the gaming actions describe the player's perspective as a gamer when s/he has to solve a puzzle and perform a pattern matching, whereas the learning actions consider actions a player performs as a learner, such as recalling information and using skills to apply tasks. On the contrary, instructional actions describe the instructor's perspective aiming at providing help, guidance, and feedback to support learners achieve the learning objectives of the game and reflect on their accomplishments.

2.3.6. Adaptability Model

Adaptability in educational systems such as serious games, learning management systems, and simulations have drawn much attention in the past years, as it improves the effectiveness of the learning process. Adaptability makes a serious game more interesting and motivating, as it adjusts challenging to an optimal point and it maintains the flow state in learning (i.e., the mental state according to which the learner is fully immersed, focused, and involved in performing activities in a joyful manner) (Csikszentmihalyi, 1997) (Vandewaetere et al., 2013). Adaptability is consistent with Vygotsky's theory of the zone of proximal development (Vygotsky, 1978), according to which learning is optimized when learners are challenged with learning tasks at the edge of their competencies and knowledge. Besides, adaptability improves the replay ability of a game as it makes it unpredictable by dynamically changing the gameplay (Lopes & Bidarra, 2011). Adaptability can take various forms in the context of a game according to the game's objectives.

Researchers in (Vandewaetere et al., 2013) identified that most educational games apply a low degree of adaptability mainly relying on stereotypical models. To this end, researchers studied the gameplay and player characteristics that leverage adaptability in educational games, and they proposed a theoretical player-centered adaptability framework (*Figure 2-10*). Their approach combines player characteristics and gameplay, including the learning preferences (e.g., background knowledge, interests), the gaming skills, the goals set by the players and the players' motivation.

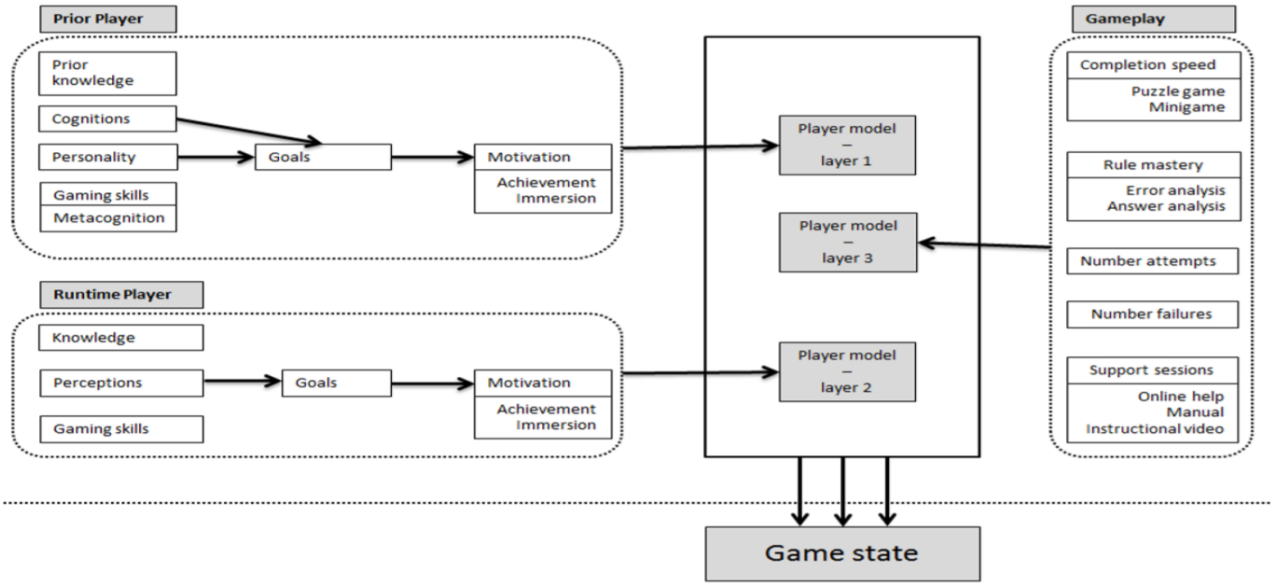


***Figure 2-10**. A multi-layered player-centered adaptivity model (Vandewaetere et al., 2013)*

The players' characteristics are categorized as runtime or prior to runtime characteristics. Runtime characteristics change during gameplay and sometimes they are affected by the gameplay itself. For example, at runtime, a player is low-motivated when starting a game, but s/he might be identified as high-motivated after a period of time in a game session. On the contrary, motivation prior to runtime relates to the notions of advancement (e.g., devotion to gather game's prizes, aim to progress rapidly

in the gameplay) and the immersion facet focusing on discovery (e.g., desire to discover things), role-play and customization (e.g., the player is pleased when changes character's appearance) (Lopes & Bidarra, 2011). Learners focusing on performing well in games by gathering points and prizes tend to apply simple strategies during the game play and usually do not put much effort in confronting challenges and in-game difficulties. On the other hand, learners devoted to pursuing the mastery of knowledge and competencies tend to plot and apply sophisticated and methodological strategies, they prefer to learn through challenging activities, and they insist on overcoming difficulties when they arise in the games (Poortvliet & Darnon, 2010). Player characteristics are associated with the gameplay characteristics focusing on the gameplay and the learning process. The gameplay characteristics include the reaction times, the number of attempts learners make to successfully perform an activity and the need for help. The main challenge in the adaptability of serious games is how to implement the adaptability characteristics into the game and how the responses of the game will be modeled without disrupting the fun façade while learners are in the flow state.

2.3.7. Assessment in Serious Games

Serious games provide a kind of assessment, as they share principles and characteristics to evaluate the possession of knowledge and competencies. Serious games and assessment describe and represent knowledge skills and abilities in a quantifiable manner, they both have rules, and they involve artificial tasks that have to be completed (i.e., conflicts) (Behrens et al., 2007). Though, in serious games the assessment takes place authentically and efficiently. The learner's behavior is observed by the game engine, his/her actions are evaluated, and they are always followed by the providence of immediate feedback (e.g., providence of level-ups, assignment of points, the play of animations and sounds etc.). In this way, the learners either move to the next level, when their knowledge and competencies are assessed as possessed, or not, otherwise. Besides, the traditional assessment is difficult to include appropriate models and appearance of what it is evaluated because "*it is hard to write a multiple-choice question about changing a car tire to actually feel like changing a car tire*" (Behrens et al., 2007). On the contrary, in serious games the use of knowledge representations, simulations and modeling techniques can create virtual environments in which learners can interact and perform their actions and be assessed in a semantically meaningful manner. For example, in a computer network assessment taking place in a simulation game, the learners interact with in-game representations of devices and networks or they connect in real-time to a real network. Several assessment design methodologies have been proposed aiming at identifying and assessing the knowledge and competencies provided in educational environments such as serious games and simulations such as the Educational Model (Joosten-ten Brinke et al., 2007), the e-Framework Reference Model for Assessment (FREMA) (Wills et al., 2009) and the Evidence-centered Design (ECD) framework (Almond et al., 2002) (Mislevy et al., 2003).

2.3.7.1. Evidence-centered Design Framework

The ECD framework is the most popular conceptual design framework for designing performance-based assessments. The ECD framework reconciles the actions (or tasks) learners perform in games with what they learn. To do so, it defines and unifies a set of conceptual and computational models:

1. The *Competency Model* describes in an explicit and structured manner the knowledge, skills and abilities that will be quantitatively evaluated at each point in the game. The *Competency Model* is associated with the *Student Model*, as it depicts the knowledge and competencies the SG's target group must acquire and the level of possession they must achieve. Besides, a *Student Model* is derived from the *Competency Model* for each learner and during the gameplay, which is updated when the learner exercises knowledge, skills and abilities. The *Student Model* represents the system's view on the learner's KSAs.
2. The *Task Model* provides formal specifications of tasks. The tasks interpret what learners do in the game to provide evidence that they exercised the target KSAs. Tasks are considered instantiations of the specifications described in the Task Model. Additionally, the Task Model describes the conditions and the contexts in which the learners can interact and perform tasks.
3. The *Evidence Model* is a link between the Competency Model and the Task Model. The Evidence Model describes the learner's observable behaviors (i.e., the evidence) elicited in the game as a result of the completion of tasks described in the Task Model. The Evidence Model also describes the associations of the observable behaviors or evidence with the KSAs specified in the Competency Model including the actions the learner is expected to do, scoring schemes and rubrics.
4. The *Assembly Model* orchestrates the other models of the ECD framework, and it determines the structure and the operation of the assessment aspect of the SGs. More specifically, the Assembly Model includes details for the organization of the game's elements, the manner the game's elements appear, the sequence of tasks etc.

The ECD framework divides the process of developing the assessment into five activities (i.e., the *layers* of the framework) (Riconscente, 2015):

1) The *Domain Analysis*: involves the investigation of the domain of interest and an identification of the critical characteristics.
2) The *Domain Modeling*: based on the details of the Domain Analysis layer the assessment arguments are described in narrative form according to Toulmin's general structure for assessment arguments (Toulmin, 1958) (*Figure 2-11*).
3) The *Conceptual Assessment Framework*: the assessment arguments described in the Domain Modelling layer are expressed as blueprints by utilizing design patterns and with respect the elements and the processes that will be embodied in the assessment.
4) The *Assessment Implementation*: the implementation of assessment includes creation of the structures depicted in the *Conceptual Assessment Framework* layer. This layer includes the

activities of creation of scoring algorithms, programming of simulations, authoring of tasks and items, creation of rubrics.

5) The *Assessment Delivery*: describes the assessment taking place including the learner's interactions with the environment, the performance of tasks, the evaluation of the learners' achievements and the production of reports and feedback.

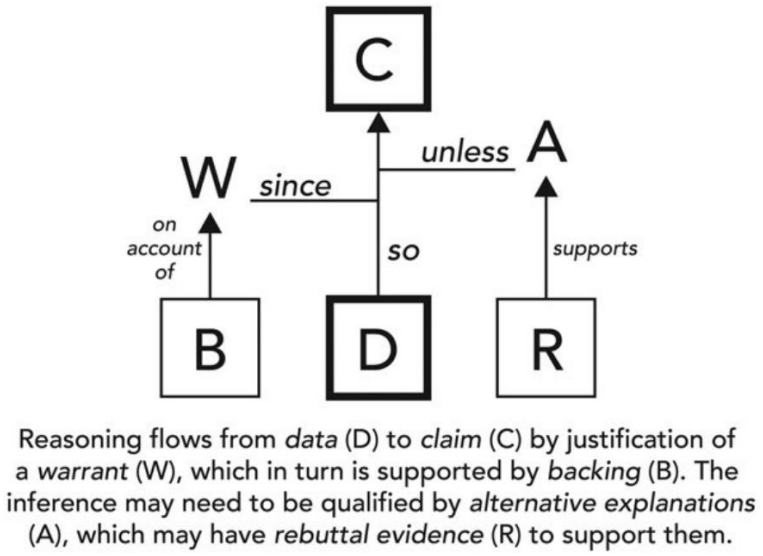


***Figure 2-11**. Toulmin's structure for arguments adopted by (Riconscente, 2015)*

The creators of the ECD framework have also proposed the Four-Processes architecture (Almond et al., 2002) which can be applied in various contexts (e.g., computer-based testing procedures, tutoring systems, etc.).

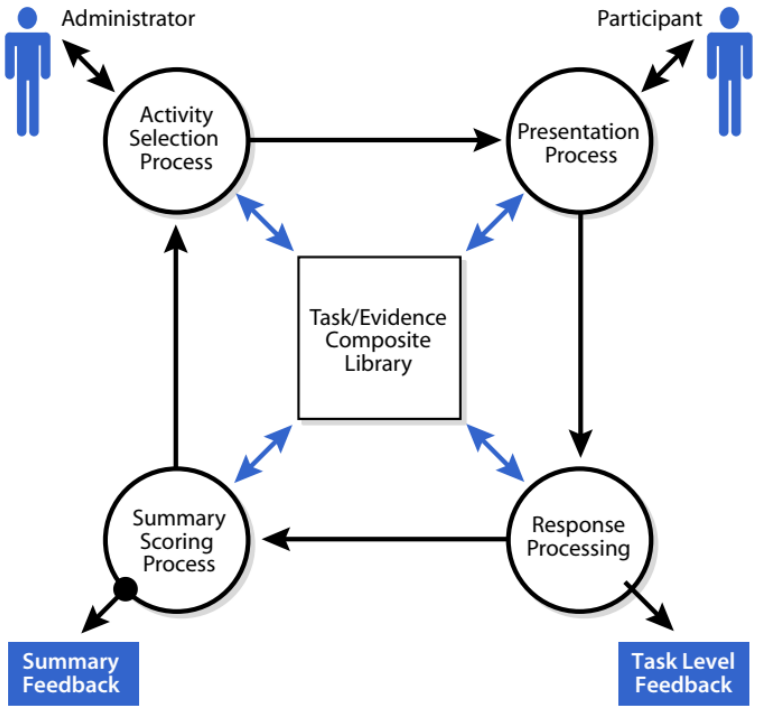


***Figure 2-12**. The Four Processes*

The Four-Processes architecture divides the activities of the Assessment Delivery layer into the processes of *Activity Selection*, *Presentation*, *Response Processing* and *Summary Scoring*. The four processes interact with the Task/Evidence Composite Library, a repository of tasks that holds the appropriate details of the tasks (i.e., details for the tasks' selection, presentation, and scoring), required by the processes of the Four-Processes architecture.

1. The *Activity Selection*: the subsequent activity is selected from the Task/Evidence Composite Library. The new activity can be associated with any kind of objective (e.g., instructional, evaluation, assessment administration).
2. The *Presentation*: the subsequent activity is presented to the learner along with the related materials such as images and audio. Materials are stored in the Task/Evidence Composite Library, although materials can be brought in the presentation by external resources.
3. The *Response Processing*: identifies and captures the evidence produced by the learner's actions (i.e., the tokens of the *Evidence Model* related to the competencies of the *Competency Model*). The captured evidence indicates the learner's status on the knowledge and competencies under evaluation. The captured evidence is passed to the next process.
4. The *Summary Scoring*: summarizes evidence passed by the *Response Processing,* and based on this evidence it produces reports and updates the *Student Model*.

2.3.7.2. The Quality Characteristics Evaluation Framework for Serious Games

The researchers of (Abdellatif et al., 2018) performed a literature review on the quality characteristics used in the evaluation of serious games and they discussed their dependencies and associations. As a result, they also proposed the quality characteristics framework for evaluating the use of serious games (QC framework). The framework divides the quality characteristics found in the literature into primary and secondary characteristics. The primary characteristics include the characteristics of learning outcomes, user experience, user satisfaction, engagement, motivation, understandability, and usability. They affect mainly the quality of serious games, whereas their absence downgrades the effectiveness of serious games. On the contrary, the secondary characteristics provide minor impacts to the quality of serious games and they are not crucial for their success in delivering educational content. The secondary characteristics include the game design, the effectiveness, the user interface, the acceptance and the usefulness.

The primary quality characteristics are associated with each other and with the secondary ones. For example, according to the literature review (Abdellatif et al., 2018), the user experience is associated with the engagement, whereas the engagement is based on the motivation characteristic (Dele-Ajayi, 2016) and it is associated with the acceptance characteristic, as the learners will not engage with a game they do not accept.

2.3.7.3. The Didactic Framework

The didactic framework proposes four stages for the process flow of simulation games in the business field (Utesch, 2016), whereas in (Nilüfer et al., 2018) three stages of the framework have been associated with corresponding phases of assessment (*Figure 2-13*).

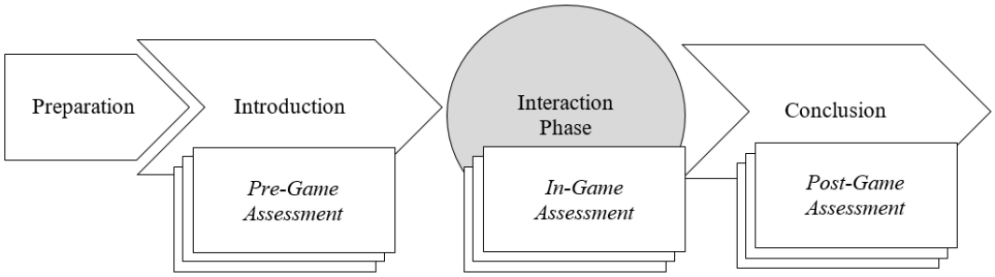


***Figure 2-13**. The Didactic Framework associated with three assessment phases*

In the Preparation stage, the appropriate organizational conditions are managed, and the participants are informed about the aims and objectives of the course. In the Introduction stage, the participants are familiarized with their roles and the problems they will have to solve in the game. In the Interaction phase stage (herein will be stated as the *Interactions* stage), the participants interact with the simulation game (i.e., perform game sessions) and face the problems they have to solve. The Interactions phase stage consists of five sub-stages in which participants analyze the problem, develop a business strategy, implement a business strategy, run the simulation and present the results. In the Conclusion stage, the participants reflect on their decisions and applied strategies and their work is summarized.

On the other hand, in the pre-game assessment usually, the knowledge and capabilities of the learners are measured. Moreover, depending on the context of the educational approach, several data can also be collected such as demographic information (e.g., gender, age), participants' learning styles, and attitudes (Smith et al., 2015). The in-game assessment involves the collection of qualitative data which denotes the participant's performance (e.g., sequences of actions, percentage of goals accomplished, goal completion times). The data of the in-game assessment phase is collected though the games' logging mechanisms or through questionnaires and interviews. Finally, in the post-game assessment of the Conclusion stage the knowledge and capabilities of the participants are measured through questionnaires, discussions, interviews, or performance evaluation by observers (Nilüfer et al., 2018), (Smith et al., 2015).

2.4. Cyber Security Standards

One of the main challenges of the presented study was to utilize well-known cyber security methodologies and models which will verify the validity, applicability, and sustainability of the presented work. Several models have been considered and studied, though the Mitre's Common

Attack Pattern Enumeration and Classification (CAPEC) (CAPEC, 2021a) and Lockheed Martin's Cyber Kill Chain (CKC) (Martin, 2014) were embraced. CAPEC and CKC are widely used in the fields of threat intelligence and modeling of the cyber security domain to describe the adversaries' actions, techniques, and strategies. The CAPEC was considered more appropriate than related approaches (e.g., MITRE's Adversarial Tactics, Techniques & Common Knowledge (ATT&CK)) as it is a comprehensive attack knowledge repository containing a large collection of attack patterns more appropriate for education and training and application threat modeling (CAPEC, 2021b). The CKC model was considered more appropriate than related approaches (e.g., ATT&CK, Diamond model (Caltagirone et al., 2013)), as it is a universally accepted and adopted model focusing on the high-level objectives of the adversaries. Additionally, the National Cybersecurity Workforce Framework (NCWF) (Newhouse et al., 2017) was adopted in the presented study to form a basis for the definition of games' LOs and learners' roles (e.g., forensics analyst, vulnerability assessment analyst). The remainder of this section presents the CAPEC, CKC, and NCWF.

2.4.1. CAPEC

Common Attack Pattern Enumeration and Classification (CAPEC) (CAPEC, 2021) is a publicly available dictionary and classification taxonomy of attack patterns (APs) maintained by the MITRE Corporation and sponsored by the US Department of Homeland Security. CAPEC specifies a schema that defines APs and describes their attack methods. CAPEC includes more than 550 APs organized hierarchically according to the mechanisms employed when exploiting a vulnerability and the domain of attack. The mechanisms of attacks are the '*Engage in Deceptive Interactions*', '*Abuse Existing Functionality*', '*Manipulate Data Structures*', '*Manipulate System Resources*', '*Inject Unexpected Items*', '*Employ Probabilistic Techniques*', '*Manipulate Timing and State*', '*Collect and Analyze Information*' and '*Subvert Access Control*'. The domains of attack are the '*Software*', '*Hardware*', '*Communications*', '*Supply Chain*', '*Social Engineering*' and '*Physical Security*'.

APs are defined as generic representations of attacks from the point of view of an attacker that represent the critical features of the attack. Several AP templates have been proposed considering various attributes. These attributes include goals, preconditions, post-conditions, perpetrators, motivation, targets, methods, prerequisites, resources, skills, knowledge, etc. (Mischel, 1971).

Although CAPEC is not a threat modeling approach, the CAPEC's APs and categories are widely used today for the identification of systems' vulnerabilities, threat analysis and modeling. Besides, CAPEC is also an asset in cyber security learning and training, as it can be used for teaching the scope of the threat landscape. CAPEC's APs represent the common attack methods and techniques, and they use the problem-solution in a specific context approach of the design patterns. More specifically, the problem in an AP is the goal that the attacker wants to achieve; the solution states the actions the attacker follows to perform the attack; the context describes the prerequisites of the attack and the information of the environment such as the technical details. CAPEC's APs are presented in a semi-structured and human-readable manner and they are usually translated into more

formal representations to be utilized in threat analysis and simulation tools. CAPEC has three different types of attack patterns according to the provided details: the *meta*, the *standard* and the *detailed*. The meta-type includes APs described in an abstract manner particularly useful when designing a system and a set of general attack patterns is required. The *standard* APs provide details on the attack technique and strategy employed, whereas the *detailed* APs focus on describing in detail the flow of tasks performed. Each AP might have a more abstract parent (i.e., a *meta* AP) and more detailed children APs (i.e., *detailed* APs). For example, the *standard* type AP '*Host Discovery*' belongs to the '*Collect and Analyze Information*' mechanism of attack and the '*Software*' and '*Communications*' domains of attack; it is a child of the '*Footprinting*' *meta* AP and it has several children APs of *detailed* type (e.g., '*TCP SYN Ping*', '*ICMP Echo Request Ping*' etc.).

2.4.2. Cyber Kill Chain

Lockheed Martin's Cyber Kill Chain (CKC) (Martin, 2014) is a model (*Figure 2-14*) used in a military context before it was introduced in cyber security. The CKC model describes the phases that an adversary follows when performing an advanced persistent threat (APT) cyber security attack. APT attacks require sophisticated and long-term actions, deep knowledge of cyber security concepts and skills, and control over the appropriate resources and tools. In APT attacks, adversaries need to gain deep knowledge of the system before launching the attack, a stage that can last for a long period. Then, adversaries plot a carefully planned attack aiming at making the most out of the attack while eliminating the risk of exposure. During an APT attack, the adversary aims at gaining access to the target system for an extended period to achieve her/his ultimate goals (e.g., data exfiltration) and she also makes the appropriate precautions to remain undetected.

***Figure 2-14**. The Cyber Kill Chain Model (Martin, 2014)*

The CKC model consists of the following phases:

1) *Reconnaissance*: the adversary performs activities of research, information gathering, identification and selection of target(s).
2) *Weaponization*: the adversary, based on the information gathered in the reconnaissance phase, couples an exploit and creates malicious payload to distribute to the target(s) (i.e., the weaponized file such as pdf and docx documents).
3) *Delivery*: the adversary sends the weaponized file to the victim by email; distributes it through a website; or delivers it physically through removable media and devices (e.g., USB flash drive).
4) *Exploitation*: includes the triggering of the malicious code in the target machine.
5) *Installation*: a remote access trojan or backdoor is installed in the target machine, to allow attackers to maintain persistence inside the target's system.
6) *Command and control (C2)*: the malware establishes a covert communication channel with the adversary.
7) *Actions on objectives*: includes the subsequent actions the adversary performs to achieve her/his goals.

Nowadays, the CKC model is widely used to facilitate the analysis of APT attacks by describing the structure of such attacks. The CKC model is used to raise the awareness of cyber security experts, detect vulnerabilities in information systems, develop mitigations against the possible threats and prioritize investments in organizations' security. Besides, it is used to produce attack scenarios and simulations of incidents to train the cyber security personnel (e.g., incident response team).

2.4.3. National Cybersecurity Workforce Framework

The National Cybersecurity Workforce Framework (NCWF) (Newhouse et al., 2017) is a common definition of cyber security workforce roles, tasks, knowledge, skills, and abilities (KSAs) elaborated by the National Initiative for Cybersecurity Education (NICE), a division of the National Institute of Standards and Technology (NIST). The NCWF is a valuable resource for cyber security educators, as it connects the production of the cyber security workforce with the cyber security education programs. NCWF maps the tasks and the corresponding KSAs, required by cyber security professionals to perform their duties, to cyber security work roles aiding in elaboration programs, teaching materials and suitable learning aims and objectives.

The NCWF consists of seven (7) high-level categories including:

1) Securely Provision: includes job roles responsible for overseeing, conceptualizing, evaluating, and building secure information systems and networks utilizing concrete policies, processes, and controls.

2) Operate and Maintain: encompasses specialty areas of administrators, analysts, and knowledge managers that install, configure, analyze, test and maintain hardware and software components of the systems.
3) Oversee and Govern: provides leadership, management, direction or development and advocacy so the organization may effectively fulfill cyber security requirements. Work roles range from consultants and policy makers to managers and educators.
4) Protect and Defend: covers specialty areas responsible for responding to cyber security incidents; and identification, analysis and mitigation of threats to internal information technology systems and networks.
5) Analyze: involves work tasks of analysis and assessment of information collected from multiple sources (e.g., agencies, cyber criminals, foreign intelligence entities) to identify threats, vulnerabilities, targets and potentials for exploitation.
6) Collect and Operate: involves work roles focusing on operations that deny malicious intentions; deceptive threat actors; and collect information that may be used to develop intelligence.
7) Investigate: focuses on the investigation of cyber security incidents or crimes related to information technology systems and networks by collecting, analyzing and processing digital evidence.

Each of the aforementioned categories is divided into specialty areas comprised of work roles. For example, the 'Protect and Defend' category is divided into the 'Cyber Defense Analysis', the 'Cyber Defense Infrastructure Support', the 'Incident Response' and the 'Vulnerability Assessment and Management' specialty areas. Subsequently the 'Vulnerability Assessment and Management' specialty area is comprised of the 'Vulnerability Assessment Analyst' work role. According to the NCWF, the vulnerability assessment analyst "*performs assessments of systems and networks within the network environment or enclave and identifies where those systems/networks deviate from acceptable configurations, enclave policy, or local policy. Measures effectiveness of defense-in-depth architecture against known vulnerabilities*" (Newhouse et al., 2017; pp 20). Each work role is associated with the cyber security tasks and the KSAs required to successfully perform those tasks. For example, the tasks of the vulnerability assessment analyst include:

- Analysis of organization's cyber defense policies and configurations and evaluation of compliance with regulations and organizational directives.
- Conduct and/or support of authorized penetration testing on enterprise network assets.
- Recommendations that regard the selection of cost-effective security controls to mitigate risk (e.g., protection of information, systems and processes).

Additionally, the KSAs of the vulnerability assessment analyst include:

- Knowledge of computer networking concepts and protocols, and network security methodologies.
- Knowledge of cyber threats and vulnerabilities.

- Skill in conducting vulnerability scans and recognizing vulnerabilities in security systems.
- Skill in assessing the robustness of security systems and designs.
- Ability to identify systemic security issues based on the analysis of vulnerability and configuration data.

2.5. Chapter Conclusion

As the presented study is highly interdisciplinary, this chapter presents and clarifies central concepts of this thesis, which belong to different domains. The first part of this chapter (section 2.2) presents an overview of serious games, game-based learning, and the connection of SGs with the related concepts. The outcome of this study is that in some cases there are no clear limits between SGs and the related concepts (e.g., SGs .vs Educational games, SGs .vs Training Simulators). According to Taylor, 2014, the concept of training simulators, which is central to this study, when viewed from the experiential perspective compare to SGs. However, on the contrary with training simulators, SGs embrace factors that transform them into games (elements which enhance entertainment, competition, fiction, etc.) and they focus more on psychological-cognitive fidelity by making the learner evoke a sense of believability on a psychological level without involving a high degree of realism (Taylor, 2014). Section 2.2 also analyzes central concepts of this thesis such as the learning, training and foundational knowledge, and stresses the importance of SGs in learning and training.

The second part of the chapter (section 2.3) presents the existing SGs design frameworks and models, which depict the high-level aspects that should be considered during the design of SGs. Most of the presented frameworks and models form a guide for the development of SGs that embrace the necessary features for successful learning and training, but they do not provide a methodology to associate the high-level aspects with the low-level game's components. An exception is the ATMSG model, which analyzes the game's components under the gaming and pedagogical perspectives and associates them with the high-level aspects of SG. Finally, the last part of the chapter (section 2.4) presents the cyber security standards adopted in this study, their role and the reasons that they were selected.

3. Cyber Security Game-based Learning and Training

3.1. Introduction

This chapter investigates the studies on cyber security game-based approaches for learning and training. As game-based learning and training is a recent approach for cyber security education, there are few studies in this field (Hendrix et al., 2016). A thorough inspection of the literature revealed only a limited set of studies on cyber security game-based learning, focusing on diverse target groups and methodologies. Moreover, the lack of design standards and conceptual analysis tools was pointed out. However, the identified approaches were analyzed and the structure of cyber security learning and training approaches that utilize gamification and game-based learning notions was investigated. Additionally, the structure of the identified approaches was decomposed into their respective elements, the elements were categorized, and the relations of the decomposed elements were specified. Finally, the concept map of game-based approaches' key elements was constructed.

3.2. Current Approaches

Researchers in (Nagarajan et al., 2012) explore the field of cyber security training. Their approach is based on Anneta's design framework for serious games (Anneta, 2010), presenting considerations for the design of games for cyber security training. The proof of concept of their study is the CyberNEXS gaming tool, a multi-mode game aiming at teaching cyber security in colleges and high schools. In their work, authors present the shortcomings of the non-game-based learning approaches; the design elements of their approach; and the improvements that can be made on their approach to upgrade the engagement, the entertainment and the educational impact.

Compte et al. in (Compte et al., 2015) rely on the LM-GM model (Arnab et al., 2015), the DPE framework (Winn, 2008) and the FDF framework (FDF) (De Freitas & M. Oliver, 2006). In their study, they present a renewed approach to the design and development of cyber security serious games aiming at raising awareness to novices. Additionally, they discuss the limitations of the current approaches including the issues of considering only the serious games deployment in formal settings, e.g., colleges, corporations, schools. Reflecting on these limitations they propose a six-step design framework for the development and deployment of games particularly in informal contexts.

Vykopal and Barták (Vykopal & Barták, 2016) present their study on the design and deployment of a prototype cyber security game for penetration testing training in a networked environment. The game was deployed on the KYPO Cyber Exercise & Research Platform (Čeleda et al., 2015), a platform that forms a virtualized environment for the modeling and simulation of complex computer systems and networks. According to their approach, the training activity of the game is decomposed

into individual levels that learners have to accomplish to satisfy specific learning objectives. The researchers' efforts focus on scaffolding the learners towards the objectives and on user actions prediction. The users' scaffolding facet is featured through a hints system that measures time, presents optional hints when learners struggle to solve an exercise and penalizes them with negative points. On the other hand, the predicting users' actions facet is featured through an advanced logging system that records participants' learning activities. Researchers conducted sessions with their prototype game to evaluate their approach. Their experiments and results led to some notable considerations regarding cyber security game design.

Allen and Straub in (Allen & Straub, 2015) presented an approach for the construction of an effective cyber warrior's training program using various digital and physical games. They identified the pitfalls of the current training models and based on these findings they proposed a framework that augments current training programs. Their approach suggests a gamified integrated and layered solution ranging from always-on cell phone games to full-scale operational exercises. Additionally, it provides continuous training and proficiency feedback to assess learning and mission readiness on an individual and team basis. According to their strategy, physical flashcards and phone games introduce or reinforce foundational knowledge and skills. While trainees advance their skills, they traverse towards more realistic and immersive environments (e.g., Lumosity-like games) that require reinforcement of soft skills, such as collaboration. Finally, trainees participate in simulation multiplayer exercise games to prove individual and team cyber proficiency. In these exercises players collaborate or compete against humans or computers in a context defined by a specific scenario.

Amorim et al. in (Amorim et al., 2013) discuss a gamified training system for cyber defense that complies with the training needs they encountered. Researchers claim that traditional training schemes fail in the cyber security domain because the cyber world changes continually and rapidly. They claim that new training approaches are needed to provide new 'on demand' material during the confrontation of a new threat. Due to this 'on demand' requirement, a component is necessary that will dynamically keep track of trainees' profiles and backgrounds. Moreover, the authors claim that serious games and simulations are more suitable for cyber security training for which agile philosophy needs to be adopted.

Obviously, the presented approaches are not aiming at the same target groups and do not have the same structure. The former two address the design of a serious game in cyber security training for novices, whereas the latter three study the formation of training programs for cyber security professionals. Moreover, the approach of Allan and Straub uses a collection of physical and digital games. However, all the studies lie in a common domain and additionally the former two approaches have foundations on general serious game frameworks. Thus, we included them in our study.

3.3. The Key Elements of Cyber-Security Game Based Approaches

Figure 3-1 depicts the proposed concept map of key elements of the cyber security game-based approaches referred to the presented approaches. The concept map contains numerous elements that reflect the diversity of the topic. Particularly, it consists of 78 concepts organized in 8 segments that share 14 cross-links represented in *Figure 3-1* with dashed lines for readability reasons. The concepts in the proposed concept map are organized in a nonhierarchical network structure, as cyber security game-based learning is a complex topic containing several concepts with multiple connections among them. Consequently, the concept map is logically organized in two clusters. The inner cluster contains the central node of cyber security game-based learning characteristics and the general concepts, whereas the latter is depicted as labels of the concept map's segments. The outer cluster includes more specific notions belonging to the domain of each general concept.

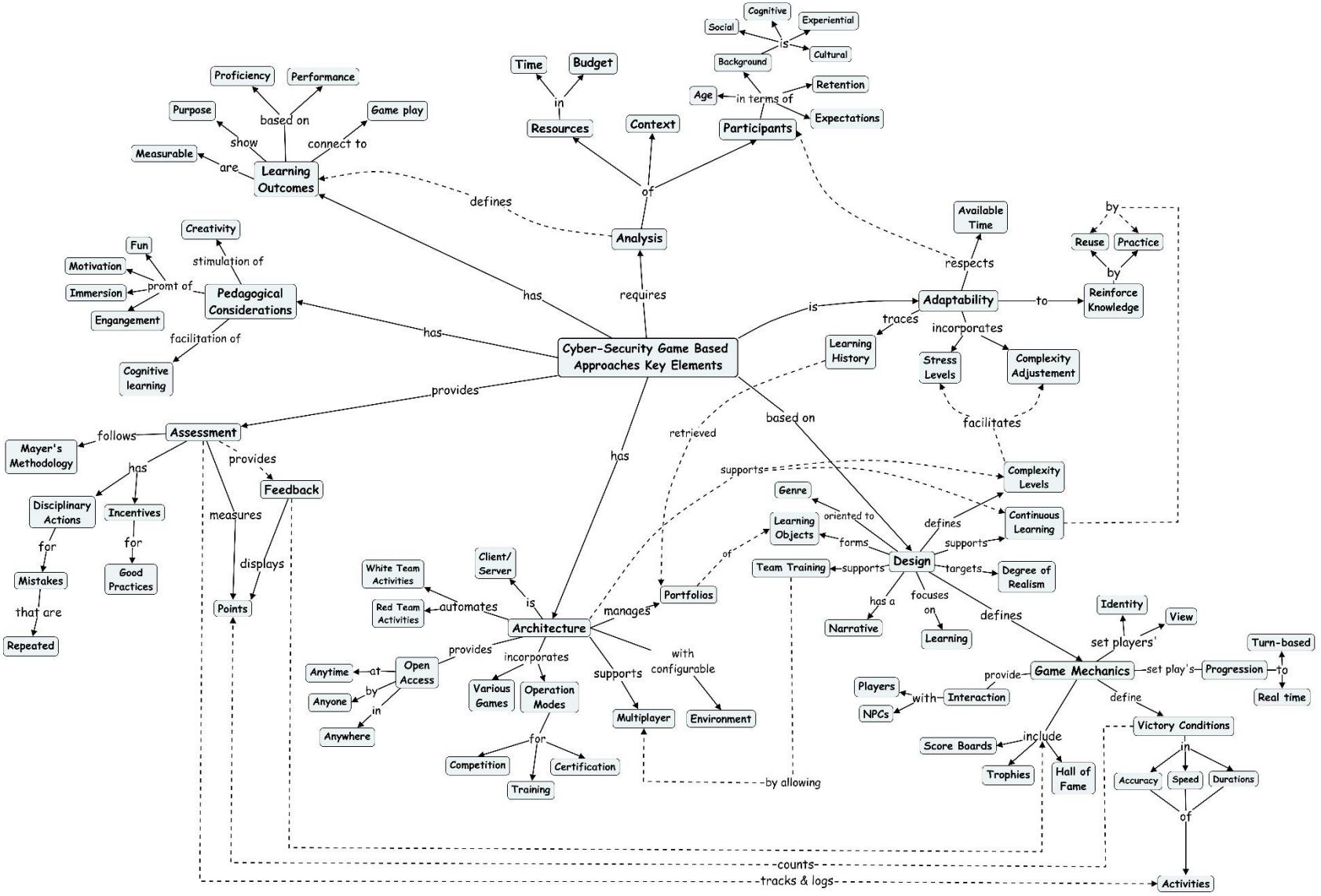


***Figure 3-1**. Concept Map of Cyber-Security Game Based Approaches Key Elements*

3.4. Rational and Reflections

3.4.1. Pedagogical considerations segment

This segment contains six notions associated with the educational impact of cyber-security game-based approaches. The concepts of *immersion*, *engagement*, *motivation,* and *fun*/*entertaining* are very important as they keep the players focused on the game. At the same time, they provide learning and training opportunities while their infusion in the games constitutes a significant challenge for game designers. Allan and Straub state in (Allen & Straub, 2015) that *engagement* and *motivation* can be leveraged in a competitive and challenging environment through game mechanics (e.g., scoreboards), whereas Nagarajan et al. in (Nagarajan et al., 2012) denotes that *motivation* can be achieved by creating goals like financial prizes or certificates that will enhance players' professional career. Moreover, Compte et al. in (Compte et al., 2015) claim that *immersion* is not the privilege of simulation games that feature realistic environments, but also it can be achieved with other techniques like the players' emotional involvement and the use of themes providing narratives and appropriate look and feel to the games (Nagarajan et al., 2012). A high degree of immersion and engagement can be targeted by following a distinct game genre or a combination of them concerning games' objectives and the target group characteristics. For example, games of the fighting genre are simple, direct but engaging, while role-playing games lead to user emotional involvement (Nagarajan et al., 2012). Finally, Allan and Straub set forth the significance of the fun concept in achieving objectives related to rote memorization of important information for cyber-warriors, e.g., commands and port-to-protocol mappings.

By reviewing pedagogical considerations of the field, we observe that only researchers in (Nagarajan et al., 2012) implicitly refer to a learning theory, the cognitive learning theory. Cognitive learning arises from the *Instructional* principle of Annetta's framework (Anneta, 2010) stating that players should use existing knowledge and skills to assimilate the new ones, a proposition that is consistent with Piaget's ideas on cognitive development (Piaget, 1952).

3.4.2. Analysis segment

This segment contains notions related to the identification of learning outcomes, the players' characteristics, the context of the game, and the available resources in terms of time and budget (Compte et al., 2015). The context is described in (De Freitas & M. Oliver, 2006) as the combination of historical, political, economic factors, the availability of specific resources and tools. Additionally, it considers the characteristics of the instructors and the availability of technical support. Besides, Compte et al. (Compte et al., 2015) state that the notions related to the players' characteristics and the learning outcomes relate to the context in which serious games are going to operate (Compte et al., 2014). Moreover, Amorim et al. state that serious games purposed for delivering on-demand

content, i.e., the material identified during emergencies, should include features that dynamically trace the players' training history.

3.4.3. Learning Outcomes segment

This segment includes five notions related to the learning outcomes (LOs) of the cyber-security game-based approaches. Allan and Straub indicate the importance of identifying and defining measurable and clear-purpose learning outcomes. Additionally, they distinguish the learning outcomes in terms of proficiency and performance. Proficiency LOs relate to a high degree of skills and expertise while performance LOs are just a measure of capabilities under particular conditions (Allen & Straub, 2015). Subsequently, the LOs of game-based approaches aiming at training cyber-security professionals should focus on the proficiency of related KSAs and not only on the performance of exercising elementary knowledge and skills. Finally, learning outcomes constitute a positive engagement factor when they are well-defined and connected to the gameplay through the game mechanics (Arnab et al., 2015).

3.4.4. Design and Game Mechanics segment

The *Design* and *Game Mechanics* segments contain numerous concepts related to the design and the mechanics facets of cyber-security game-based approaches. Most of these concepts are based on common serious game design frameworks and methodologies. However, some concepts are considered discrete in the design of cyber-security games for professionals, as they expose the distinctive challenges of the field. The *Continuous Learning* concept is a significant notion in cyber-security training, as it facilitates the need for frequent training and exercise that will allow cyber-security experts to be mission-ready (Winn, 2008). Specifically, continuous learning will not allow the foundational knowledge of cyber-security experts to decay. Furthermore, the opportunities and motivation for continuous learning will help the cyber-security experts to increase skills in terms of speed and effectiveness (*speed*, *accuracy* and *duration* concepts in the *Game Mechanics* segment) towards confronting a problematic case. For example, a training approach that implements a continuous lifecycle of learning, updating and reinforcement can foster trainees' skills to analyze a problematic case and envisage a solution; perform the proper sequence of actions on time in incident response cases; enter instantly the appropriate commands and options to carry out certain activities. Moreover, the concept of *Continuous Learning* in relation to the *Complexity Levels* concept facilitates the development of approaches that employ learning and training complexity levels. More specifically, such approaches implement activities that increase their complexity and demand collaboration in team-based sessions (*Team Training* concept). Additionally, the *Realism* concept is a critical factor that affects the design process and predicts positive learning outcomes (Allen & Straub, 2015). For example, a high degree of realism in game-based approaches may include simulations of computers or networks, whereas a lower degree of realism would point to games in which events and contexts are simulated e.g., the CyberCIEGE and the CyberProtect games. Besides,

cyber-security training should consider on demand formation of learning content and learning objects to facilitate the confrontation of zero-day threats (Amorim et al., 2013). Finally, the supplementing information needed to be provided during a cyber-security training game session (e.g., copy of textual commands) should be delivered inside the game platform, to avoid causing learning distractions (Vykopal & Barták, 2016).

3.4.5. Architecture segment

This segment depicts concepts related to the operation facets and game structure. The *Architecture* segment is affected by the *Continuous Learning* and *Complexity Levels* concepts, as they put forward the requirement to provide training opportunities to anyone always. Specifically, an architecture aiming at instantiating the layered training strategy has to encompass various increasingly complex and collaborative games (e.g., flashcard games, aptitude games, challenge-in-a-box exercises like Cyber Flag or Cyber Guard (Allen & Straub, 2015)). Moreover, approaches aiming at the reinforcement of cyber-security experts' foundational knowledge needed to deliver training games in multiple places and/or devices like the computers in a training laboratory or the personal mobile devices of the trainees. On the other hand, the CyberNexus architecture (Nagarajan et al., 2012) provides various modes of operation according to the objectives of the session (i.e., training, certification, and competition). Besides, the architecture of a serious game for team-based training has to support multiplayer functionalities and possibly automated activities of non-player teams and characters (e.g., red and white teams in a capture-the-flag competition setting). Finally, the architecture of serious games that include adaptive features has to manage participants' portfolios (e.g., portfolio of learning objects (Amorim et al., 2013)) to retrieve their learning history.

3.4.6. Adaptability segment

This segment contains notions related to the adjustment of serious games functionality according to the predefined elements. Adaptability elements are the participants' learning history (the *Learning History* concept), the available time (the *Available Time* concept) and the main concepts of the analysis segment (i.e., the defined learning outcomes), the participants' characteristics and the current learning context. According to the adaptability elements, a serious game may present games of different genres: adjust the gameplay experience to target complex learning objectives (the *Complexity Adjustment* concept); involve an increased number of conditions that a learner has to consider in order to solve a problem (Greitzer et al., 2007); present repeatedly similar content for knowledge reinforcement (reinforce *Knowledge* concept); select the optimal stress level of the game (*Stress Levels* concept).

3.4.7. Assessment segment

This segment includes the *Feedback* sub-segment and various concepts related to the estimation of players' performance reflecting their progress to the fulfillment of the learning objectives. Assessment and feedback can be performed dynamically, while players play the game, based on the game mechanics for the estimation of points and the tracking and logging of players' activities. More specifically, users' efforts can be tracked (the *tracks & logs Activities* proposition) in order to measure the speed and accuracy of carrying out the actions required towards the solution of a challenge, as well as the time taken to deal with the problematic situation (the *Duration* concept). The details of users' activities are logged to provide feedback to the players and the instructors (Anneta, 2010). Moreover, the assessment facet of the game can also trigger disciplinary actions for the players that repeat the same mistakes even after training and reinforcement, whereas it can reward players with good practices (Nagarajan et al., 2012). According to Compte et al. (Compte et al., 2015), assessment can also be carried out by following Mayer's methodology (Mayer, 2012), an evaluation method for serious games carried out in three distinct phases before, during and after the game. Finally, assessment can also be performed with the use of tests, surveys and questionnaires probably in pro-game and post-game phases to avoid learning distractions.

3.5. Chapter Conclusion

In this chapter, the literature's game-based approaches for learning and training were presented and decomposed into their elements. A concept map with the characteristics of cyber security game-based approaches are presented. Furthermore, the specific requirements of the cyber-security field were stressed and observations and suggestions regarding the development of effective cyber-security game-based paradigms were provided. The necessity of filling the gaps of the standard KSAs required for the cyber-security professionals and the exploitation of the appropriate learning theories for the field was highlighted. The concept map of cyber security game-based approaches key elements (*Figure 3-1*) along with the analysis presented in this chapter can be put into effect in the analysis and the preliminary evaluation of new cyber security game-based approaches. The work presented in this chapter is utilized in the analytical evaluation scheme elaborated in the second iteration of phase 4 (*Figure 1-1*) presented in chapter 10.

4. LIVE COMPETITIONS

4.1. Introduction

In this chapter, the field of live competitions, such as Capture the Flag (CtF), is reviewed and analyzed. Live competitions are an integral part of the cyber security domain and they are utilized by cyber security educators in educational contexts. That is because live competitions provide noteworthy experiences for the participants while offering both hands-on practice and entertainment. Incorporating live competitions in the learning procedure, adds real-time value that facilitates motivation and deep involvement. Moreover, it introduces the crisis factor associated with many security situations. However, various issues have been identified that limit the pedagogical values of live competitions. Under this perspective, a thorough reading of the literature revealed the lack of a conceptual framework to help in effectively studying the characteristics of live competitions and provides a basis for improving their pedagogical utilization. Aiming at performing a conceptual analysis as a basis for improving their pedagogical utilization of live competitions, we investigated several live competition paradigms, and we analyzed their structure by decomposing them into their respective elements and defining their relations. Moreover, we recorded the possible obstacles related to the pedagogical utilization of live competitions and grouped them into distinct categories. As a result, we constructed a concept map of the technological and pedagogical characteristics of live competitions. Based on the proposed concept map and the recorded obstacles, we formed a comparative evaluation scheme that we employed on three live competition approaches from the literature to reveal their value with respect to the educational impact.

4.2. Description

Live competitions are contests in which participants compete on their technical skills and knowledge in real-time. Such contests have been organized for many years since the DefCon CtF, which was the first one employed in the nineties. The community appreciated the impact of such events and various competitions have been designed and developed, ever since. Nowadays, there are more than seventy CtF competitions organized on an annual basis (Ctftime.org, 2016), whereas there are numerous small-scale exercises organized in colleges and organizations that are not listed in a CtF ranking site.

According to the format and scale of the event, live competitions can be addressed as competitions, exercises, or games. In addition, they have several directions that require participants to attack other teams, defend the team's settings, or independently solve challenges in a so-called jeopardy style event (Bratosin, 2014). Consequently, in attack and defend modes participants are required to interact directly with adversaries whereas in jeopardy mode events they act independently (Vigna et al.,

2014). Many competitions use a combination of the aforementioned modes by setting the jeopardy style events like the playoffs of the competitions, and the attack/defense format in the finals.

Contestants take part in such events either as individuals or as team members. The knowledge barrier of the events may require participants to have a good background and experience in scripting languages, (e.g., Perl or Python), reverse engineering, operating systems, networking, system administration, and application services (Cheung et al., 2012) (Mirkovic et al., 2015) in order to be competitive. Events occur on either physical or virtual machines and their development may require participants' physical co-location or allow remote connections by contestants from around the world. According to the motive of the competition, participants may be provided with numerous settings. Such settings may include network topologies, configured or deliberately misconfigured machines, operating systems, and application software known or unknown prior to an event. Participants may also be provided with certain privileges and rules that permit or prohibit the use of certain tools and techniques (e.g., denial of service attacks and flooding). These settings vary according to each event's specified scenario. Scenarios typically demand to 'capture' a specific file, called 'flag', which is used as a proof that contestants have compromised a service or solved a challenge. Flags usually are long, random strings that are hard to guess (Davis et al., 2014). They contain information and timestamps regarding the team, the host and the service they belong, their creation time and validity periods (Doupé et al., 2011). Some scenarios, in particular, may require compromising the settings of an adversary team, defending a system's services and files, attacking web sites, carrying out forensics investigations, reverse-engineering programs, attacking encrypted tokens, etc.

Participant assessment is usually based on a scoring scheme. According to the competition form and scenario, a participant's score increases when, for example, s/he manages to acquire a flag from another team or when s/he responds to a challenge correctly. On the contrary, the score of a team decreases when, for example, an adversary captures and submits one or more team's flags or its system's services become unavailable. Usually, the scoring scheme includes the employment of automated score-bots, whereas sometimes participants are required to write up their actions or evaluators supervise individuals' progress. Some scoring schemes are not typical; e.g., the iCtF scheme (Childers et al., 2010), which introduced the concept of money that allows teams to use the event's infrastructure to earn points and the notion of toxicity that constitutes a measure of damage effectiveness caused in a specific service (Doupé et al., 2011).

The scoring system usually employs a setting, e.g., a web server, that includes a repository of flags or a service for automated submission testing and a displaying score device that provides feedback to all the concerned parties. The feedback is a critical aspect in the operation of live competitions (Dabrowski et al., 2015). Contestants that compete in the attack or jeopardy mode of a competition, receive feedback directly at the time that they submit a flag or other token, e.g., source code, to the scoring system (Chung & Cohen, 2014). When competing in the defend mode, they get feedback indirectly through the updates of scores that are based on the information provided by score-bots and on the submissions of the adversary participants (Werther et al., 2011).

Moreover, competition organizers always cater for the reliability and security of the contests so that participants will not be able to cheat the scoring schemes. Nevertheless, participants may be able to cheat by using prohibited tools and techniques or by successfully attacking the score services or by applying tricks. Specifically, contestants may try to:

- brute force flags,
- attack the scoring system to modify participants' records (Chung & Cohen, 2014),
- tamper their own flags to make sure that no adversary team will submit them to the scoring system (Vigna et al., 2014),
- make their services available only to the bots and deny access to everyone else (Vigna et al., 2014).

Live competitions represent a useful pedagogical utility, particularly valuable in the multidisciplinary and complex domain of cyber security education. Their pedagogical significance has been widely stated in many studies (Chothia & Novakovic, 2015). In particular, it has been stated that live competitions provide the means to motivate participants to focus on the cyber security field by engaging participants in hands-on practices. They also include an entertainment factor as they constitute a gamified environment, in which contestants can compete, cooperate, and express their feelings. Furthermore, live competitions can harvest to competitors the willingness to engage in continuous self-directed learning, experimentation, and development in order to cope with the increasing demands of harder challenges and events (Carlisle et al., 2015). Competitions that include the attack factor tend to be more enjoyable (Davis et al., 2014) and motivating for the participants.

Live competitions promote experiential learning as the participants are engaged in hands-on activities, e.g., when they make efforts to reach competitions' goals (Rege, 2015). Learners observe the results of their actions while getting feedback during the course of an event. For example, an effectively applied defensive policy can protect the participant's system from attacks, whereas an unsuccessful policy can lead to loss of points. In both cases, participants will reflect on their ideas and actions. According to the event's settings and rules, participants might share and discuss their ideas and feelings with their teammates or peers and instructors. Sometimes they are required to report their ideas or discuss them after the end of the process. In this way, learners create mental models and generalization concepts on what part of the event they accomplished. Acquired concepts could then be applied in different experiences and settings in subsequent competition challenges or in real-world situations (Konak et al., 2015).

Live competitions are consistent with problem-based learning as they require participants to apply their knowledge and skills to solve authentic problems. They also support situational learning by transferring capabilities and experiences in realistic situations (Pusey et al., 2014). Besides, team competitions can embrace socio-cultural learning approaches that can maximize their educational impact through collaboration, communication, and teamwork (Rege, 2015). Such approaches can train the participants to act effectively in team settings and subsequently prepare them to work and cooperate in similar settings of organizations and departments (Mauer et al., 2012).

4.3. The Key Technological and Pedagogical Characteristics of Live Competitions

Figure 4-1 depicts the proposed concept map of live competitions' key characteristics located in the selected papers of the literature. Although the proposed concept map includes only the typical key characteristics of live competitions, it contains numerous elements that reflect the diversity of the topic. In particular, the concept map consists of 77 concepts organized in 6 segments that share 14 cross-links represented in *Figure 4-1* with dashed lines for readability reasons.

The concepts in the proposed concept map were organized in a nonhierarchical network structure, as live competition is a complex topic containing several concepts with multiple connections among them. Consequently, the concept map is logically organized in two clusters. The inner cluster contains the central node of live competitions characteristics and the general concepts. The latter is depicted as labels of the concept map's segments listed in *Table 4-1*. On the contrary, the outer cluster includes more specific notions belonging to the domain of each general concept.

The rationale of the proposed concept map segments is described below:

- *Contest Form*: includes concepts related to the mode (format) of the event.
- Pedagogical *Benefits*: contains the notions associated with the educational impact of the competition.
- *Participant(s)*: includes characteristics associated with the contestants, i.e. profile in terms of background and experience in cyber security, and the manner they affect the live competitions' format.

***Table 4-1**. Number of Nodes and Cross-Links per Segment*

Segments	***# of nodes***	***# of cross-links***
Contest Form	13	7
Pedagogical Benefits	9	5
Participant(s)	13	5
Infrastrucutre	14	4
Preparation	5	0
Policies & Mechanisms	23	7
Sum	77	28

- *Infrastructure*: depicts concepts related to the framework of the competition in terms of devices and software.

- *Preparation*: involves notions related to the demands for the set-up of the competitions.
- *Policies & Mechanisms*: depicts the concepts related to the rules and the processes applied to ensure the reliability and the fairness of the competitions and to perform the evaluation of the participants.

Observing *Figure 4-1*, it can be noticed that *Policies & Mechanisms* is the most substantial and complex segment, as it involves the *Assessment* and *Reliability* sub-segments and numerous concepts, including the *Flag(s)* and *Score* concepts that have a large number of relationships. Besides, the *Policies & Mechanisms* segment has a considerable number of cross-links comprised of four relations to the *Infrastructure* segment, two to the *Participant(s)* segment, and one to the *Contest Form* segment. The segments *Policies & Mechanisms* and *Infrastructure* also share the *require* relation to the *Preparation* segment. The fact that *Policies & Mechanisms* and *Infrastructure* segments have multiple cross-links, and they are both related to the *Preparation* segment is considered ordinal as the former is instantiated and applied on the devices of the latter.

Likewise, the *Contest Form* segment is a crucial factor in the implementation of live competitions as it encompasses numerous cross-links and critical elements. These elements need to be defined early in the design phase of an event, as they can affect the entire development process. More specifically, the *Contest Form* segment includes the *Scenario* concept that has an influence on the policies and mechanisms necessary to implement. It embraces two cross-links to the *Scales* and *Teams* concepts of the *Participant(s)* segment that symbolize the number of contestants contributing to the event and whether they are separated in teams. Additionally, it includes the contest and attack notions that affect the pedagogical impact of live competitions (*Pedagogical benefits* node), through the formed cross-links to the *Motivation* and *Enjoyment* concepts. Finally, pedagogical benefits are also designated by the *Participant(s)* segment that includes the notion of *Teams*.

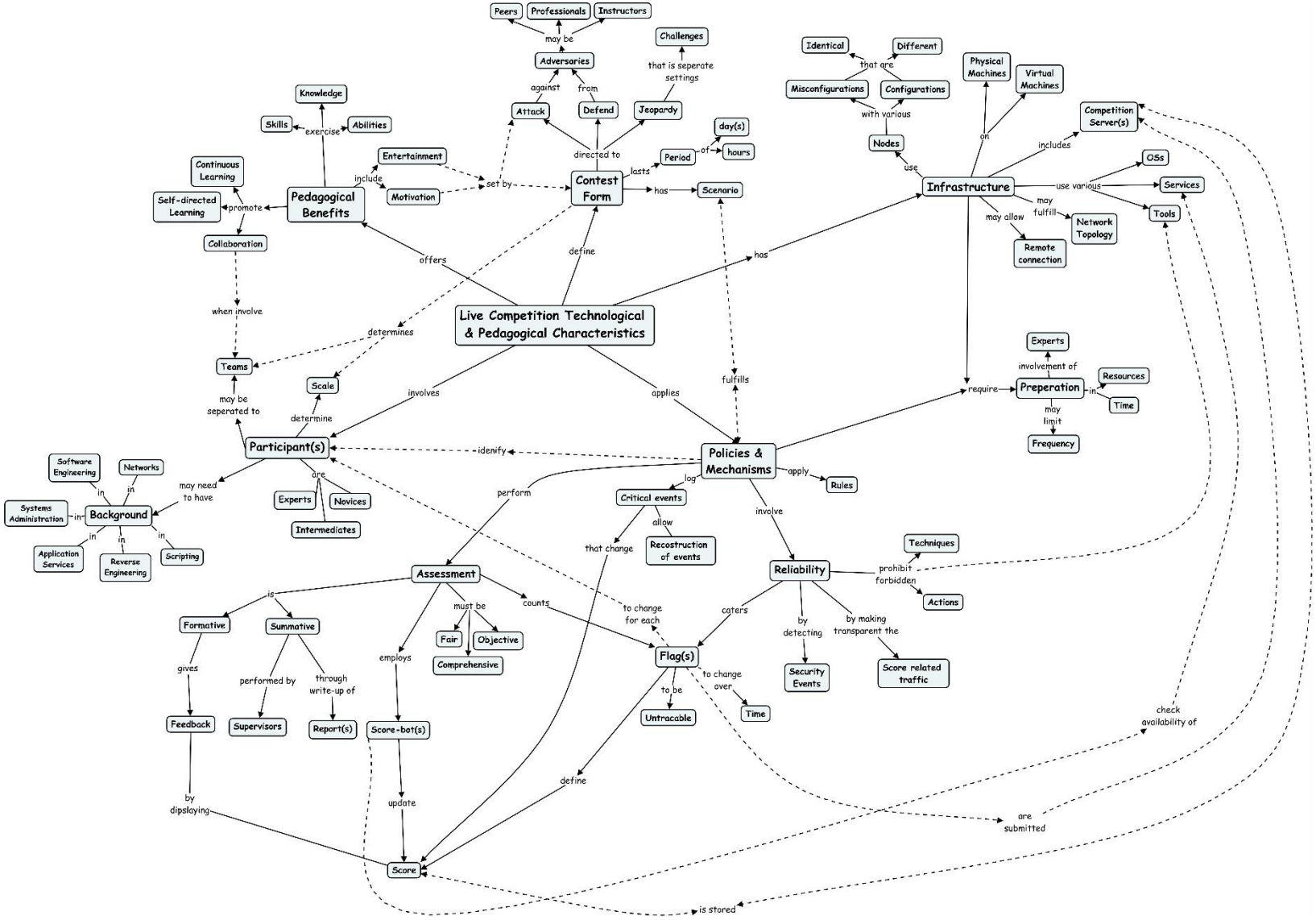


Figure 4-1. *Concept Map of Live Competitions Technological and Pedagogical Characteristics*

4.4. Identified Problems and Issues

During the past years, various problems and issues of live competitions have been identified in the literature. We reviewed and grouped them into three categories: drawbacks in the competitions' aims, learning obstacles affecting their value, and concerns on competitions' organizational and functional issues.

1) Drawbacks in competitions' aims:

a) *Contests aim is to measure skills*: A competition, in general, is concerned with the measurement of skills while participants acquire knowledge and skills (KSAs) in a self-directed and unstructured manner. Conversely, an educational approach in cyber security aims at a different purpose. More specifically, it aims in setting the environment and defining the processes that will guide learners to adapt by acquiring new knowledge, skills and abilities (Silva et al., 2014).

b) *Fail to address the management of settings realistically*: The aims of the contests are unlinked to the day-to-day management of network settings and services. Participants use ad-hoc methods and strategies, and they often adopt unsuitable behaviors because they deploy extreme defense approaches. These approaches do not take into account the operational costs of the systems, i.e., amount of memory, CPU time and size of log files, and thus - in real settings - they are inapplicable. In addition, contestants often take into consideration only the initial setup of their system, but they do not pay attention to keeping their system up-to-date, implementing disaster-recovery policies, and employing effective backup schemes (Catuogno & De Santis, 2008).

c) *Diversity of topics is not supported*: Live competitions usually focus on a restricted set of topics, e.g., performing exploits or protecting vulnerable code (Mirkovic & Peterson, 2014), while they do not address particular subjects in their aims, e.g., threats related to the availability of resources and brute-force attacks. This happens because live competitions are limited by certain characteristics such as the duration of events or because the contest organizers are biased towards certain types of problems (Chung & Cohen, 2014). As a result, organizers suppress certain aspects by rule sets, e.g., intentional loss of availability, and participants do not practice in handling them (Koch et al., 2012).

2) Learning Obstacles:

a) Not calibrated to participants' needs: Nowadays, there are many competition events available, some of which are oriented towards specific profiles for their participants. For example, DefCon is organized for cyber security experts that are experienced in offensive tactics, whereas CSAW (Cyber Security Awareness Week) aims at novice students new to cyber security concepts. Nevertheless, designers of competitions' challenges still face issues in deciding and adjusting the right level of difficulty. For instance, sometimes they try to create

difficult challenges by making the solutions convoluted. Consequently, participants often are overwhelmed and discouraged, whereas in other cases they are not sufficiently challenged (Chung & Cohen, 2014) (Silva et al., 2014).

b) Not an experiment environment: In competitions where participants compete against each other, there are no comparable and repeatable results related to the contestants, either as individuals or as a team. Participants do not have the opportunity to refine failed policies instantly and try different approaches to receive new feedback (Koch et al., 2012).
c) Partial credit is not supported: Scoring schemes usually assign points to the competitors, when they accomplish tasks, or they do not assign points at all. By applying such policies, participants are forced to modify their approaches until they succeed (Chung & Cohen, 2014). However, they do not get the appropriate feedback and rewards while making progress towards their target. Moreover, they do not reinforce their positive feelings (Dabrowski et al., 2015). As a result, they can be discouraged and disengaged from the learning process. Furthermore, competitors tend to assess the difficulty of a challenge by the appointed score value. Since there is no partial credit, when they believe that it is difficult for their level of expertise, competitors avoid trying to solve it.

3) Competitions' organizational and functional issues:

a) High demands in resources and preparation time: Competition organization demands a high number of hardware and software resources for the infrastructure. According to the scale of a particular event, a proper place is needed to host the event and weeks of preparations (Mirkovic & Peterson, 2014).
b) Needs for expert support personnel: Arranging a proper environment requires personnel of expertise dedicated to the event's preparation for a long period prior to the competition. Personnel, e.g., administrations and technicians (Hoffman et al., 2005), may also be required to support the event during its operation.
c) High quality assurance standard: Designers of live competitions need to follow strict quality assurance processes to ensure that there are no errors in the contest. Faults in the organizational structure or ambiguities in contest's challenges might interrupt event operation (Chung & Cohen, 2014) and discourage future participation.
d) Events do not take place frequently: To mitigate the costs, organizations tend to set fewer, larger and multi-participant events rather than smaller but more frequent ones (Allen & Straub, 2015).

4.5. Analysis scheme

In this section, the formed comparative evaluation scheme is employed on three live competition approaches from the literature in order to reveal their value with respect to the educational impact.

Issues of live competitions triggered the academic community to propose several approaches that attempted to utilize the merits of live competitions, mitigate the aforementioned problems and fit such events in particular educational contexts. Our study can be used to analyze new approaches and make some assumptions on their feasibility and educational impact. In the remainder of this section we refer to some recent and notable efforts from the literature, mainly aiming at decreasing demands in cost and resources required for organizing live competitions.

Class Capture-the-Flag exercises: Mirkovic and Peterson (Mirkovic & Peterson, 2014) describe the Class Capture-the-Flag exercises (CCtFs) approach. CCtFs are small scale-attack and defense style competitions, in which students alternate between offensive and defensive roles. The exercises are conducted on DeterLab (cyber DEfense Technology Experimental Research Laboratory) platform (Mirkovic & Benzel, 2012), a virtual facility which allows allocation of resources among users for the implementation of cyber security experiments. CCtFs can be repeated frequently throughout a semester and can decrease the organizational demands typically required in the preparation and operation of such events. They require a few weeks of preparation with the involvement of students instead of experts. Usually, CCtFs have a few hours duration, so that they can be arranged during classes and labs. Moreover, they provide the instructors with the option to use a wide range of scenarios that focus on versatile security topics such as cryptography, exploits, denial of service, etc. CCtFs are also facilitated with automated setup and assessment features that tolerate the least involvement of instructors during the events. Each CCtF is followed by a post-mortem analysis that helps students to better assimilate the cyber security concepts they have been taught, as well as to reflect on the strategies employed during the exercises (Mirkovic et al., 2015).

Offline Capture the Flag Virtual Machine: Chothia and Novakovic (Chothia & Novakovic, 2015) presented the Offline Capture-the-Flag Virtual Machine (OCtF VM) framework as part of the formative assessment of a cyber security course. According to their approach, a virtual machine is created and distributed through the web. The virtual machine hosts jeopardy style CtF challenges that students have to solve individually. Students download the virtual machine at the beginning of the semester, and they employ it in their own hardware. The virtual machine has certain services pre-installed and configured, whereas specific settings are configured on its first boot, e.g., unique flags are generated for each student. As students are progressively introduced to miscellaneous cyber security topics of the university's course, they are required to solve challenges in the virtual machine. The challenges include implementing methods for decrypting files, auditing access control mechanisms, analyzing and attacking key exchange protocols, attacking websites, and reverse-engineering programs. They are usually straightforward so that cheating would be more time-consuming than solving the exercises. When a student solves a challenge, s/he acquires a flag that s/he has to submit on a flag submission server. The submission server verifies the token and provides feedback instantly to the student. The results are only acknowledged to the student that made the submission, whereas some specific details, e.g., students and virtual machine identification, are recorded on the server. Students are also required to hand in reports explaining their activities for the solution of the challenges. Reports aid in reflecting on what they have accomplished and providing

information to instructors in order to assess their work. At the end of the sessions, instructors mark the written reports and provide feedback to the students.

Tracer Fire Exercise: Researchers of Sandia National Laboratories (McClain et al., 2015) (Silva et al., 2014) describe the Tracer Fire (Forensic and Incident Response Exercise) training program. Tracer Fire is a classroom based multiday jeopardy style competition that focuses on forensics. Participants are individuals from U.S. government agencies, law enforcement, industry and universities, which work in teams of four to six, as they are required to solve realistic challenges to gain points. Challenges require contestants to use cyber security software tools, to utilize forensic analysis techniques (e.g., review server logs to identify suspicious entries) and to analyze adversary tactics. At the beginning of the event, participants are provided with laptops that have installed basic utility tools and the essential forensics software. Furthermore, participants are allowed to download additional tools and applications and install them on their laptops. The event's infrastructure is based on a specific software architecture that includes a web-based game server and a news server. The game server provides challenges to the participants, receives their answers, and delivers feedback, whereas the news server makes announcements providing information relevant to the scenario of the event.

In this section, we put the live competition approaches presented above on the test of the proposed analysis scheme. More specifically, we resolve them into their elements, identify the problems they tried to solve and appreciate their pedagogical effectiveness.

The results of our test scheme are summarized in *Table 4-2* that consists of two parts. The first part analyzes the elements of the investigated approaches with respect to the proposed concept map characteristics, depicted in *Figure 4-1*. The column 'Characteristics' of *Table 4-2* contains the concept map's segments (described in 4.3) and the *Assessment* and *Reliability* sub-segments. The second part examines the effectiveness of the inspected approaches in confronting the identified problems of live competitions (presented in 1.1). In the remainder of this section, the results of our test scheme are discussed.

Live competitions are characterized by certain limitations that hold back their efficiency when they are integrated into particular educational contexts. The approaches we analyzed aim at decreasing the demands in cost and resources required for the organization of regular and durable security competitions (Chothia & Novakovic, 2015). However, our analysis proves that the problems stated earlier were mitigated by trading other attributes of live competitions. Attribute trading is notable to the OCtF VM approach. In this approach the preparation issues are solved quite effectively by minimizing the demands of preparation, as the contest's infrastructure is encompassed in a virtual machine. The 'duration' and the 'limitation of repetition' issues are tackled, as the exercise can last the whole semester and the presence of an instructor is not required. However, the exercise lacks the 'attack' aspect, whereas the 'contest' factor is downgraded because students do not interact with each other and they do not get feedback on the progress of their classmates. Subsequently, the pedagogical benefits of 'enjoyment' and 'motivation' are downgraded (Chung & Cohen, 2014). The authors in

(Chothia & Novakovic, 2015) rightly claim that the lack of the competition factor is useful for weaker students. Nevertheless, an optimal solution would be to have a setting that contains the 'attack' and 'contest' ingredients with additional features that group or pair the participants according to their background and capabilities. In this a way, everyone has an opportunity to win with the appropriate scaffolding.

On the contrary, our analysis scheme clarifies that the CCtF approach preserves the qualities of live competitions that were downgraded in the OCtF VM approach. However, by contrasting the CCtF and OCtF VM approaches, we can observe that the preparation considerations have only been mitigated in the first case, the scale is set to the size of a class, the instructors' presence is required during the exercises, and ad hoc assessment methods may be used. Therefore, CCtF seems a more balanced approach but in the cost of not addressing drastically the organizational issues.

On the other side, the Tracer Fire exercise differs significantly from the OCtF VM and CCtF approaches. Tracer Fire exercises are organized for intermediate level participants that have background knowledge and experience in cyber security. They focus on the domain of forensics and they incorporate some noteworthy logging and assessment capabilities that constitute them valuable research tools. Tracer Fire relies on an elaborated software framework that facilitates its preparation arrangements by explicitly defining some elements of the infrastructure, like *topology*, *competition server* and 'nodes' (*Figure 4-1*). However, the issues related to the organization of the event remain unsolved, as the event arrangement demands significant resources in terms of time, physical devices, and personnel of expertise. Consequently, the frequency of events is limited to once per year.

The notion of attribute trading that is derived from our scheme is identified not only in the approaches we included in the presented analysis scheme but also in other approaches from the literature. Another fact we observed during our study is that very few works explicitly studied sound learning theories in live competitions (Martini & Choo, 2014), as for example experiential learning in (Chothia & Novakovic, 2015). Moreover, the lack of empirical data in the majority of the studies does not provide the ability to explicitly connect live competition characteristics with particular educational impacts.

***Table 4-2**. Analysis Scheme for Live Competition Approaches*

Characteristics	***CCtF***	***OCtF VM***	***Tracer Fire Exercise***
Contest Form	- Attack mode - Defense mode	Jeopardy mode	Jeopardy mode
Pedagogical benefits	- Exercise KSAs - Enjoyment and motivation - Collaboration	- Exercise KSAs - Enjoyment and motivation are downgraded due to the lack of contest and attack factors	- Exercise KSAs - Enjoyment and motivation are downgraded due to the lack of attack factor

Characteristics	***CCtF***	***OCtF VM***	***Tracer Fire Exercise***
Participants	- Students, novices to intermediates - Organized in teams - Small scale - Low to medium background prerequisite	- Students, mainly novices - Participate as individuals - Unlimited scale - Low background prerequisite	- Intermediates from U.S. government agencies, law enforcement, industry, universities - Organized in teams - Small scale - Low to medium background prerequisite
Infrastructure	- Depends on the exercise's scenario - Employed on virtual settings - Requires physical co-location	- Challenges hosted in virtual machines - Virtual machines can be distributed remotely	- Fulfills a distinctive topology - Employed on physical devices - Requires physical co-location
Preparation	- Arrangements require a few weeks - Requires expert support personnel - Demands pre-installed resources - Allows frequent repetitions - The event lasts a couple of hours	- Arrangements require a little time - Some expertise is needed to prepare for the challenges in the virtual machine - No resources are required as students bring their own hardware - Possibilities for unlimited duration and repetition	- Demands high preparation in all terms that limits the potential in the frequency of repetitions
'Policies & mechanisms' and 'Reliability'	- Depend on each exercise's scenario	- Identify participants - Different flags per challenge - Operate safely outside universities' network	- Depend on exercises' scenarios - Advanced logging capabilities
Assessment	- Custom scoring mechanisms that depend on each exercise's scenario	- Based on flag submissions and students' write-ups - Do not require instructors	- Provided by efficient scoring mechanisms
Issues	***CCtF***	***OCtF VM***	***Tracer Fire Exercise***

Characteristics	*CCtF*	*OCtF VM*	*Tracer Fire Exercise*
Contests aim to measure skills	Solved extrinsically: exercises supported by feedback and classes	Solved extrinsically: exercises supported by feedback and classes	Not solved
Fail to address the management of settings realistically	Not solved	Not solved	Not solved
Diversity of topics is not supported	Solved	Not solved	Not applicable as the exercises focus on forensics
Not calibrated to participants' needs	Solved	Solved	Solved
Not an experiment environment	Mitigated indirectly since events can be repeated frequently	Solved	Not solved
Partial credit is not supported	Depends on exercises' scenarios and custom assessment mechanism	Mitigated extrinsically through students' reports	Supported intrinsically due to efficient assessment mechanisms
High demands in resources and preparation time	Solved	Solved	Not solved
Needs for expert support personnel	Not solved	Mitigated	Not solved
High-quality assurance standard	Mitigated	Mitigated	Not solved
Events do not happen frequently	Solved	Solved	Not solved

4.6. Chapter conclusion

The availability of proper means to mitigate the problems associated with live competitions could lead to essential improvement of particular educational impacts. Organizers need to reduce the demands and the logical complexity of live competitions. They also need to consider the proper learning theory their approach will embrace and utilize it to guide the entire competition design process. More specifically, they need to provide the means to create and parameterize the educational environment in order to set the conditions for an effective learning process. In this way, the

assessment and feedback factors will be better facilitated. Moreover, live competitions need to adapt to the learners' knowledge, capabilities and expectations and to provide scaffolding facilities to contestants. Hence, we argue that the unstructured nature of current approaches does not help to better facilitate these features and effectively deal with their challenges without trading other significant attributes. The comparative analysis elaborated in this work provides a proof of this concept, as well as indicative directions for its utilization in the phases of analysis, feasibility study, and assessment for the development of successful live competition approaches.

The presented analysis scheme can be put into effect in the development of new live competition approaches and the produced deductions can be used in the development of new pedagogical methodologies relative to the concept of live competitions, e.g., gamification and game-based learning.

5. THE COFELET FRAMEWORK

5.1. Introduction

In this chapter, COFELET, a conceptual framework proposed for the design and implementation of cyber security serious games is presented (*Figure 5-1*) (Katsantonis et al., 2019). COFELET is a multidisciplinary framework embracing several features for the creation of effective cyber security game-based approaches appropriate response to the challenges of the cyber security field. COFELET realizes cyber security education as an attractive and open subject for a broad spectrum of people, including young individuals and women. For this reason, it encompasses the game-based learning perspective, and it draws elements from live competitions (e.g., capture the flag or CtF competitions) and cyber security modeling techniques. Moreover, COFELET envisages approaches that rely on sound learning theories and innovative teaching methods that advocate the effectiveness of COFELET compliant approaches (COFELET approaches). On this ground, COFELET complies with the activity theory to analyze and the interactions of the learners with the games; it assumes the layer learning approach (Katsantonis et al., 2019) (Greitzer et al., 2007) to apply cognitive principles and to enhance the learning process; and it uses the continuous learning paradigm (Sessa and London, 2015) to engage learners in a cycle of learning, updating and reinforcing knowledge. Besides, COFELET assimilates well known cyber security models and methodologies (e.g., MITRE CAPEC's attack patterns (MITRE, 2020) and Cyber Kill Chain model or CKC (Lockheed Martin, 2020)), generally used in cyber security threat analysis and modeling, to form highly organized and parameterized learning and training environments.

In the remainder of this section the COFELET's key concepts are presented along with a brief description of the proposed COFELET methodology. Then, the manner that the cyber security standards (e.g., CAPEC, CKC, NCWF) are adopted in the context of COFELET compliant games to form highly structured and organized learning and training environments, is presented. Moreover, the assessment and the adaptability features of the COFELET based approaches are explained.

5.2. Issues & Challenges of Cyber Security Education

Cyber security educational programs usually apply curricula and deliver learning and training programs specifically designed to satisfy the demands of cyber security personnel. However, cyber security educational programs fail to make cyber security a more appealing and accessible subject to a wide range of people. For example, young people are not attracted to cyber security at an early stage, neglecting the opportunities to develop foundational skills and knowledge at an early age. Moreover, women make up only a small percentage of the overall cyber security workforce, due to current perceptions and beliefs (Haney & Lutters, 2017).

Cyber security educational programs also fail to deliver effective services that will supply mission ready experts. The current programs usually apply traditional teaching methods (e.g., lectures, workshops, lab sessions) and they often utilize live competitions (e.g., capture the flag) in which participants compete on their knowledge and skills. Despite their pedagogical benefits when incorporated in educational contexts, live competitions are also associated with many obstacles (analyzed in section *4.4 Identified Problems and Issues*) that can limit their pedagogical value and effectiveness. Nevertheless, the frequency of live competitions is limited, and cyber security educational programs do not provide opportunities for '*continuous*' and '*always-on*' accessible learning and training (Allen & Straub, 2015). Continuous learning and always-on training are significant features of cyber security education, as cyber security personnel need to frequently train, exercise and reinforce the critical knowledge and skills to remain mission ready. Furthermore, cyber security training approaches require setting the opportunities and providing motives to trainees to increase their speed and effectiveness when confronting a problematic case. A training approach that implements a continuous lifecycle of learning, updating, and reinforcement can foster trainees' skills to analyze a problematic case and envisage a solution; perform rapidly the appropriate sequence of actions in incident response cases; and enter instantly the appropriate commands and options to carry out certain activities (Allen & Straub, 2015).

Additionally, current cyber security education approaches do not utilize sound theories, standards, and methodologies and especially they do not advocate underlying learning theories to envisage innovative educational strategies (Martini & Choo, 2014). They fail to quickly adapt to a continually and rapidly changing cyber world and provide on-demand solutions to confront a landscape of new threats by fostering up-to-date knowledge, skills, and experiences to the cyber security personnel (Amorim et al., 2013). Moreover, assessment schemes need to consider proficiency rather than just performance-based assessment. Proficiency relates to a high degree of acquired skills and expertise, while performance measures the acquired skills under conditions (e.g., a trainee applies skills in a simulator for a fixed period of time) (Allen & Straub, 2015). On the contrary, proficiency-based assessment promotes the evaluation of the acquired skills under different conditions and in various contexts.

5.3. Key Concepts

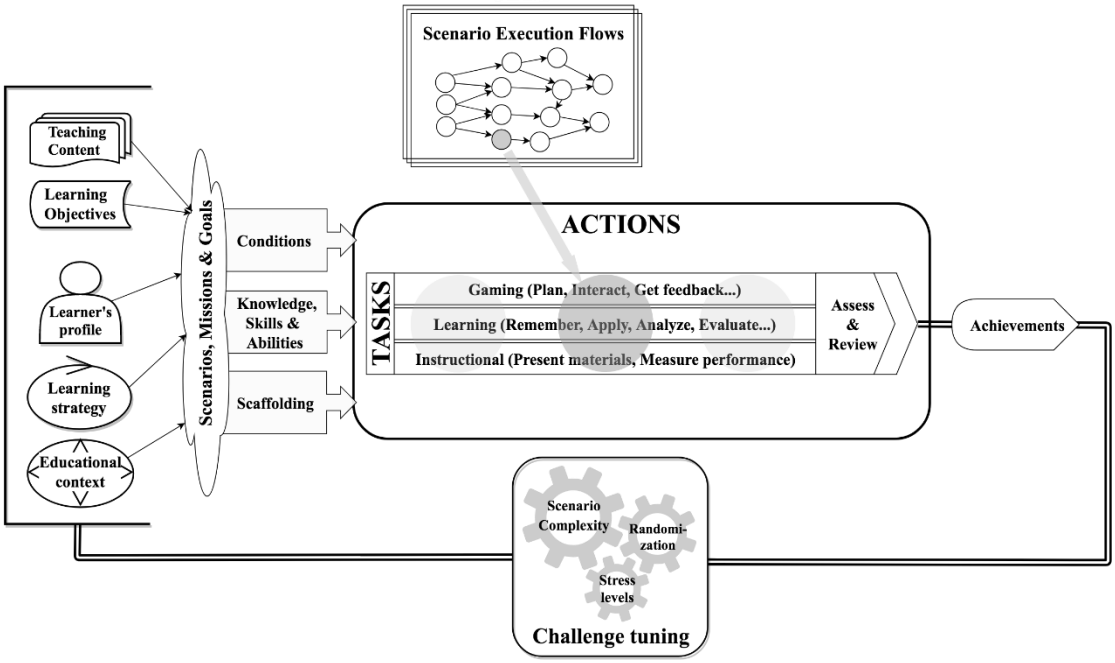


Figure 5-1. The COFELET Framework

The primary element in the COFELET framework is called *task* (represented in *Figure 5-1* by circles). A task represents actions performed in the game by learners or non-playable characters (e.g., mentors, teammates, adversaries) directed at the fulfillment of the game's goals (e.g., the unleash of a cyber-attack).

- *Goals* are problems the learner has to solve in the context of the game.
- *Conditions* in the COFELET framework represent prerequisites needed to perform tasks, while the activity theory regard the operations of actions (presented in 2.3.5).
- *Scenario execution flows* (SEFs) describe the sequences in which tasks have to be performed, as well as their interdependencies and relations. Each task can be associated with one or more SEFs. SEFs are proposed to be defined in analogy to attack patterns that describe the sequence of actions of attackers, as generic representations of cyber-attacks.
- *Educational context* represents the characteristics of the environment a serious game operates.
- *Gaming context* describes the state of the game including the majority of game properties (e.g., properties regarding scores, durations, coordinates of avatars) and the existing conditions.
- *Scenario* contains the appropriate information for the setup of a game session such as game narratives (i.e., the game's story) and a description of the gaming context, including the provided conditions, along with the goals that learners have to accomplish. Scenarios can involve one or more stages, called *steps* that require the accomplishment of one or more goals.

- *Knowledge, skills and abilities* (KSAs) designate the knowledge and the competencies learners have to utilize to execute tasks and apply SEFs.
- *Learning objectives* (LOs) are brief statements describing the KSAs that students are expected to gain by the end of a game session. A session is the period that a learner will spend on a game trying to fulfill one or more missions. The teaching content contains specific materials (e.g., text, videos) aiding the learner to assimilate the new knowledge.
- *Learning strategy* is the plan followed in a session that aids learners achieve the learning objectives. The learning strategy explicitly defines the learning theory it embraces (e.g., cognitive theories).
- *Scenario complexity* refers to the number of tasks that the learner has to perform to fulfill missions. *Stress levels* denotes how much pressure the games put on learners. Depending on the games' properties, stress levels can refer to the amount of time provided for the execution of missions or the policy used to manage the scoring facet of the games (e.g., presence of negative points).

5.4. Brief Description

The COFELET framework (*Figure 5-1*) specifies the main elements that have to be taken into consideration for the design and development of effective cyber security serious games, together with the interconnection of these elements in the structure of the game. In COFELET framework compliant games (COFELET games), learners perform tasks. According to conditions, learners have to perform the proper sequence of tasks to successfully apply one or more SEFs and fulfill the game's goals. Learners envisage the appropriate SEFs and perform the corresponding tasks when they utilize the appropriate KSAs. For example, a learner can decode the encoded text "*Q0VGRUxFVA==*" (i.e., the word "COFELET" base64 encoded), if s/he recognizes and comprehends the coding schemes and she knows how to use decoding tools (e.g., the base64 tool in Linux). Furtherly, the COFELET framework contains associations of KSAs (and the related tasks) with LOs and teaching contents. In the previous case, the KSAs are associated with the learning objective: "*the learner identifies the common encoding techniques such as XOR, ASCII, Unicode, base64 etc.*"; and with the teaching content that presents the encoding techniques and provides examples of encoded texts.

COFELET games use SEFs to foresee the subsequent tasks leading to the fulfillment of goals (solution). Subsequently, they contrast the tasks performed by learners with the solutions and record the results to dynamically assess the learners' performance. Moreover, COFELET games support learners' efforts, as they include a hint system that gradually reveals parts of the solution and provides scaffolding capabilities. The hint system can be triggered in cases that the learner does not perform the proper task after several tries; or when s/he asks for help; or when she spends much time on a task or a mission (provided that the game counts time).

At the end of a session, the efforts of the learner are reviewed, and feedback is provided (i.e., achievements). Subsequently, the learner's profile is updated, and the challenging levels of the subsequent missions are tuned to the optimal point by altering the game's stress levels and the scenario complexity. Finally, the subsequent scenario is selected for a successive session. The scenarios' selection depends on the learner's profile, the LOs along with the teaching content, the learning strategy, and the educational context. The COFELET framework also envisages a certain degree of randomization to improve the replay ability of games and to keep learners motivated.

5.5. Learning Strategies

The COFELET framework has been established on the principles of the activity theory (Jonassen & Rohrer-Murphy, 1999), through its conformity with the ATMSG model. Activity theory is a social constructivism theory used to analyze the components of interactive, composite and dynamic learning environments (e.g., the COFELET games) as well as the learners' activities in such environments. Besides, the learning process in a social constructivist learning environment is efficient as learners are encouraged to perform meaningful and realistic activities and to interact with the environment to solve problems (Vygotsky, 1978). Therefore, the number of passive activities such as reading, hearing and watching, is reduced and learners are not yet passive receivers of information as happens with traditional teaching methods (e.g., lectures, workshops, lab sessions) (Dewey, 1933) used in many cyber security education programs (Allen & Straub, 2015). In addition, learners use prior knowledge to experiment with the system and make assumptions and errors that are not traumatic but pedagogically productive (Ausubel, 2000).

The COFELET framework adopts a *layered learning* approach (Greitzer et al., 2007), under a perspective that explicitly maps its layers to the Bloom's taxonomy levels (*Figure 5-2*). More specifically, the lower layer(s) map to the *Remember* and *Understand* levels of Bloom's taxonomy in which COFELET requires trainees to perform simple tasks of identifying, comprehending and recalling concepts, structures, facts and practices. The mid-layer(s) maps to the *Application* level of Bloom's taxonomy, in which trainees will have to carry out simple missions by applying the knowledge of the first levels. The higher layer(s) map to the higher levels of Bloom's taxonomy, in which learners will face genuine problems that they will have to solve. In these levels, learners should be able to exhibit deep knowledge and broad skills and to think outside of the box by adapting the tools and strategies to unleash APTs in a constantly changing game environment. In the highest levels, COFELET aims to foster problem-solving abilities, analytical and creative skills and critical thinking.

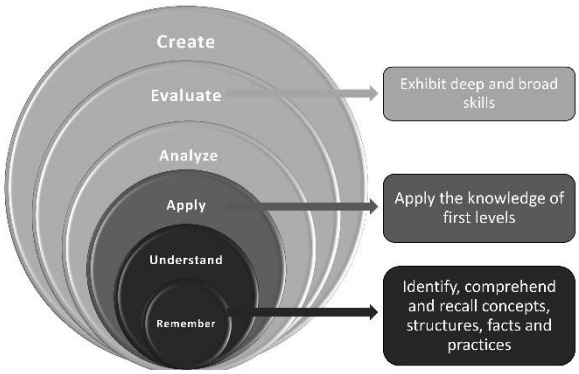


***Figure 5-2**. COFELET layers mapped to the revised model of Bloom's taxonomy (Katsantonis et al., 2019)*

The utilization of the *layered learning* approach features an effective repertoire of learning strategies. In particular, the COFELET framework assumes modern educational methodologies that comply with modern learning theories and traditional learning and training paradigms. The modern educational methodologies foster critical thinking, problem-solving abilities, and analytical and creative skills by forming realistic environments in which learners solve genuine problems. On the contrary, the COFELET framework also considers traditional learning and training in which learners perform simple activities such as comprehension and recalling of concepts, utilization of tools and practice on tasks. Traditional learning and training paradigms are important in cases that the objectives of the learning session include the update and reinforcement of critical cyber security knowledge and competencies. For example, for a cyber security professional working in an incident response team, it is important to immediately recall knowledge such as the port to protocol mappings or the utilization of proper tools to dump a computer's memory.

Nevertheless, the COFELET framework adopts the *continuous learning* approach (Sessa & London, 2015) under two perspectives: a) a learner has to try new experiences and challenges; b) a learner has to try known things in new ways. The COFELET framework supports the first perspective by forming scenarios in which the environment becomes increasingly immersive and complex, and the learners have to confront new problematic cases tuned to their needs and cognitive levels. On the other hand, under the viewpoint of the second perspective, the COFELET framework defines different contexts and conditions when the learner has to retry activities that update or reinforce knowledge and capabilities already possessed.

5.6. Conformity with the ATMSG model

The COFELET framework complies with the ATMSG model (Carvalho et al., 2015), an extension of the LM-GM model (Arnab et al., 2015), to facilitate the fusion of the learning aspect in serious games. The adoption of the ATMSG model in COFELET facilitates the systematic analysis, and organization of the games' components and the identification and classification of the actions and

activities (i.e., a series of actions) that occur in the COFELET game. The identified activities are classified under the gaming, the learning, and the instructional perspectives (the game perspectives). The gaming activities describe the learner's actions assuming the role of a gamer. For example, in a cyber security serious game, such actions are the unleashing of an attack and the acquisition of a flag. The learning activities refer to the actions a player performs assuming the role of a learner. Such actions in a cyber security serious game are the utilization of information (e.g., recall the sequence of actions and stages to unleash a cyber-attack), the utilization of cyber security tools, and the application of critical thinking to evaluate conditions and plan solutions (e.g., assess the applicability of tools and methods according to the game's context). Instructional activities refer to the actions carried out by the game aiming at providing scaffolding and feedback to support learners to achieve their learning objectives and reflect on their accomplishments. In particular, instructional actions refer to the providence of hints and the presentation of teaching contents related to the gaming and learning objectives of the game; the assessment of learner's efforts; and the presentation of achievements and scores or grades (i.e., feedback). To ensure that the HackLearn game operates under the game perspectives, the game designers can initially oppose questions such as "what are the activities of the learner and what do these activities teach her?", "how does the game aid the learner in achieving the gaming goals and the learning objectives?", "how the monitoring and the assessment of learner's efforts is facilitated?". Subsequently, the game designers employ the ATMSG approach to analyze the activities, the actions, and the components of the HackLearn game under the gaming, the learning and the instructional perspectives.

5.7. Conformity with Cyber Security Standards

COFELET envisages the utilization of standard methodologies, models and strategies that are generally used in threat analysis and modeling approaches such as the Mitre's CAPEC (CAPEC, 2021), the Lockheed Martin's Cyber Kill Chain (Martin, 2014), and the National Cybersecurity Workforce Framework (NCWF) of the National Initiative for Cybersecurity Education of National Institute of Science and Technology (NIST) (NCWF, 2021). CAPEC can be used as the main reference for the construction of COFELET primary concepts such as tasks, goals, conditions and SEFs. CKC can be used as a guide for the definition of complex missions such as the unleashing of APTs. NCWF can form a basis for the definition of games' LOs and learners' roles (e.g., forensics analyst, vulnerability assessment analyst).

In the remainder of the section, the conformity on cyber security is justified by presenting the assimilation of attack patterns and cyber security workforce roles in the context of the COFELET framework.

5.7.1. CAPEC's Attack Patterns and SEFs

The COFELET framework embraces CAPEC's APs and uses them as the main reference for the realization of SEFs. *Figure 5-3* depicts the realization of CAPEC's '*TCP SYN Scan*' AP (also known as '*half-open*') as a SEF. The '*TCP SYN Scan*' is a common type of port scanning technique aiming at determining the status of a remote target's ports or the existence of a firewall mechanism. It involves the formation of a TCP connection request starting with the transmission of an empty TCP packet with the SYN flag set (SYN packet) to the target. The target responds with a SYN/ACK packet indicating that there is a service running in the requested port; hence the port is *open*. If there is no service running on the requested port (i.e., the port is *closed*), the target responds with a RST packet. If no response is solicited or an ICMP packet of type 3 (*Destination unreachable*) is received, then the port is thought to be protected by a firewall mechanism and marked as *filtered*.

The tasks of the *TCP SYN Scan* SEF belong to the reconnaissance phase of the CKC model. *Figure 5-3* represents learners' tasks as rectangles, the SEF's interim tasks as polygons and the SEF's completion tasks as circles. The learner's tasks are performed manually by a learner or a non-playable character, whereas the interim tasks are automatically performed by the game's mechanics (e.g., tools, game's operating system, card dealers or card holders). The learner's tasks are performed when, for example, a learner enters a command in a game's terminal or a learner interacts with the game's graphical user interface (e.g., clicking on buttons and links, entering data in text boxes or forms) or a learner plays a card. On the other hand, the SEF's completion tasks denote the end of the SEF execution by providing feedback and by triggering game events. The *TCP SYN Scan* SEF has the gaming goal of discovering the status of target's ports and it involves the following conditions:

- C1: Learner knows target's IP address or target's domain name to trigger the scan.
- C2: A network scanning tool, such as Nmap or netcat, is available for the learner.
- C3: Learner has administrator rights (in the gaming context) to access raw sockets.
- C4: Learner's host belongs to a network that routes packets to the target.
- C5: If a firewall mechanism exists, it accepts TCP packets with the SYN, ACK, RST flags set.
- C6: There is a service running on the requested port.

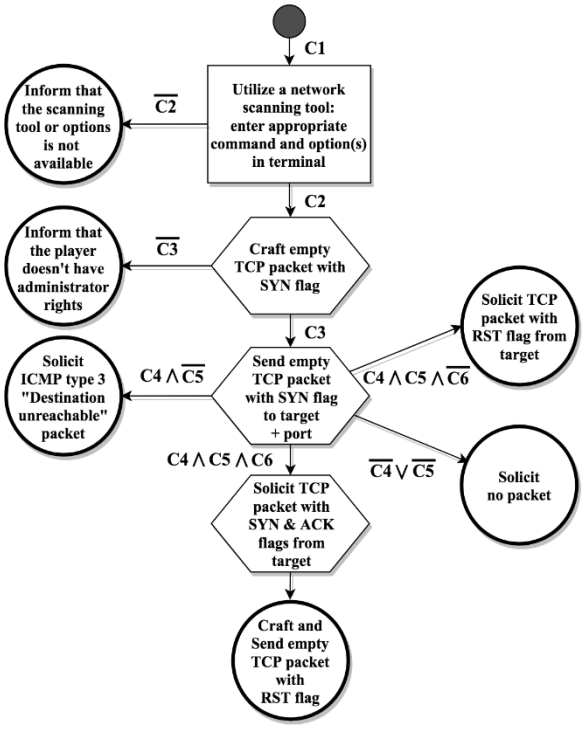


***Figure 5-3**. The TCP SYN Scan SEF*

Each one of the above conditions is associated with one or more tasks in the *TCP SYN Scan* SEF. For example, the task "*Utilize a network scanning tool*" is doable when the "*A network scanning tool is available*" condition is present, whereas the "*Solicit TCP packet with SYN & ACK flags from target*" is doable when "*If a firewall mechanism exists, it accepts TCP packets with flags SYN, ACK, RST*" is met. Consequently, during a game session and according to the existing game's conditions, the execution flow will result in the disclosure of the corresponding target's port status. For example, when the "*Solicit TCP packet with SYN & ACK flags from target*" task is executed, the detail that the port is *open* is disclosed to the learner.

5.7.2. National Cybersecurity Workforce Framework

In COFELET compliant games, the NCWF forms a basis for the organization of the teaching content and the definition of suitable LOs. For example, according to the NCWF the aforementioned port scanning technique is related to the following skills:

- The skill S0081: Using network analysis tools to identify vulnerabilities (e.g., fuzzing, Nmap, etc.).
- The skill S0191: Assessing the applicability of available analytical tools to various situations.

According to the NCWF, the *S0081* skill is required for a '*Vulnerability Assessment Analyst*', whereas the *S0191* skill is required for a '*Target Network Analyst*'. The '*Vulnerability Assessment Analyst*' role is related to the vulnerability assessment of host systems and networks to identify

misconfigurations and policy flaws. On the other hand, the '*Target Network Analyst*' role is related to the collection of information, analysis, and profiling of targets. For each cyber security workforce role, the NCWF defines a list of tasks and KSAs required to perform these tasks. For example, apart from the *S0081* skill a '*Vulnerability Assessment Analyst*' has to master 50 more KSAs including:

- The knowledge K0177: Knowledge of the cyber-attack stages (e.g., reconnaissance, scanning, enumeration, gaining access, escalation of privileges, maintaining access, network exploitation, covering tracks).
- The skill S0052: The usage of social engineering techniques (e.g., phishing, baiting, tailgating, etc.).
- The ability A0001: Identify systemic security issues based on the analysis of vulnerability and configuration data.

Consequently, when the COFELET considers that a '*Vulnerability Assessment Analyst*' trainee has achieved the LOs related to the *S0081* skill, it can select LOs and teaching content related to the aforementioned KSAs. In such a way, the COFELET framework foresees games that group teaching content and LOs with respect to the NCWF's workforce roles. The workforce roles are specified in the learner's profiles by either an instructor or the learner and they are essential part of the selection policy of subsequent scenarios and missions.

5.7.3. Cyber Kill Chain

The COFELET framework envisages composite scenarios by taking into account the Lockheed Martin's Cyber Kill Chain (Martin, 2014) methodology. The phases of CKC are considered missions that have to be accomplished by putting into effect the appropriate SEFs. In the remainder of the section an example scenario is presented that implements CKC as a sequence of SEFs realizing CAPEC's attack patterns:

1) Reconnaissance: learner performs SEFs executing host discovery, port scanning and fingerprinting techniques such as CAPEC's '*Scanning for Vulnerable Software*' AP.
2) Weaponization: learner puts into effect CAPEC's '*File Content Injection*' AP to create a weaponized file. In game terms, the weaponized file can be processed and created by utilizing an in-game tool such as a gamified version of the Metasploit penetration testing platform or the msfvenom tool.
3) Delivery: the learner uses APs aiming at delivering the weaponized file such as the '*USB Memory Attack*' and the '*Counterfeit Websites*'.
4) Exploitation and Installation: the '*Local Execution of Code*' CAPEC APs can be put into effect to simulate the exploitation and installation of malware to the target.
5) Command and control: a command-and-control SEF is applied to simulate the establishment of a channel to the master server (i.e., adversary's server), based on the CAPEC's '*Communication Channel Manipulation*' AP.

6) Actions on objectives: according to the game's scenario several of the CAPEC's APs can be employed (e.g., '*Disable Security Software*', '*Privilege Escalation*').

5.8. Adaptability and Scaffolding

The COFELET framework emphasizes the adaptability and scaffolding capabilities of cyber security serious games. Initially, the COFELET framework analyzes the adaptability characteristics of the participants' learning profile and the attributes of the educational context. The educational context is inspected before the learners start the game and its considerations can include the available time and budget, the accessible tools and resources, the characteristics of the game's environment, such as the delivery platform and the presence of an instructor, and a combination of historical, political and economic factors (De Freitas & Oliver, 2006). The learner's profile characteristics are analyzed before the learner starts a game session. The proposed learner's profile characteristics include the learner's learning history, the learner's characteristics (e.g., age, retention ability), the learning preferences, (i.e., background knowledge and interests) and the learners' motives such as the notions of advancement and the immersion facet.

Subsequently, COFELET considers the learning strategy, the LOs and the teaching content. In this phase, COFELET chooses the game's scenario, and it tunes the scenario complexity. The scenario complexity determines the number of tasks required to complete a mission as well as the number of conditions the learner has to take into account to envisage and apply a strategy. Then COFELET adjusts the game's adaptive elements such as the game's interface, the scoring scheme, the reward system, the tasks' durations, the game resources provided to the learners (e.g., resources learners have to consume to perform tasks), the stress levels, etc.

During a game session, COFELET records and assesses the learner's tasks and supports his/her efforts through a hint system. The hint system presents successively a number of hints, when the game notes that a learner spends too much time on a task (e.g., above a threshold adjusted for the specific task and learner (Johnson et al., 2015)) or s/he asks for help. For example, the initial hint presented to the learner can point his/her attention to the right direction (e.g., a specific service of the target host), whereas a subsequent hint can explain and present the execution of the following tasks that have to be performed.

At the end of a session, COFELET assesses and reviews the learner's performance and presents the corresponding feedback to the learner to make him/her reflect on his/her achievements. The framework also re-adjusts the challenging level, the complexity of the subsequent mission and it tunes the adaptive elements. In cases that a learner has to repeat a game's session to improve his/her performance or to reinforce his/her knowledge, challenge re-adjusting is not required. However, as the learner is expected to have low-motivation, a certain degree of randomization can be applied to alter the scenario (e.g., the scenario's narrative) and conditions (e.g., the available tools or tools' options) that will keep the learner motivated to achieve the LOs. Finally, the adaptability

characteristics of the participants' learning profile are updated and a new game session can be initiated.

5.9. Assessment

The COFELET framework dynamically measures learners' performance during the gameplay and at the end of each session, it logs his/her tasks, reviews his/her progress, and provides feedback to the learners. COFELET determines whether learners have executed specific tasks and SEFs, associated with the session's LOs. For this reason, COFELET foresees the definition of clear-purpose and measurable LOs, connected to the gameplay through their association with tasks. Additionally, it distinguishes the LOs in terms of proficiency and performance. Performance LOs measure capabilities under particular conditions while proficiency LOs relate to a high degree of skills and expertise. To this end, COFELET aims at assessing the achievement of specific LOs in different gaming contexts. Moreover, COFELET tracks learner efforts to measure the speed and accuracy of carrying out the tasks required towards the solution of a challenge, as well as the time taken to deal with the problematic situation. At the end of each session, COFELET rewards learners with good practices, whereas it also triggers disciplinary actions for the learners that repeat the same mistakes even after training and reinforcement (Nagarajan et al., 2012). Finally, COFELET can further assess learner progress with the use of tests and questionnaires in pro-game sessions and post-game sessions to avoid distractions.

5.10. Chapter conclusion:

In this chapter, the COFELET framework was presented. The COFELET framework illustrates the main elements that have to be taken into consideration for the design and development of effective cyber security serious games, and the interconnection of these elements in the structure of the games. The proposed framework envisages cyber security game-based approaches established on effective serious games models (e.g., the ATMSG model) and on modern learning theories (e.g., the activity theory, the layer learning approach) along with cyber security standards and methodologies used in cyber security threat analysis and modeling (e.g., CAPEC, Cyber Kill Chain). We presented how these standards can be adopted in the context of COFELET compliant games to form highly structured and organized learning and training environments. In such environments, the game mechanics can supervise learner's efforts and provide advanced adaptability, scaffolding and assessment features. Furthermore, the conformity to cyber security standards and methodologies verifies the validity and sustainability of the COFELET approaches.

6. COFELET ONTOLOGY

6.1. Introduction

One of the main challenges of the presented study is how the COFELET games can seamlessly integrate the cyber security standards described in section 2.4. To this end, the COFELET ontology is proposed aiming at providing a foundation for the development of a universal knowledge base for modeling such environments. The COFELET ontology provides an analytical description of the key elements of COFELET's compliant serious games along with the appropriate classes and their properties. These elements include the cyber security domain elements that model the actions attackers perform to unleash cyber security attacks (i.e., the tasks) and the strategies they employ to achieve their malicious objectives (e.g., CAPEC's attack patterns, the CKC model). The cyber security domain elements are associated with the educational elements (e.g., hints, utilized knowledge, exercised skills) that provide the means to in-fuse the didactics in the COFELET compliant approaches. The COFELET ontology aims at constituting a universal knowledge model for cyber security e-learning and training.

In the remainder of this chapter, the COFELET ontology is presented along with the methodology and the approach we employed to develop it. Subsequently, an illustrative set of COFELET instances is presented. Due to the complexity of the subject, the COFELET ontology is presented at different levels that evolve in complexity and detail.

6.2. Methodology

For the development of the COFELET ontology, which is compliant with the 'Ontology Development 101' guide (Noy & McGuinness, 2001), we employed a middle-out process. According to the middle-out process we defined a set of middle-level concepts that we generalized and specialized to produce a set of high-level concepts and low-level concepts. In the first stage, we leveraged CAPEC's attack patterns, and we identified the operationalizable APs that could be modeled as COFELET game's primary elements. Subsequently, we defined scenario execution flow (SEF) elements as realizations of CAPEC's APs. SEFs are composite elements that constitute generic representations of attacks describing the sequences of tasks attackers perform to unleash an attack along with the relevant information (e.g., prerequisites). Thus, we specified SEFs in terms of COFELET's primary elements (i.e., tasks, goals, and conditions) and we generalized these elements to specify the primary element class. We also associated SEFs with the knowledge, skills, abilities (KSAs) and attitudes the learners have to utilize to apply them; and with the hints that can be presented to the learners to scaffold their efforts towards the achievement of the games' goals. In the subsequent stage of the COFELET ontology process, we specialized the defined elements to form the COFELET ontology.

6.3. The Domain and Scope

The scope of the COFELET ontology is highly organized and parameterized cyber security learning environments (such as the COFELET games). Specifically, we envisaged a COFELET game aiming at teaching cyber security fundamentals, methods and techniques (domain) to professionals working at law enforcement agencies, organizations and companies. According to the NCWF, cyber security professions are assigned to job profiles corresponding to the roles, the tasks and the KSAs defined in the NCWF framework (presented in subsection 2.4.3). Due to the numerous roles defined in the NCWF framework, we limited the scope of the presented ontology to a COFELET game focusing on the training of vulnerability assessment analysts and target network analysts. Additionally, due to the numerous KSAs, the NCWF framework assigns to the vulnerability assessment analyst and the target network analyst workforce roles, only a set of KSAs were utilized for the definition of the learning objectives. These KSAs refer to the networks' operation (e.g., protocols, addressing), the stages of cyber-attacks and the utilization of cyber security tools (e.g., network analysis tools).

6.4. Primary Elements

Primary Elements are represented by objects (the Primary Objects) denoting that an agent acts on an entity or an entity has a property. Specifically, the primary objects are interpreted as statements of the form <subject, verb, object> or <entity, property, property_value> that are called triples (e.g., <Player, provides, host scanner discovery command>). The expression of such statements as triples is widely used in various frameworks and methodologies such as the Resource Description Framework (RDF) (Fallon and Brown, 2016) and the ADL's Training & Learning Architecture (TLA) (Poltrack, 2014). For brevity and simplicity, we adapted an extension of this approach based on quintuple statements in the form of <entity, property, property_value, source, destination> that can be effortlessly translated to the corresponding triples. For example, the quintuple <Port scanner, sends, ICMP type 8 packets, from player host, to destination network> can be transformed to the triples <Port scanner, sends, ICMP type 8 packets>, <ICMP type 8 packet, has source, player host>, <ICMP type 8 packets, directed to, destination network>.

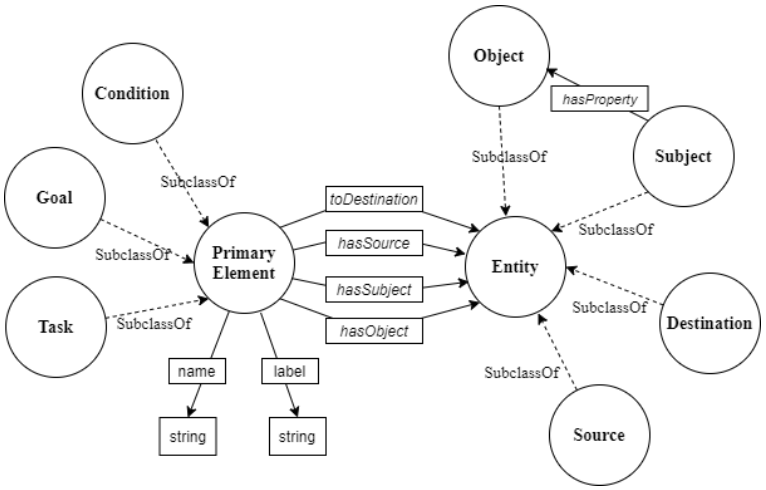


***Figure 6-1**. COFELET Primary Elements*

The Primary Objects instantiate the *Task*, the *Condition* and the *Goal* classes (primary classes) that are subclasses of the *PrimaryElement* class (*Figure 6-1*). The instances of the Goal class represent the gaming goals, the instances of the *Condition* class represent the prerequisites needed to make the tasks doable and

The primary classes inherit the *hasSubject*, the *hasObject*, the *hasProperty*, the *hasSource* and the *toDestination* object properties and the *name* and *label* data properties. The ranges of these properties are the subclasses of the *Entity* class that are the *Subject*, the *Object*, the *Source* and the *Destination* classes representing in-game entities such as players, hosts and networks. The *Subject* and *Object* classes are related by the *hasProperty* object property. The *hasProperty* is an object property with several sub-properties that represent the actions subject entities perform on object entities. These actions represent the tasks performed by agents (learners and non-playable characters such as mentors, teammates, adversaries) directed at the unleash of cyber-attacks such as entering commands, connecting to hosts, searching information and routing packets. The actions also represent the tasks performed by non-agent subjects (e.g., a tool that crafts and sends a packet, a firewall that drops a packet). Finally, the name and the label data properties represent the name and the human readable name of a primary object.

PrimaryElement objects are expressed in first-order predicate logic as follows:

$$\forall\, pe{:}\, PE \rightarrow\ \exists\, subj, obj, src, dst{:}\, Entity\ \wedge\ \mathrm{hasSubject}(\mathrm{pe}, \mathrm{subj}) \wedge \mathrm{hasObject}(\mathrm{pe}, \mathrm{obj}) \wedge hasSource(pe, src) \wedge toDestination(pe, dst) \wedge \exists hasProperty(subj, obj) \quad (1)$$

Moreover, a *PrimaryElement* object can be an instance of either of its subclasses (i.e., *Task* class, *Condition* class, *Goal* class), whereas an object of the aforementioned classes cannot be an instance of more than one of these three classes (i.e., class disjointness):

$$\forall x(PE(x) \rightarrow Task(x)\ \vee\ Condition(x)\ \vee Goal(x) \quad (2)$$

$$\forall x(Task(x) \wedge Condition(x) \wedge Goal(x) \rightarrow \perp) \quad (3)$$

6.5. Entities and Properties

Entities in the COFELET ontology are represented by the *Entity* class. The *Entity* class is a top-level class with several middle-level classes (*Figure 6-2*) that represent the most important concepts in the context of a COFELET game. For example, the middle-level *Tool* class is important as it represents the in-game tools used by the learners. The middle-level classes have numerous subclasses that represent various types and attributes. The *Tool* class is furtherly subclassed to represent various types of in-game tools along with the tools' roles and characteristics. For example, a *Tool* object representing the *ls* Linux tool (i.e., list directory contents) is an instance of *DirNavigator* class, whereas the *msfvenom* Linux tool is an instance of *PayloadGenerator* class. The *Command* class represents the commands entered by learners to perform tasks (e.g., the command 'nmap -sS 192.168.1.0'). The *Facade* subclass stands for the different roles that the system plays (e.g., the instructor that manages the hints, the narrator that presents the narrations), whereas the *Agent* class represents the roles of the game's characters (e.g., the learner, non-playable characters, host users).

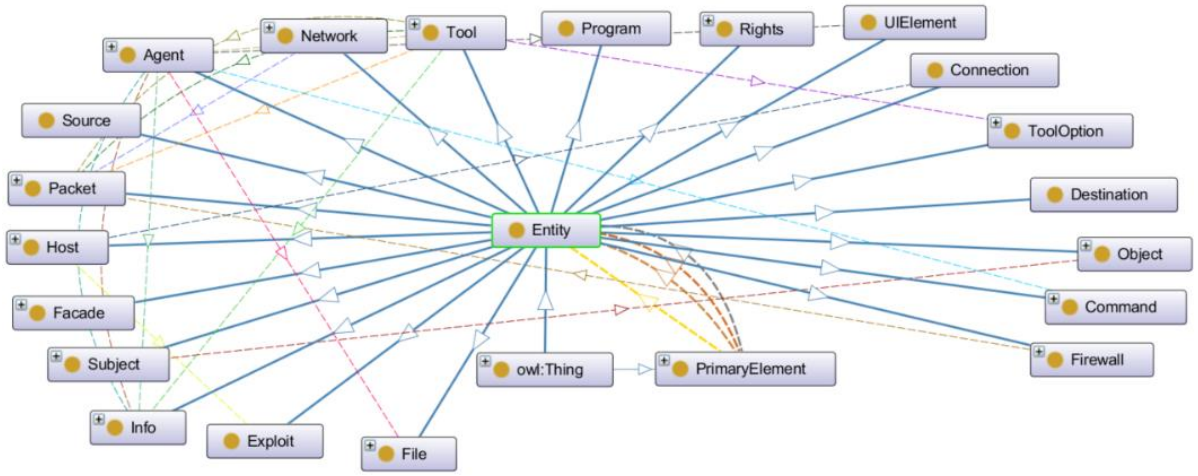


***Figure 6-2**. COFELET middle-level Entity classes (adopted from Protégé ontology editor tool (Protégé, 2021))*

Apart from the higher-level properties presented in the Primary Elements subsection (e.g., *hasSubject*, *hasObject*), the COFELET ontology contains several properties defining the relations of *Subjects* to *Objects* and describing various actions occurring in the COFELET games. These properties are sub-properties of the *hasProperty* property a set of which is listed in Table 1 along with the corresponding Domains and Ranges and an example of usage.

***Table 6-1**. Sub-properties of the hasProperty relating Subjects to Objects.*

Domain	Properties	Range	Example
Agent	enters	Command	The learner enters "ping '*target*'"
Tool	crafts	Packet	Ping crafts ICMP type 8 packet

Tool	sends	Packet	Ping sends ICMP type 8 packet
Tool	solicits	Packet	Ping solicits ICMP type 0 packet
Agent	finds	Information	Learner finds the address of a host
Agent	hasAccess	Tool	Learner has access to use nmap tool
Firewall	accepts	Packet	Firewall accepts ICMP packets
Agent	knows	Information	Agent knows a vulnerable service
Host	establishes	Connection	A connection is established to the target
Agent	creates	File	Learner creates a weaponized file
Host	executes	Program	Payload is executed in the target host
HostUser	hasRights	Administrator	The user of the in-game host has administrator rights

6.6. Scenario Execution Flows

The primary elements are combined to form the *Scenario Execution Flow* (SEF) elements representing APs. The *SEF* elements are represented by the *ScenarioExecutionFlow* class (*Figure 6-3*). The *ScenarioExecutionFlow* class includes a *Goal* object property, sequences of *TaskNodes* and sequences of *Conditions*. The *TaskNode* class is a composite task class. Apart from the properties it inherits from the *Task* class (referring to the *name*, the *label*, *hasSubject*, *hasObject*, *hasProperty*, *hasSource* and *toDestination*), it contains the object properties: *achieves Goal*, *sequenceOf Condition* and *next TaskNode*. The *next TaskNode* property denotes the association of *TaskNode* objects with the subsequent *TaskNode* (or *TaskNodes*) to represent the chain of tasks of an AP. The *has Goal* property represents the association of the *TaskNode* with the goal that the represented AP achieves, while the *relates Condition* denotes the association of the interpreted task with the relevant condition(s). The *relates Condition* property is bidirectional as a condition can be a prerequisite for an executable task, while a task execution can activate a condition or it can cease its influence. The *TaskNode* also defines the *type* data property denoting the type of the element such as the console-command task, the gui-event task and the auto-task. The console-command task designates a command that a learner enters to an in-game console; the gui-event task stands for a task that a learner performs in the game's graphical user interface; and the auto-task represents the tasks performed by the game's engine (e.g., packet sending and crafting). Finally, the *TaskNode* contains the *interval* property denoting the time period that a task requires to execute.

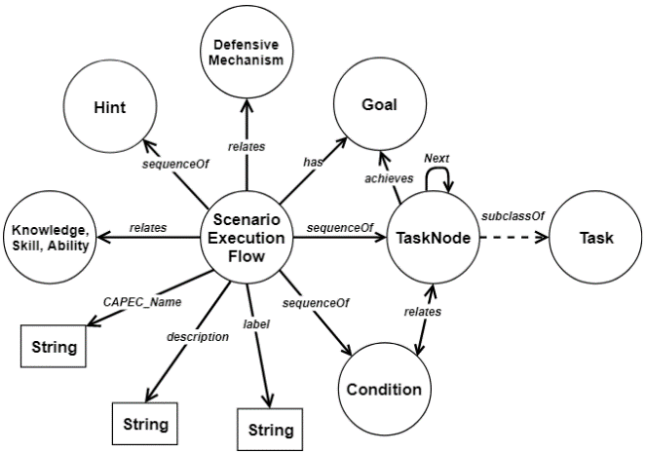


***Figure 6-3**. COFELET Scenario Execution Flow*

The *ScenarioExecutionFlow* class also defines the *relates KnowledgeSkillAbility* object property representing the knowledge and the competencies associated with the corresponding AP and the *sequenceOf Hint* object property representing the list of hints presented to the learner. The *defensiveMechanism* data property is a description of the AP countermeasures, the CAPEC_Name data property holds the name of the CAPEC's attack pattern, and the *description* stores a description of the AP.

The *ScenarioExecutionFlow* objects are grouped according to their goals. During the playtime, learners envisage an attack and they select a single scenario execution flow from a group of SEFs to fulfill a specific SEF's goal. For example, the CAPEC's APs *ICMP Echo Request Ping*, *TCP SYN Ping* and *TCP ACK Ping* are represented by *SEF* objects that share the goal "learner finds network's hosts". The learner can apply any of the aforementioned SEFs to achieve the host discovery goal.

6.7. Roles and Learning Objectives

In general, a role is a position or responsibility that an individual has in an organization or an association. In the COFELET ontology, *roles* are defined based on the cyber security workforce roles found in the National Cybersecurity Workforce Framework (NCWF) (Newhouse et al., 2017), which are herein called *parent roles*. The NCWF associates each one role with a set of tasks, knowledge, skills and abilities required by a cyber security professional assigned the role to successfully perform his/her duties. The COFELET ontology adopts the manner *parent roles* are organized, but it associates them with a sequence of LO elements, which herein are called *parent LOs* and they are defined by utilizing the NCWF's KSAs. In the COFELET ontology, the LO elements are represented by the *LO* class (*Figure 6-4*).

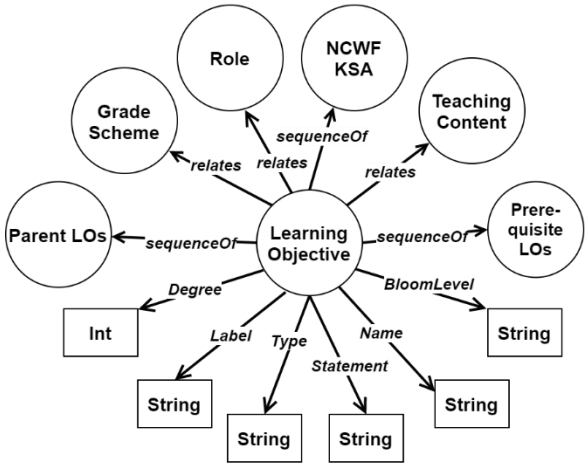


***Figure 6-4**. COFELET Learning Objective*

A Role element is represented by the Role class. The *Role* class contains a role description and a sequence of LOs. The *Role* class associates the LOs it embraces with the following data properties:

- *LO degree*: demonstrates the degree the learner possesses the parent LOs associated with his/her assigned role. The *LO degree* property can be used as a dynamic metric that changes overtime. Its value increases by the amount specified in the *Degree* attribute of the associated LO class (defined in Table 1) when the LO possession is achieved, and decreases by the value of the decay factor property after a period of time specified by the value of the *inactivity* property.
- *inactivity*: indicates the time period of learner's inactivity.
- *decay factor*: specifies the amount of decreasing the value of the *LO degree* property.
- *last update*: is the date and time of the last change of the value of the *LO degree* property.

The *LO* class contains several objects and data properties listed in Table 6-2. The Attributes of LO Class.

***Table 6-2**. The Attributes of LO Class*

Attribute	Rational
Name	the LO's unique name
Label	the LO's user-friendly name
Statement	the LO's statement in the form <`Learner - Property - Object`>
Type	indicates whether the LO is task, knowledge, skill or ability
Degree	is associated with the *LO degree* property defined in the Role class. It indicates the amount of increasing the value of the *LO degree* property

	after a LO possession achievement. Its value is specified by an instructor based on the scenario's complexity and the mission's difficulty
Role	the associated role(s)
BloomLevel	the mapping level in the Bloom's taxonomy
Prerequisite LOs	a sequence of prerequisite LOs the learner has to possess
Parent LOs	a list of parent LOs utilized for the definition of the LO
Teaching Content	texts, figures, and videos presenting the KSAs aimed to be transferred to the learners
Grade Scheme	a rubric according to which the learner's efforts and progress are assessed

As cyber security is a rapidly changing field, new LO objects are necessary to be created on-demand (Katsantonis et al., 2019). In such cases, instructors can define LOs that might not be based on *parent LOs*. Likewise, new roles can be created and associated with the new LOs. In a COFELET game, a new role can be created as a combination of two (or more) COFELET roles, and it can be associated with a sequence of LOs consisting of existing and new LOs. For example, the creation of the role of *Penetration tester* is the result of combining the *Vulnerability Assessment Analyst* and the *Target Network Analyst* roles (parent roles). The *Penetration tester* inherits the LOs from its parent roles (resulting so in a hierarchy of roles), which are related to the knowledge of the penetration testing principles and techniques, the knowledge of threats and cyber-attacks and the proficient use of penetration testing tools.

6.8. Scenarios

A Scenario element contains the appropriate information for the setup of a game session and it consists of three parts (*Figure 6-5*):

1. *Attributes*: the name, the label, the description, the narration and the difficulty level of the scenario.
2. *Cyberspace*: a collection of conditions (i.e., scenario's preconditions) that are in effect when the game session starts, and a set of entities forming the scenario's cyberspace.
3. *Steps*: a sequence of steps corresponding to the stages of a mission. Each step contains a sub-goal, a set of conditions (e.g., pre-conditions and post-conditions), a set of LOs and a sequence of hints (*Table 6-3*).

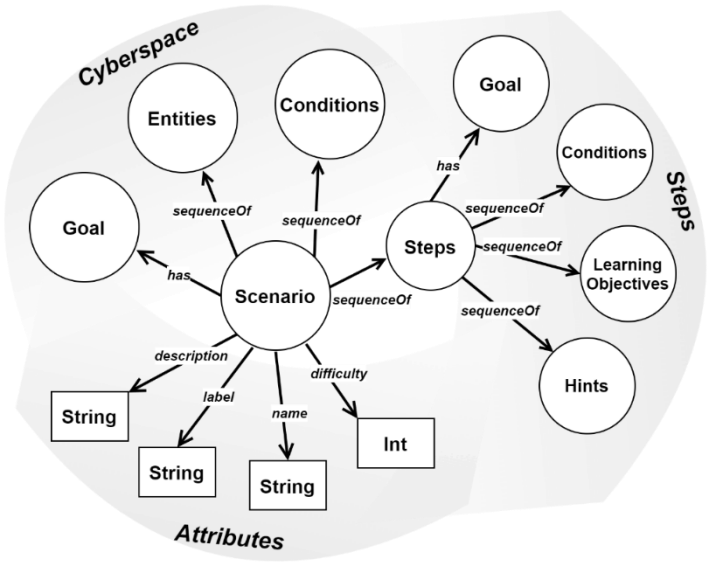


***Figure 6-5**. COFELET Scenario*

***Table 6-3**. The attributes of the Hint class*

Attribute	Rational
Name	the unique name of the Hint
Label	the user-friendly name of the Hint
Ord	the order in which hints are provided to the learner
Text	The suggestion provided to help learners achieve the game goals
Time	Denotes the period after which learners are notified to read the hint

***Table 6-4**. The attributes of the TeachingContent class*

Attribute	Rational
Name	the unique name of the teaching content
Label	the user-friendly name of the teaching content
Content	texts, figures and videos presenting the KSAs aimed to be transferred to the learners
References	references for additional information

6.9. Grade Scheme

The Grade Scheme element is associated with a LO element and a LO element is associated with a Scenario's step (presented in sub-section *6.8*). Thus, a grading scheme is applied to assess the learner's efforts at the end of a step. The Grade Scheme class consists of the following attributes:

1. *grade*: specifies the points assigned to the learner. The grade's value calculation is based on the values of the attributes 2 to 5 presented below.
2. *assessed*: denotes how many times the LO associated with the current grade scheme has been assessed. The value of the attribute is retrieved in the learner's learning history.
3. *hints*: logs the number of hints provided to the learner with respect to the number of available hints in the associated step.
4. *time*: records the time it took the learner to complete the associated step and achieve the possession of the LO.
5. *actions*: logs the number of actions the learner performed in the step to achieve the possession of the LO.
6. *score*: specifies the signed percentage factor applied to the value of the *Degree* attribute specified in the LOs. The result determines the amount of value affecting the *LO Degree* attribute of the Role class.

In many cases, the *score* and *grade* attributes have the same values. However, in some cases the instructor can assign a negative value to the *score* attribute to reflect a negative impact on the *Degree* attribute of the LO class as a disciplinary action (Nagarajan et al., 2012) when learners do not achieve the game's LOs, even when the LO possession has been exercised a number of times. On the contrary, the value range of the *grade* attribute is from 1 to 100.

6.10. Chapter Conclusion:

In this chapter, the COFELET ontology is proposed, an ontology for modeling cyber security learning and training environments, and especially cyber security serious games. The proposed ontology provides an analytical description of the key elements the COFELET games need to comprise to represent cyber security attacks of varying complexities. To this end, the COFELET ontology describes the primary elements (i.e., the high-level elements such as tasks, conditions and goals) and the manner that these primary elements can be combined to form the scenario execution flow elements (SEFs). The SEFs represent attacks that virtually happen in COFELET scenarios and they are described in analogy to CAPEC attack patterns. The COFELET ontology is a step towards the implementation of solutions that respond to the challenge of developing COFELET compliant serious games that dynamically adapt to learners' characteristics and the educational environment; and integrate cyber security standards generally used in threat analysis and modeling approaches (e.g., CAPEC, Cyber Kill Chain, National CyberSecurity Workforce Framework). The COFELET ontology elements are independent of game genres and underlying platforms and technologies. The

COFELET ontology elements can be utilized to form a shared knowledge base that will be extended and used in cyber security learning and training approaches that assimilate hacking activities.

7. THE COFELET GAME LIFE-CYCLE

7.1. Introduction

Aiming at providing insights on how COFELET compliant approaches can be developed, the COFELET game life-cycle (illustrated in *Figure 7-1*) is presented, a blueprint illustrating the design aspects and the course of phases for the development of COFELET compliant games. The COFELET game life-cycle exhibits how the game's major components and the elements of the COFELET ontology are organized in the structure of a COFELET game.

This chapter is organized as follows: initially, the life-cycle of a COFELET game is briefly introduced, followed by the presentation of the main actors involved in the life-cycle of a COFELET game and the manner they have to cooperate. Subsequently, the phases of the life-cycle of a COFELET game are presented along with the manner that the COFELET ontology elements, and the game's components are organized in the structure of the game. Finally, the way that the COFELET game life-cycle realizes the Four-process architecture of the Evidence-centered Design framework is presented.

7.2. Brief Description

The life-cycle of a COFELET game consists of two phases: *Run-Time* and *Build-Time*. The *Build-Time* phase contains two sub-phases: *Game Foundations* and *Game Construction*. In the *Game Foundations* sub-phase, the key elements described in the COFELET ontology are created, whereas in the *Game Construction* sub-phase the COFELET scenario elements are formed. In the *Run-Time* phase, the major components of the game are depicted along with the functions they perform and their interconnections (*Game Operation*).

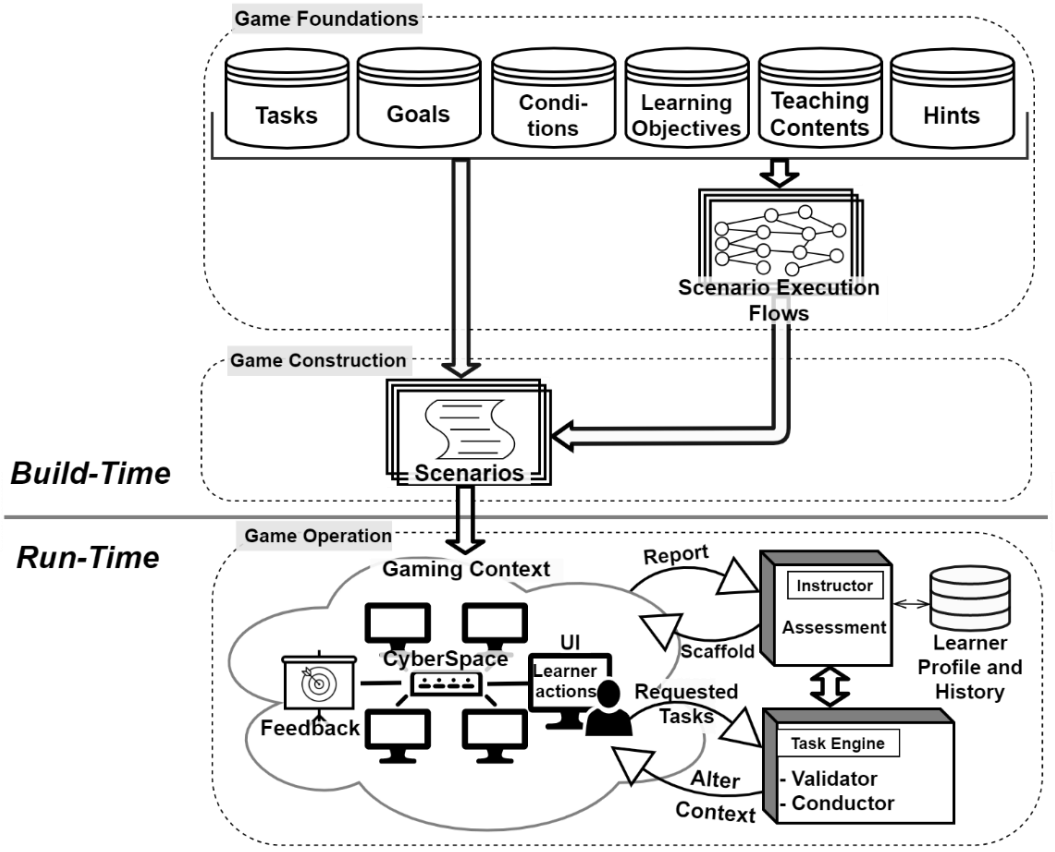


***Figure 7-1**. The COFELET game life-cycle*

7.3. Actors

The use case diagram in *Figure 7-2* depicts the actors' involvement in the COFELET game life-cycle. Specifically, the actors involved are: game developers, cyber security specialists, instructors and learners. *Game developers* work at *Game Foundations* and *Game Construction* sub-phases to create the games by implementing the designs of the cyber security specialists and the instructors. *Cyber security specialists* have deep knowledge of cyber security methodologies and models (e.g., the CKC model (Lockheed Martin, 2014)) and utilize the COFELET ontology at the *Game Construction* sub-phase to design the key elements that will be interpreted in the particular games (e.g., SEFs). *Instructors* are educators, aware of the parent roles and the corresponding KSAs, who complement the work of cyber security specialists at the Game Construction sub-phase by adding the elements that determine the learning and instructional perspectives of COFELET games (i.e., LOs, hints and teaching content). Instructors also can cooperate with game developers at the Game Construction phase to create or edit game scenarios. Conclusively, *learners* are the final recipients of the COFELET games using them at the Game Operation phase.

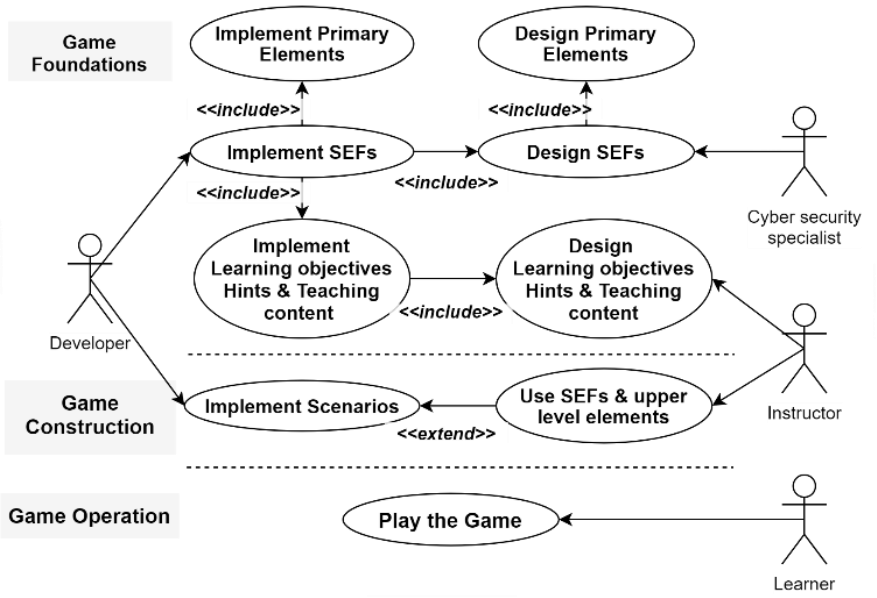


***Figure 7-2**. Use case of the actors involved in the life-cycle of a COFELET game*

7.4. Build-Time

During the *Game Foundations* sub-phase, the repositories of key elements, depicted in the upper part of *Figure 7-1*, are created and they are stored in a manner that facilitates their adoption in diverse games and educational contexts.

During the *Game Construction* sub-phase, instructors create the COFELET scenarios by utilizing the key elements stored in the repositories. Scenarios describe in-game entities by providing the necessary properties for imitating the behavior of real devices (e.g., networked hosts), including some attributes with randomized values that change from session to session. Scenarios can also contain additional elements when instructors need to add extra functionalities and features. In such cases, instructors need to cooperate with game developers during the Game Construction sub-phase. For example, a scenario can include the *Question* elements, which are additional elements representing the questions issued during the game play. The *Question* elements must be explicitly associated with particular LO elements and they regard cyber security concepts (e.g., employed attack patterns).

7.5. Run-Time

The Run-Time phase of the COFELET game life-cycle depicts the following components of a COFELET game:

- *Gaming Context*: contains the user interface façade (UI) and the game's Cyberspace. The game's Cyberspace is the virtual environment in which learners perform their actions and

they unleash their cyber-attacks. It embraces numerous game entities, such as the learner's host, networks, target hosts, servers and services, firewalls, files etc. The UI depends on the genre of the COFELET game. For example, the UI of a hacking simulation game usually includes a command terminal in which the learner enters commands along with a set of windows that embrace additional functionalities (e.g., display information, send messages etc.). On the other hand, the UI of a card game includes a card deck and a game menu. The game's Cyberspace provides feedback to the learner through the facilities that it embraces (e.g., the terminal in the learner's host) and through the game's UI.

- *Task Engine*: is a task operator that conducts the performed tasks and provides feedback to the learner through the game's Cyberspace. It consists of a *Validator* and a *Conductor*. The *Validator* confirms that a task belongs in the sequence of tasks of the employed SEF and validates that a task is executable by inspecting the occurring conditions. The *Conductor* virtually executes a task and checks whether its execution provokes the fulfillment of a goal or a mission. The *Conductor* also sets the post conditions of the executed task and communicates with the *Gaming Context* and *Instructor* components.
- *Instructor*: assesses the learning session and scaffolds the learner's efforts. Specifically, the Instructor component:
    - monitors the learner's progress and acquires the necessary information from the *Task Engine* and the *Gaming Context*. The details acquired include the learner's actions, the tasks performed, the goals achieved and the currently applied SEF by the learner.
    - manages the appropriate key elements such as the scenario's hints, the teaching contents and the LOs whose possession the learner has to achieve.
    - has access to the game's back-end storage facility (e.g., a database, or a collection of XML files) and queries information regarding the learner's profile and history of learning and training.
    - scaffolds the learner's efforts through the provision of hints and teaching contents that are associated with the LOs whose possessions the learner has to achieve. For example, it counts the game play time period and it monitors the learner's progress. Whenever, the game play time period is beyond a time threshold specified by the instructor in the game's scenario (i.e., in the *time* attribute of a hint element specified in subsection '3.2 Scenarios'), the learner is notified and the appropriate hint(s) are shown to her.
    - assesses the learner's fulfillment by applying a grading scheme specified by the instructor actor in the scenario (i.e., the *Grade Scheme* objects). Subsequently, the assessment details are stored in the back-end storage facility and the learner's profile is updated.

The *Run-Time* phase cycle exhibits the manner according to which COFELET games realize the perspectives of:

- Gaming: Particularly, a COFELET game renders the learner actions in two sites: in the game's Cyberspace and in the Task Engine. The game's Cyberspace emulates the real world and it interprets the learner actions under the gaming perspective. For example, a COFELET game can imitate the settings of a live competition by embracing the suitable entities with the appropriate functionalities and attributes. In such contexts, the learners assume the role of a live competition's participant.
- Learning: The learners' actions are additionally interpreted under the learning perspective as the requested tasks are passed to the Task Engine. In the Task Engine, the requested tasks are compared with the SEF's tasks, which are explicitly related with the learning and the instructional aspects of the game (e.g., the LOs, the hints and the teaching materials). In such a way, a game can translate the learner's actions to accomplishments of LOs possession related with obtaining the required KSAs.
- Instructional: The Instructor component assumes the role of an instructor by carrying out activities that take place in the game under the instructional perspective. Such activities are related with assessing the learners' efforts, provisioning hints, and teaching contents explicitly related with the learners' tasks.

7.6. Assessment

The COFELET game life-cycle realizes the Evidence-centered Design (ECD) framework (presented in *2.3.7.1*) as follows:

- In analogy to the Task/Evidence Composite Library of the ECD' Four-process architecture, the COFELET game life-cycle proposes a repository of scenarios.
- In analogy to the *Activity Selection* process of the ECD, the instructor or a scenario selection process chooses the subsequent scenario presented to the learner. The scenario selection algorithm depends on the scenario's learning objectives and complexity in terms of the scenario's number of steps, the size of the Cyberspace, the stringency of the *grade scheme* for the grade evaluation (e.g., number of actions for the performance of tasks specified in the *grade scheme*) and the stress levels (e.g., available time for the performance of tasks specified in the *grade scheme*).
- The scenario is presented to the learner (the *Presentation* process of the ECD). The scenario describes the Cyberspace, provides a narration including the game objectives and the teaching materials associated with the scenario along with references to external sources (texts, videos, images etc.).
- The *Task Engine* and the *Instructor* identify and capture the learner's observable actions (i.e., the tasks provided to the Task Engine and passed to the Instructor) generated by the learner's effort to solve the game's challenges (the *Presentation* process of the ECD). The tasks are

associated with LOs and a *grade scheme*, according to which the score and the grade of learners are calculated.

- The *Instructor* generates the score and provides feedback to the learner (the *Summary Scoring* of the ECD framework). The *Instructor* updates the learner's profile (in analogy to ECD's *Student Model*) stored in the back-end storage facility.

7.7. Chapter Conclusion

After the examination of how COFELET compliant games can be structured using the COFELET framework as a guide, the COFELET game life-cycle was proposed. The COFELET game life-cycle aims at reducing the complexity of designing COFELET compliant games by presenting the architectural and design aspects of COFELET compliant games. More specifically, the COFELET game life-cycle describes the main components COFELET games contain and the manner the COFELET ontology elements are organized in the structure of such games.

8. THE HACKLEARN GAME

8.1. Introduction

In this section, a design excerpt of a COFELET game, called HackLearn, is presented. Initially, the characteristics of HackLearn are presented including its genre, the features it adopts from the live competitions and the characteristics that distinguish it from other cyber security game-based learning approaches. Subsequently, the application of the ATMSG model is demonstrated to exhibit the HackLearn's game flow and to analyze it under the gaming, the learning and the instructional perspectives. Finally, the HackLearn prototype scenario is presented that puts together the elements discussed in previous sections 3 and 4. Specifically, the HackLearn prototype scenario:

- allows learners to apply an attack based on the CKC model (CKC attack),
- exemplifies the COFELET game life-cycle by providing details on the manner its components interact with each other,
- exhibits how the layered learning and the continuous learning approaches (as analyzed in section 5.4 Learning Strategies) are employed,
- demonstrates how the new elements of the COFELET ontology (e.g., Roles, LOs, and Grade Schemes) facilitate the learning and the instructional aspects of the game.

8.2. HackLearn's Characteristics

Cyber security educational games are already used for teaching various cyber security topics in miscellaneous contexts. Researchers in (Hendrix et al., 2016) reviewed and categorized cyber security serious games according to their game type, the methodology they apply, the cyber security topics they aim to teach, the target audience, and the evaluation performed. However, in their literature review, very few cyber security serious games were found with target audience cyber security professionals and university students. Besides, none of the identified games offer opportunities for hands-on experiences and practices, such as the use of cyber security tools to unleash cyber-attacks. On the other hand, commercial hacking simulation games have been around for many years and they are becoming more popular over the past years. At the moment, the Steam game distribution platform (Valve, 2020a) offers more than 20 commercial entertainment hacking simulation games such as 'HackNet', 'hack_me' and 'NITE Team 4'. Hacknet (Fellow Traveller, 2020) is one of the most popular hacking simulators with 1.000.000 to 2.000.000 owners, more than 70.000 followers, and more than 10.000 positive comments (Valve, 2020b). However, the hitherto known hacking simulation games are not included in cyber security education research ((Hendrix et al., 2016), (Katsantonis et al., 2017a), (Mostafa & Faragallah, 2019)), as they are out of the scope of such a research. The reason for this is that they are commercial games that do not clearly have learning and training as their primary objective, rather fun and entertainment. Besides, they are not

designed to operate as learning or training tools in educational contexts, they have a different target audience than cyber security educational games, they do not have clear learning objectives and often they are not based on authentic cyber security topics (e.g., they use logical puzzles).

HackLearn is a cyber security serious game that can be mainly included in the hacking simulation game genre (i.e., hacking simulator), as it is a cyber security educational tool that adopts the characteristics of the genre of hacking simulators. Specifically, HackLearn includes a Unix-like terminal in which players utilize emulations of real-world tools by typing and executing text-based commands (e.g., Nmap, base64, whoami); simulations of cyber-attacks; representations of common cyber security entities and concepts (e.g., hosts, firewalls, services); role-playing experiences as a player assumes the role of a hacker that faces various challenges; and cyber security missions based on scenarios.

HackLearn draws many elements from CtF competitions, designed for educational purposes, as the learners unleash cyber-attacks during the game-play, they collect flags and points, they exercise their knowledge and skills, and they try to beat the clock. Though HackLearn is not a CtF competition exercise or game, it adopts CtF competitions' features and it also tries to overcome the CtF's limitations reviewed and analyzed in (Katsantonis et al., 2017a). Moreover, HackLearn diverges from CtF games because such approaches provide cyber-security hands-on experiences in an unstructured and self-directed manner (Katsantonis et al., 2017a). On the contrary, HackLearn forms a highly organized and parameterized environment that dynamically monitors learners' actions, evaluates their progress and it scaffolds their efforts. To do so, it utilizes the COFELET framework and the COFELET ontology to model strategies and cyber-attacks and to support the learning and instructional aspects of the game.

8.3. Design

This section provides information on the manner HackLearn is designed by presenting how the COFELET game life-cycle components are organized and collaborate. Initially, the ATMSG model is used to describe the HackLearn's game flow and demonstrate the systematic organization of the HackLearn's serious game components. Subsequently, the manner the COFELET game life-cycle components cooperate is presented in a sequence diagram along with a UML class diagram depicting a collection of typical entity objects with the appropriate attributes and functionality, which imitates the behavior of the real devices.

According to the ATMSG model approach, after the identification of the HackLearn's activities, the game components are initially presented in a UML activity diagram, the game's sequence diagram. The diagram in *Figure 8-1* shows HackLearn's sequence diagram presenting the game's components and the manner that they are interconnected throughout the game. The HackLearn's sequence diagram consists of seventeen (17) components, some of which embrace the functionalities of the

Instructor, *Validator*, *Conductor* and *Gaming Context* components of the COFELET game life-cycle (described in *chapter 7*).

Table 8-1 exhibits the analysis elaborated upon the components of the HackLearn's game sequence diagram. Particularly, *Table 8-1* identifies HackLearn's components and classifies them in the perspectives of gaming, learning and instructional according to the activities they embrace. In *Table 8-1* is also specified for each component the actions performed in the game, the tools that make these actions possible and the goals as the objectives that will be achieved after the accomplishment of the actions.

For the design of the HackLearn's components, the elements of the ATMSG taxonomy for serious game components (Carvalho et al., 2015) were utilized. Subsequently, detailed descriptions of the HackLearn's components were derived (presented in *Table 8-2*), including further details on the activities taking place in HackLearn, the game's tools utilized (e.g., the terminal, the progress bar) and the purpose driving these activities. For brevity, the components already presented in chapter 7 are not included in tables *Table 8-1* and *Table 8-2*, though they are illustrated and specified in the HackLearn's sequence diagram (depicted in *Figure 8-1*).

***Table 8-1**. HackLearn's Serious Game Components*

		1.Choose Role	2.Interactive tutorial	3.Diagnostic Assessment	4.Introduce Mission	5.Plan of the Attack	Instructor	8. Perform Action	11. Gaming Context		15. Answer Question(s)
							6-7. Scaffold		Feedback	Reward	
Gaming	**Actions**	Customize	Obtain help	-	Read Story	Plan/Strategy, Match	Obtain help	Create, Generate	Read Information	See Performance Evaluation	-
	Tools	Role	Tutorial	-	Story	Information	Advice and Assistance, Information	2D space, Time pressure	Complete Information	Progress bar, Points, Role/Virtual skills, Status level, Information	-
	Goals	Configure game	Learn to use interface	-	Get acquainted with story (and mission)	Complete quest & side quests	Complete side quests, form/discover goal	Complete quest & side quests	Collect Information	Maximize Performance	-
Learning	**Actions**	-	Observe, Practice	-	Observe, Identify	Observe, Identify Hypothesize, Combine, Plan, Restate	Read, Find more information about	Apply, Recall, Repeat	Verify, Find More Information About	Verify, Review	Describe, Explain, Summarize
	Tools	-	Tips, Tasks	-	Problem, Challenge	Creations, Inventions	Texts, Information, Tips, Definitions	Simulator, Experiment	Texts, Information, Illustrations (text images)	Information, Graphics	Test, Definitions, Conclusions
	Goals	-	Apply, Remember	-	Understand, Analyze	Active Experimentation, Abstract Conceptualization	Remember, Understand	Apply, Concrete Experience	Remember, Understand, Reflective Observation	Understand, Reflective Observation	Understand, Reflective Observation

		1.Choose Role	2.Interactive tutorial	3.Diagnostic Assessment	4.Introduce Mission	5.Plan of the Attack	Instructor	8. Perform Action	11. Gaming Context		15. Answer Question(s)
							6-7. Scaffold		Feedback	Reward	
Intrinsic Instruction	**Actions**	-	Demonstrate, Scaffold	Present Quiz	Tell Story, Present Problem	Repetition, Scaffold	Scaffold, Present material	Reward good performance, Repetition	Demonstrate	Qualitatively assess performance	Present Quiz
	Tools	-	Tips	Question and Answers	Story	Information, Multiple choices, Limited set of choices	Tips/Assistance, Help text	Performance measures, Multiple chances,	Tips, Warning messages	Performance measures	Questions & Answers
	Goals	-	Provide learning guidance	Assess Performance (Initial knowledge)	Inform Learner, Gain Attention	Provide learning guidance	Stimulate recall of prior knowledge, Provide learning guidance	Elicit performance	Provide feedback	Assess performance, Provide feedback	Stimulate recall of prior learning

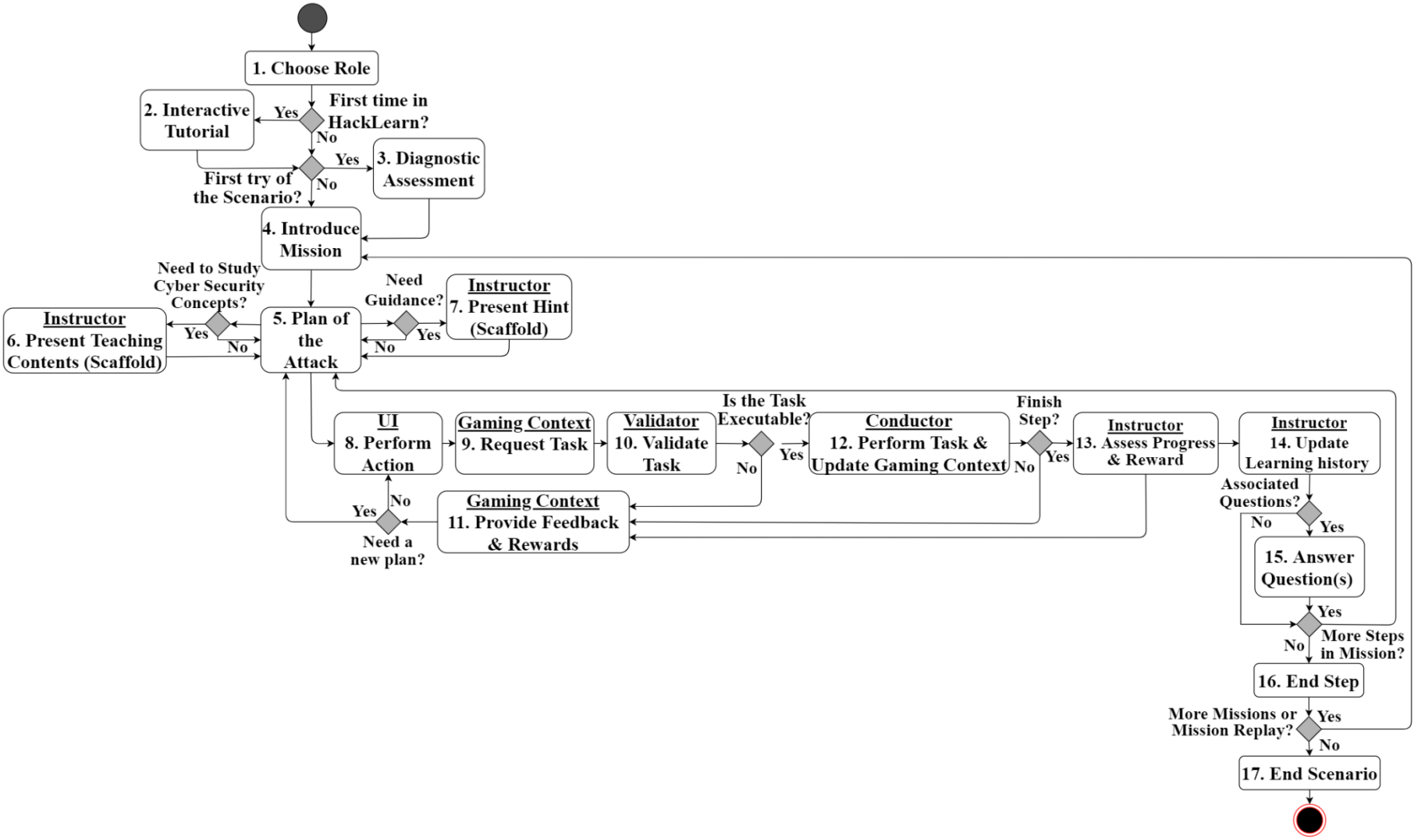


Figure 8-1. *HackLearn's Sequence Diagram*

***Table 8-2**. Detailed Description of HackLearn's Serious Game Components*

Node	Gaming	Learning	Intrinsic Instruction
1. Choose Role	The learner chooses a game role associated with LOs whose possession must be achieved. LOs are also associated with the scenarios to be assigned to the learner.		
2. Interactive tutorial	At the beginning of a game session, the learner encounters pop-up messages that explain the basic features of the UI and require him/her to perform certain actions (e.g., enter a help command in HackLearn's terminal, use the toolbar).	The learner reads the text of the pop-up messages, applies basic UI actions and observes the results of his/her actions. In such a way, s/he remembers basic UI actions when the mission starts.	The tips in the pop-up messages and the related graphics provide learning guidance by directing the learner to perform basic UI actions and acquire the necessary skills in using HackLearn.
3. Diagnostic Assessment			Participants complete a questionnaire to determine their initial knowledge prior to the game session.
4. Introduce Mission	It demonstrates the mission (i.e., the narration attribute of the scenario object) stating the story and the goals of the mission (e.g., to capture the flag).	The learner reads the scenario's narration to understand the problem, to analyze it and to identify objectives and clues.	It informs about the game's objectives and it uses storytelling to grab the learner's attention and to motivate her.
5. Plan of the Attack	The learner envisages an attack, decides on the strategy she will employ to reach the game's goals and selects a SEF to apply.	The learner observes the mission's objectives and the scenario's cyberspace, combines information, makes hypotheses and elaborates the plan of the attack (active experimentation). If the plan fails, the learner comes to conclusions (abstract	It scaffolds the efforts of novice learners by displaying the available SEFs and tools or by requiring learners to select only applicable SEFs.

		conceptualization) and repeats the process by utilizing what s/he has learned from his/her previous experiences.	
6-7. Scaffold	The learner obtains short suggestions for the completion of the current step (e.g., applicable SEFs, usage of tools, command syntax,) and teaching materials explaining the cyber security concepts involved (e.g., attack patterns, techniques, tools etc.).	The learner studies and recalls the SEFs, the tools and their applicability in the cyberspace of the HackLearn's scenario.	It scaffolds learner's efforts (e.g., to choose the proper SEF or use the appropriate tools) and it keeps learners motivated. It gradually supports the player's efforts by successively providing hints that merely reveal part of the solution and increase the game's support.
8. Perform Action	The learner performs actions by entering Unix-like textual commands in the HackLearn's terminal.	The learner repeats commands to develop patterns of tool usage under the appropriate conditions.	It measures the number of tries and it counts the time taken to successfully perform a *Task* (i.e., action directed at the fulfillment of *Goals*).
11. Gaming Context: Feedback & Reward	It provides: 1) feedback mainly through the game's terminal, 2) rewards through the UI elements (e.g., score) and the game's graphics (i.e., progress bar) informing the learner about a successful or an unsuccessful execution of a task and an attack.	The learner realizes the results of his/her efforts, reflects on the actions performed and envisages how to improve performance.	It provides points as an incentive for good practices or penalty points (or no points) as disciplinary actions for repeated mistakes. Warning messages and tips inform the learner about the *Conditions* ignored or overlooked.
15. Answer Question(s)		The learner answers questions that make him/her reflect on his/her achievements and describe or discuss concepts, techniques, methodologies etc.	It assesses the learner's knowledge gain

HackLearn includes a terminal in which players enter text-based commands that utilize Unix-like tools to perform the game's tasks (the command execution action). The sequence diagram in *Figure 8-2* shows the manner that the components (depicted in the Run-Time phase of *Figure 7-1*), interact to perform command execution actions. Once the learner enters a command, the terminal renders the command's arguments and passes the command to the appropriate built-in tool. The tool makes all the appropriate audits, reports the learner's action to the instructor component, and passes the corresponding task to the *Task Engine*. Then, the tool gets the response, performs the command and alters the gaming context. Finally, the learner receives feedback from the UI in various forms (e.g., scores, visualizations, sounds, etc.) and from the terminal in textual form.

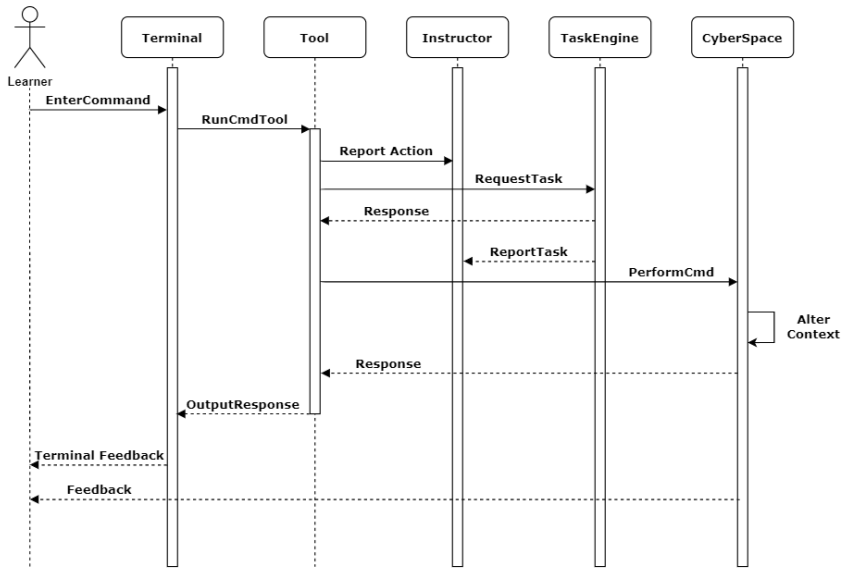


***Figure 8-2**. Command execution sequence diagram*

In the UML class diagram of Figure 8-3, a collection of entity objects of a typical HackLearn scenario including a network with target hosts, interfaces, services, files and a firewall is presented. Figure 8-3 depicts the attributes and functionality of entity objects required to imitate the behavior of the real devices.

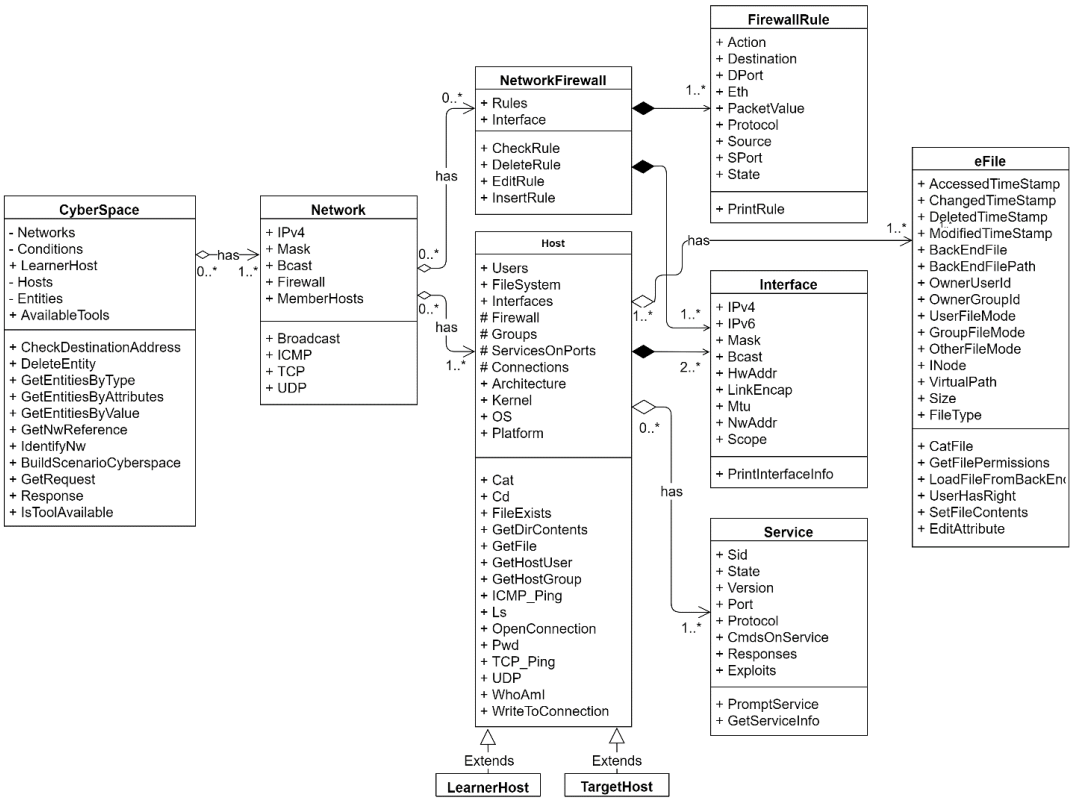


***Figure 8-3**. Class diagram of a typical scenario's entities*

8.4. Environment

To play HackLearn, the learner has to create an account (Figure 8-4 (b)) by registering her/his details and choose the role that she will have in the game (Figure 8-4 (b)). Once the learner has an account, she visits the HackLearn's login screen (Figure 8-4 (a)) and enters the username and password.

(a) (b)

***Figure 8-4**. HackLearn's log in (a) and register screens (b)*

Then, the game's front-end communicates with the game's back-end to check the learner's credentials and the learner's profile. If the learner has previous experience with

HackLearn, the main scene is loaded with the mission panel enabled (*Figure 8-5*) to read the mission. The main scene consists of the *terminal* in which the learner executes Linux-like commands, the *right panel* in which the game's windows appear, a *toolbar*, and a *progress bar*. The *toolbar* contains the icons *profile*, *teaching contents*, *inquiry*, *leaderboard*, *mission*, *hint popups*, *messenger* that allow learners to pop up the game's windows in the *right panel*. The *toolbar* also displays a *time counter* and the *learner's name* and it includes the pause button from which the learner enables the pause menu and quits the game. The progress bar displays the learner's *progress* in the mission and his/her *score*.

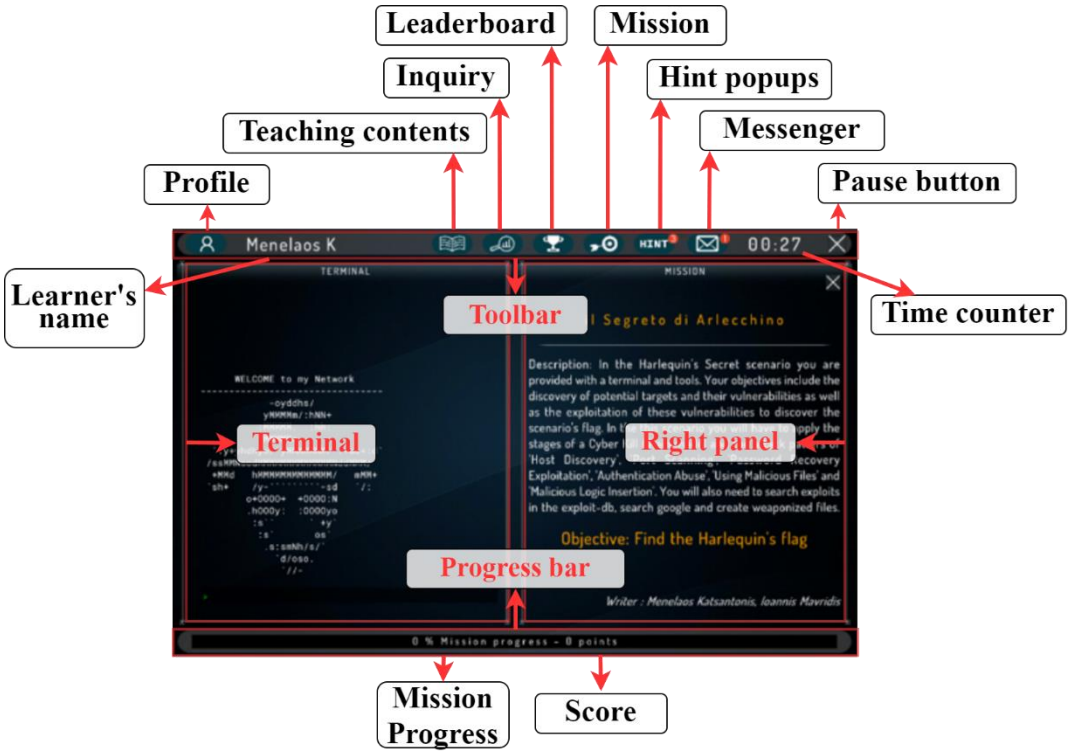
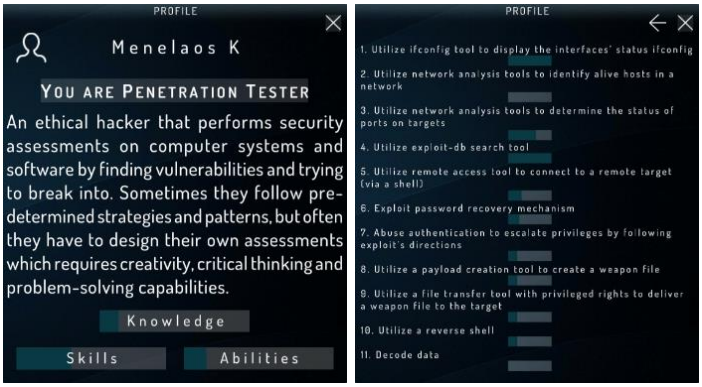


***Figure 8-5**. HackLearn's main scene*

The learner can open her/his profile window to review the competencies she has to acquire progress, the progress s/he made and the progress she has to make (*Figure 8-6* (b)).



(a) (b)

***Figure 8-6**. Profile window*

If it is the first time that a learner logs in the game, an interactive tutorial is presented to help the learner familiarize with the game's interface (*Figure 8-7*).

***Figure 8-7**. HackLearn's tutorial*

HackLearn's missions can be associated with in-game questions that pop up during the gameplay in the inquiry window (*Figure 8-8*). The in-game questions are short answer questions that pop up after the completion of the steps. An in-game question can be *compulsory* or *optional*. A compulsory in-game question requires the user to answer it in order to proceed, whereas an optional in-game question does not oblige the user to answer.

***Figure 8-8**. Pop-up windows with an in-game question example*

8.5. The Prototype Scenario

The COFELET framework envisages scenario-based learning and training approaches tailored to the LOs to be achieved, the learners' characteristics, and the properties of the educational context. In this study, the il Segreto di Arlecchino prototype scenario (i.e., the secret of Harlequin in Italian) was developed and used along with HackLearn in the presented study. The aim of the il Segreto di Arlecchino scenario is to make the learner comprehend and apply the most of the seven (7) stages of the CKC model to unleash an advanced persistent threat (APT) attack. The target audience of the il Segreto di Arlecchino scenario is computer scientists with prior knowledge in networks, operating systems and cybersecurity tools, whereas it was planned to be delivered in a formal educational context. The scenario's goal is to attack the Harlequin target host and find and capture the file flag.txt stored in this host.

8.5.1. Description

The il Segreto di Arlecchino scenario is a composite scenario consisting of nine (9) steps in which learners have to apply the stages of the CKC model (Martin, 2014) and perform 8 attack patterns. The il Segreto di Arlecchino scenario contains several game entities (i.e., the network, two hosts, the tools, the commands) and its implementation is based on the design presented in (Katsantonis et al., 2021) together with its associated COFELET elements: LOs, roles, grade scheme, hints and teaching content.

The scenario draws many elements from the cultural sector (e.g., theatre, music, cinema) that enhance the fun factor and motivation of the learner. Specifically, it draws many cultural elements from the comedic theatre commedia dell'arte (i.e., the Italian comedy) as it adopts the character names of commedia dell'arte to label the hosts, the network

and the directories. In fact, the realization of the commedia dell'arte metaphor in the game can help the learner to better comprehend the functions that take place in the target host. For example, the Harlequin host (i.e., the scenario's target host) has a directory called 'kitchen' used by the zanni user group. Zanni in commedia dell'arte are the servants that carry out the characters' orders. The kitchen directory has low privilege rights as the zanni come and go and they are expected to have low security awareness. Thus, the learner has to figure that the kitchen directory is a good place to put a malware because the zanni users have a good chance to consume a weaponized file. Moreover, in the scenario the learners are required to search details about Andrea Calmo, the author of the commedia dell'arte and when they inspect the target, they will find clues related to the Joker movie character (i.e., an associate of Harlequin) and the Queen band, a band that used elements from the Italian literature in their lyrics.

8.5.2. Cyberspace

Figure 8-9 illustrates the main entities of the scenario's cyberspace with which learners interact. The learner has to search the VictoryBall network, discover the *Harlequin* host, and scan the services of Harlequin to find the vulnerabilities that will allow his/her to gain access and capture the flag. The flag is stored in the '/home' directory of the system administrator's account (root), and thus the learner has to get the administrator's rights to have access to the target directory and capture the flag.

The learner host is the primary entity with which learners interact and it simulates the functions of a host running the Linux operating system containing the appropriate tools (e.g., Nmap, Metasploit, msfvenom), which simulate the functionality of real Linux tools used in cybersecurity.

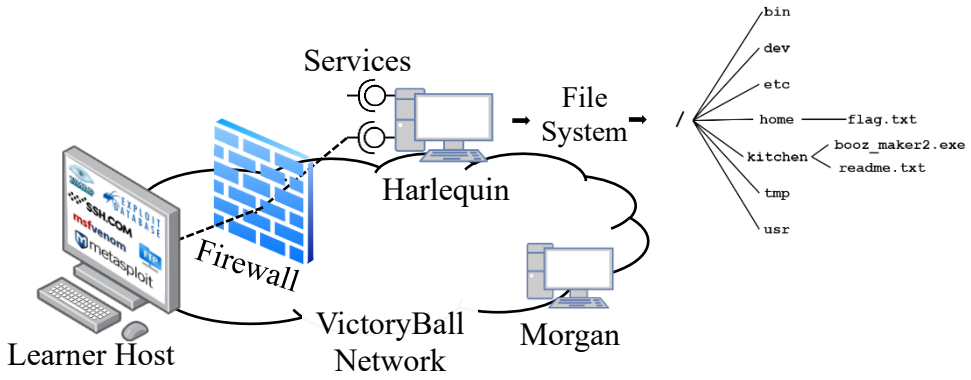


***Figure 8-9**. Cyberspace of the il Segreto di Arlecchino scenario*

The *learner host* is associated with the appropriate *condition* elements, which according to the COFELET framework indicate whether the learners' tasks are performable. For example, the cyberspace contains the condition <Learner - has - privileged

rights – to learner host> which provides administrator rights to learners to have access to several tools (e.g., the *condition* <Learner - hasAccess – nmap port scanner tool>) and functions. The *firewall* entity involves several *condition* elements controlling the flow of the packets such as the condition <Firewall - drops - All ICMP Packets> which indicates that firewall drops all the ICMP packets and the condition <Firewall - accepts - TCP Packet - from learner host- to all target hosts> which indicates that the firewall accepts the TCP SYN packets destined from learner host to the target hosts. The target hosts contain several entities (e.g., users, services, files) and conditions. For example, the cyberspace specifies that the learner needs administrator rights on the *Harlequin* target host to access the file *flag.txt*, whereas she does not need privileged rights to access the contents of the folder *kitchen*.

8.5.3. Tools

The HackLearn game includes in-game tools (i.e., the tool entities described in the COFELET ontology) which simulate the functionality of real Linux tools used in cybersecurity. The tools needed in the *il Segreto di Arlecchino* scenario are:

- base64: encodes and decodes data into ASCII text.
- cat: displays the file contents
- ftp: transfers files to and from a remote host
- ifconfig: displays the network interfaces of the host and their status
- msfconsole: is the terminal interface of Metasploit. Metasploit is a penetration testing platform in which penetration testers create exploit code, search the database of exploits, or utilize exploits.
- msfvenom: a tool for creating the weaponized files to deliver to the target
- nmap: a network analysis tool
- ping: sends a ICMP message to a host that requests a ICMP response message.
- pwd: prints the name of the current directory
- searchslploit: searches exploits in the exploit-db site (https://www.exploit-db.com/)
- ssh: connects to a remote computer
- whoami: prints the user's name in the current shell

8.5.4. Learner's Role

The *il Segreto di Arlecchino* scenario has a target group of computer scientists with a strong background in cyber security aiming at acquiring knowledge and competencies of the role of a *Penetration Tester*. The *Penetration Tester* is based on the *Vulnerability Assessment Analyst*, the *Target Network Analyst* and the *Cyber Operator* parent roles of NCWF. The *Penetration Tester* is mainly associated with the NCWF tasks and KSAs of its parent roles presented below along with their NCWF code. The first letter of the code denotes whether it is a task, knowledge, skill or ability.

1. S0051: Skill in the use of penetration testing tools and techniques.
2. T0616: Conduct network scouting and vulnerability analyses of systems within a network.
3. K0177: Knowledge of cyber-attack stages (e.g., reconnaissance, scanning, enumeration, gaining access, escalation of privileges, maintaining access, network exploitation, covering tracks).
4. T0724: Identify potential points of strength and vulnerability within a network.
5. A0058: Ability to execute OS command line (e.g., ipconfig, netstat, dir, nbtstat).
6. K0471: Knowledge of Internet network addressing (IP addresses, classless inter-domain routing, TCP/UDP port numbering).
7. K0071: Knowledge of remote access technology concepts.
8. S0267: Skill in remote command line and Graphic User Interface (GUI) tool usage.
9. K0536: Knowledge of structure, approach, and strategy of exploitation tools (e.g., sniffers, keyloggers) and techniques (e.g., gaining backdoor access, collecting/exfiltrating data, conducting vulnerability analysis of other systems in the network).
10. S0081: Skill in using network analysis tools to identify vulnerabilities. (e.g., fuzzing, nmap, etc.).
11. S0191: Skill in assessing the applicability of available analytical tools to various situations.
12. S0293: Skill in using tools, techniques, and procedures to remotely exploit and establish persistence on a target.
13. T0696: Exploit network devices, security devices, and/or terminals or environments using various methods or tools.
14. A0058: Ability to execute OS command line (e.g., ipconfig, netstat, dir, nbtstat).
15. K0471: Knowledge of Internet network addressing (IP addresses, classless inter-domain routing, TCP/UDP port numbering).

These tasks and KSAs form the basis of the LOs included in the profile of learners who assume the role of *Penetration Tester*. In the subsequent section, the Penetration Tester LOs are presented as part of the step's attributes.

8.5.5. Steps

In the il Segreto di Arlecchino scenario, the learner has to achieve the gaming goal of acquiring the file flag.txt by achieving the scenario's goals. According to the COFELET framework, the learner achieves the steps' goals by performing tasks with respect to the occurring conditions. As the primary LO of the il Segreto di Arlecchino scenario is to teach the CKC model (Katsantonis et al., 2021), the rationale of the scenario is based on this model. Thus, the learner has to plan a strategy which follows the stages of the CKC model and unleash an APT attack.

Initially, the learner performs SEFs of discovering the target hosts and the vulnerable services on the target hosts (i.e., the Reconnaissance stage of the CKC model). Then the learner creates a weaponized file (i.e., second stage - Weaponization stage), which s/he delivers to the target (third stage - Delivery stage). The file is consumed by the target (fourth and fifth stages - Exploitation and Installation), a backdoor is created to the target and a connection is delivered to the learner's host (sixth stage - Command and control).

Most of the scenario's steps are associated with corresponding in-game questions aiming to make the learner reflect on the activities she performed in a step and express the knowledge and competencies s/he exercised in a different form of representation (e.g., textual form). For example, in the question depicted in *Figure 8-8*, the learner reflects on an activity that did not bring the result she expected when s/he used the ICMP ping technique.

The remainder of this section provides information on the learner's experiences in the il Segreto di Arlecchino scenario by presenting a brief description of the tasks the learner has to perform along with the related entities (e.g., tools, hosts, files), the occurring conditions, the associated in-game questions and the associated KSAs of the NCWF. Steps 1 to 9 show that the complexity of the scenario's steps evolves as the learner proceeds with her/his mission. Specifically, in the first steps (e.g., Step 1 to Step 3) the learner performs obvious attack patterns (e.g., host discovery, port scanning) but as she proceeds to the scenario's succeeding steps, s/he has to be more creative and think outside of the box to fulfill the steps' goals (e.g., seek who Andrea Calmo was, guess Calmo's username, pick a target folder to deliver the weapon).

Step 1	
Goal	<Learner, finds, target network info>

Description	Learner uses the ifconfig tool to find the network that the learner host belongs to. The network's address is an IP of the form '192.168.*.0/24' in which the * is a random integer with a range from 10 to 222.	
Action	`ifconfig`	
	Type of Action	Console-command task
Attack Patterns	-	
Step's Question	*What does the command 'ifconfig' display?*	
Received Message	The learner receives a message from his/her mentor providing some information on the mission she has to complete such as the scenario objective, the number of steps, the strategy she has to apply (e.g., the CKC model) etc.	
LO L1.1	Learner utilizes the ifconfig tool to display the interfaces' status ifconfig.	
	Bloom Level	Application
	NCWF KSA	A0058
LO L1.2	Learner explains what the ifconfig tool displays	
	Bloom Level	Comprehension
Teaching Content	Adopted from the Linux manual (https://man7.org/linux/man-pages/man8/ifconfig.8.html)	
Hints	1. Find out your network IP address. 2. Use the Network interface configuration tool to discover your network's IP address. 3. Type the command 'ifconfig' in the console.	

Step 2		
Goal	<Learner, finds, target hosts info> The learner finds info (e.g., IP addresses, domain names) of the alive hosts in the target network.	
Description	The learner uses the Nmap network analysis tool with a host discovery option (e.g., the TCP SYN ping option or 'PS') to find alive hosts in the network. In case that the learner utilizes the ICMP ping type option, she is informed that the network's firewall drops ICMP packets and thus she has to use a different option.	
Action	`nmap -PS 192.168.*.0/24` or `nmap -PA 192.168.*.0/24` where * is a random integer with range from 10 to 222.	
	Type of Action	Console-command task
Attack Patterns	Learner can find potential target hosts by employing any child attack pattern of the CAPEC-292 *Host Discovery* parent attack pattern:	

	1. CAPEC-285: *ICMP Echo Request Ping*, 2. CAPEC-299: *TCP SYN Ping*, 3. CAPEC-297: *TCP ACK Ping*.	
Step's Question	*Did you try to ICMP ping your target (e.g., with ICMP Type 8 Echo Request datagrams) with an appropriate tool and option (e.g., 'ping -b |target|' or 'nmap -PE |target|')? Did it work or do you think it would have worked? Explain why.*	
LO L2.1	Learner utilizes network analysis tools (i.e., host scanners) to identify alive hosts in a network.	
	Bloom Level	Application
	NCWF KSA	S0081, S0051
LO L2.2	Learner interprets network address information displayed by the ifconfig tool.	
	Bloom Level	Application
	NCWF KSA	K0471
LO L2.3	Learner assesses the applicability of network analysis tools and options in various contexts network analysis tool.	
	Bloom Level	Evaluation
	NCWF KSA	S0191, S0051
Teaching Content	Adopted from the *Host Discovery* attack pattern of CAPEC (https://capec.mitre.org/data/definitions/292.html)	
Hints	1. Identify alive hosts in the network. 2. Use a tool that sends probes to an IP address to determine if the host is alive. Your goal is to send a packet through to the IP address and get a response from the host. You can start with a range of IP addresses belonging to a target network. 3. Use a host discovery tool that sends TCP Packets and solicits the response. 4. Simply TCP SYN or TCP ACK your target (e.g., 'nmap -PS |target network|' or 'nmap -PA |target network|').	

Step 3	
Goal	<Learner, finds, target ports info> The learner finds info related to the open ports of the target hosts.
Description	The learner uses the Nmap network analysis tool to scan the ports of the hosts discovered in Step 2 and finds information on the services running on these hosts. Among other services, the learner discovers the service 'FTP_dell_arte 2.5.7c' running on the Harlequin host.
Action	`nmap -sS 192.168.*.27` or

	`nmap -sA 192.168.*.27` or `nmap -sC 192.168.*.27` where * is a random integer with range from 10 to 222.	
	Type	Console-command task
Attack Patterns	The learner can find potential target hosts by employing any child attack pattern of the CAPEC-300 *Port Scanning* parent attack pattern: 1. CAPEC-287: *TCP SYN Scan*, 2. CAPEC-305: *TCP ACK Scan*, 3. CAPEC-301: *TCP Connect Scan*.	
Step's Question	*Is a filtered target port considered opened or closed?*	
LO L3.1	The learner utilizes network analysis tools (i.e., port scanners) to determine the status of ports on the targets.	
	Bloom Level	Application
	NCWF KSA	S0081, S0051
LO L3.2	Learner distinguishes the port states.	
	Bloom Level	Comprehension
	NCWF KSA	S0081, S0051
Teaching Content	Adopted from the *Port Scanning* attack pattern of CAPEC (https://capec.mitre.org/data/definitions/300.html)	
Hints	1. Identify open ports on the target hosts. 2. Use a port scanner tool that sends probes to an IP address/port and determines the status of the port. 3. Scan your target(s) using a port scanner (e.g., Nmap) and the appropriate Scan option (e.g., TCP Scan option). 4. Use the command: 'nmap –sS |target|'.	

Step 4		
Goal	<Learner, finds, exploits info> The learner finds exploits in the exploit-db matching potential vulnerable services discovered in Step 3.	
Description	Learner searches the exploit database to find an exploit that can be used on the vulnerable service discovered in Step 2. However, the exploit requires that the attacker has the credentials of a legitimate user (e.g., guest).	
Action	`searchsploit ftp`	
	Type	Console-command task
Attack Patterns		

Step's Question	*Comment on the statement: 'Vulnerabilities and Exploits are more or less the same thing'?*	
LO L4.1	The learner identifies potential points of vulnerability.	
	Bloom Level	Evaluation
	NCWF KSA	S0293
LO L4.2	Learner utilizes the exploit-db search tool (i.e., a service that provides a collection of vulnerabilities and code exploits).	
	Bloom Level	Application
	NCWF KSA	T0724
LO L4.3	Learner distinguishes the concepts of vulnerabilities and exploits.	
	Bloom Level	Comprehension
Teaching Content	Text explaining the concepts of exploits, vulnerabilities and the usage of the tool searchsploit.	
Hints	1. Find exploits related to the services you discovered on targets' open ports. 2. Use an exploit-db search tool to search the database of exploits. 3. Use the 'searchploit' tool with the names of services you discovered as keywords. 4. Search the exploit-db with: 'searchsploit |service name/version|'.	

Step 5	
Goal	<Learner, finds, password recovery info> The learner finds exploits matching the vulnerable software running on open ports by searching an exploit database.
Description	The learner uses ssh, a remote connection tool to connect to the service. When the learner connects to the service, she sees the message 'Mr Calmo please enter your login and password'. The learner has to guess the credentials of the Calmo user. To do so, she has an unlimited number of tries to guess the username (i.e., Calmo). However, she will not be able to guess the password. After two failed tries the service will enter the password recovery mode and the learner will see the message 'Please provide your hometown to verify that you are Andrea Calmo'. The learner has to search the internet in order to find out who Andrea Calmo is and the place of his birth (i.e., Venice). Then, the learner sees the message 'PASSWORD HINT: Your password is your hometown followed by your year of birth'. Learner exits the connection to the service.
Action	1) `ssh -p 1571 192.168.*.27` 2) username `Calmo`

	3) 2 password guesses (to activate the password recovery mechanism) 4) `Venice`	
	Type	Console-command tasks and search on the internet
Attack Patterns	The learner uses a remote connection tool to connect to the service and finds out that the guest account is inactive. Subsequently, learner finds a weakness in the password recovery mechanism and exploits it to get the password hint of a legitimate user. Then, learner excavates the user's personal information to find the user's credentials. The AP of this step is based on the AP of the CAPEC-50: Password Recovery Exploitation.	
Step's Question	*Explain how you would have implemented a password recovery mechanism*.	
Received Message	The learner receives a message from his/her mentor asking his/her to consider the use of Wikipedia	
LO L5.1	The learner utilizes a remote access tool to connect to a remote target (via a shell).	
	Bloom Level	Application
	NCWF KSA	S0293
LO L5.2	Learner exploits password recovery mechanism.	
	Bloom Level	Analysis, Application
	NCWF KSA	T0724
LO L5.3	Learner describes a secure password recovery mechanism.	
	Bloom Level	Comprehension
Teaching Content	Adopted from the *Password Recovery Exploitation* attack pattern of CAPEC (https://capec.mitre.org/data/definitions/50.html)	
Hints	1. The remote service has a weak password recovery policy. Exploit it!!! 2. Connect to the target with a remote access tool, guess the username and exploit the password recovery policy 3. Connect to the target with the ssh tool (use the -p option to specify the target port), use the Surname of the user as the username and enter random passwords to initiate the password recovery mechanism 4. Connect to the target, use the 'Calmo' username and enter random passwords. Then look up on the internet the place and year of birth of Andrea Calmo (the author of Commedia dell'Arte), use the password hint and you have the credentials of a legitimate user.	

Step 6	
Goal	<Learner, gets, privileged rights>

Description	The learner utilizes the acquired password (i.e., Venice1510) with the exploit information to enter the ftp service on the Harlequin host. She inspects target's files and directories, and she finds out that she has privilege rights for the kitchen directory. The kitchen directory is used by the waiters to share information and software regarding orders, the creation of cocktails etc. The learner finds out that it contains the booz-maker.exe file that can be used as a template for creating a payload file.	
Action	1) `ftp -p 21 192.168.22.27` 2) traversing in the Harlequin's file system by utilizing the `cd` (change directory) and the `ls` (i.e., list contents) commands	
	Type	Console-command task
Attack Patterns	The employed AP is based on the AP Authentication Abuse CAPEC-114. After the realization that the acquired credentials refer to a user with low privileges, learner utilizes the credentials with the exploit found in the Step 3 to obtain unauthorized access to the target host. Then, the learner inspects the files and the directories of the target and finds out that she does not have access to the directories and files of the system, and thus she cannot find the flag. However, the learner has privileged rights on a distinct directory containing an exe file.	
Step's Question	-	
LO L6.1	The learner uses an exploit to abuse authentication and escalate his/her privileges.	
	Bloom Level	Application
	NCWF KSA	-
Teaching Content	Adopted from the Authentication Abuse attack pattern of CAPEC (https://capec.mitre.org/data/definitions/114.html)	
Hints	1. Exploit the authentication mechanism of the FTP service and search your target for convenient folders to put your backdoor and for useful files in the weapon creation phase. 2. Use the ftp tool to connect to the target's service, use the exploit found in the previous phase to escalate your privileges while logging to the target service and look for template files for your backdoor 3. Use the ftp tool with 'p' option to specify the target port, login to the ftp service (username 'Calmo' and password 'Venice1510_hahaha') and search the 'kitchen' directory for template files. 4. Use the 'ftp -p 21 |target's IP|' command to connect to the ftp service, find the kitchen directory ('cd /kitchen') and download the booz_maker2.exe template file ('get booz_maker2.exe' command).	

Step 7	
Goal	<Learner, creates, weapon file> The learner creates a payload file or weapon by using a payload generation program (e.g., msfvenom).

Description	The learner utilizes the msfvenom tool and she specifies the reverse_tcp payload along with the booz-maker2.exe file to create the booz-maker3.exe payload file. Optionally, the learner can designate the encoder that will encode the payload (e.g., by specifying the 'x86/shikata_ga_nai' encoder) and the number of iterations that the encoder performs, but it is not obligatory in the presented scenario. The reason for this is that the main learning goal of the presented scenario is for the learner to comprehend and apply the CKC and not to learn the options and the possibilities of the msfvenom tool.	
Action	`msfvenom -p reverse_tcp -x booz_maker2.exe > booz_maker3.exe`	
	Type	Console-command task
Attack Patterns	Learner utilizes a payload maker tool and the exe file discovered in the previous step as a template to create a weapon file.	
Step's Question	What is the objective of the weaponization phase of the Cyber Kill Chain model?	
LO L7.1	Learner utilizes a payload creation tool to create a weapon file.	
	Bloom Level	Application
	NCWF KSA	S0293, K0177, A0058
LO L7.2	Learner explains the objectives of CKC's weaponization phase.	
	Bloom Level	Comprehension
Teaching Content	Adopted from the *Password Recovery Exploitation* attack pattern of CAPEC (https://capec.mitre.org/data/definitions/50.html)	
Hints	1. Create a weapon file that the target will consume. 2. Use a tool that creates a weapon file based on a template the target is very likely to consume. 3. Use the msfvenom tool. Make sure that you use the 'p','x','LHOST' and 'LPORT' options and an executable file retrieved from the target. 4. Use msfvenom tool with 'x' option and the |executable file| retrieved from the target (e.g., msfvenom -p reverse_tcp -x |executable file|).	

Step 8	
Goal	<Learner, uploads, weapon file, to target> Learner delivers the weapon to the target host
Description	Learner utilizes the ftp file transfer tool with privileged rights to connect to the target service and deliver the weapon file (i.e., booz-maker3.exe) to the kitchen directory.
Action	1) `ftp -p 21 192.168.22.27` (connect to the ftp service) 2) change directory to visit the *Kitchen* directory 3) `put booz_maker3.exe` (transfer the file to the target)

	Type	Console-command tasks and search on the internet
Attack Patterns	The learner employs the CAPEC-17 *Using Malicious Files* attack pattern to connect to the target and to deliver the weaponized file.	
Step's Question	*What is the objective of the delivery phase of the Cyber Kill Chain model?*	
LO L8.1	The learner utilizes a file transfer tool with privileged rights to deliver a weapon file to the target.	
	Bloom Level	Application
	NCWF KSA	S0293, K0177, A0058
LO L8.2	The learner explains the objectives of CKC's delivery phase.	
	Bloom Level	Comprehension
	NCWF KSA	K0177
Teaching Content	Adopted from the CAPEC's attack pattern *Using Malicious Files* (https://capec.mitre.org/data/definitions/17.html)	
Hints	1. Deliver the weaponized file to the target. 2. Use a file transfer protocol tool to upload a file to the proper directory. 3. Use the commands 'ftp -p 21 |target's IP|' and 'put |weapon|' to the kitchen folder.	

Step 9		
Goal	<Learner, establishes, channel, to the target> and <Learner, finds, flag file> The learner creates a new user by using the appropriate tool for remote access (e.g., the meterpreter)	
Description	The learner starts the Metasploit Framework penetration testing tool and uses the console to utilize the backdoor to connect to the host with administrator rights. Then, she inspects the files of the Harlequin host and in the '/home' directory discovers the 'flag.txt' file. The mission is fulfilled.	
Action	1) `msfconsole` 2) `run` 3) traverse to the *home* directory 4) `get flag.txt`	
	Type	Console-command tasks and search on the internet
Attack Patterns	The learner employs the CAPEC-441 *Malicious Logic Insertion* attack pattern to connect to the target and to deliver the weaponized file.	
Step's Question	Comment on your mission.	
LO L9.1	The learner utilizes a reverse shell.	

	Bloom Level	Application
	NCWF KSA	S0293, K0177, A0058
LO L9.2	The learner applies the stages of the CKC model.	
	Bloom Level	Creation, Application
	NCWF KSA	K0177
Teaching Content	Adopted from the CAPEC's attack pattern *Malicious Logic Insertion* (https://capec.mitre.org/data/definitions/441.html).	
Hints	1. Get a reverse shell to the target with privileged rights. 2. Use a metasploit to connect to the target. 3. Enter the command 'msfconsole' and use the 'run' command to connect to the target with privileged rights.	

Most of the LOs presented in this section belong to the application level of the Bloom taxonomy, as the prototype scenario mainly exercises skills in penetration testing. However, the L4.1 of Step 4, L5.2 of Step 5 and L9.2 of Step 9 belong to the higher COFELET layers because they require deep knowledge and competencies. Specifically, in Step 4, the learner has to appraise which hosts and services are most likely to be vulnerable, otherwise, s/he will end up searching all the services in order to find possible vulnerabilities. In Step 5, the learner has to improvise to exploit the password recovery mechanism after the realization that the guest account is inactive; and in Step 9 the learner creates a plan of an APT attack (creation level) by applying the CKC model (application level). On the contrary, LOs L1.2, L3.2, L4.3, L5.3, L7.2 and L8.2 belong to the low COFELET layers, as the learner usually explains notions. For example, the LO L3.2 is associated with the question '*Is a filtered target port considered opened or closed?*' prompted in Step 3. *Table 8-3* provides a full description of L3.1's rationale and attributes, associated with Step 3.

***Table 8-3**. L3.1 description*

Attribute	Value and Rational
Statement	Learner utilizes network analysis tools (i.e., port scanners) to determine the status of ports on targets'
Type	Skill
Name	PortScanLO_01
Label	Port Scan Learning Objective

Degree	34
Role	‘Penetration tester’, ‘Vulnerability Assessment Analyst’ and ‘Target Network Analyst’. ‘Penetration tester’ role is not included in the NCWF. However, it combines knowledge and competencies of the ‘Vulnerability Assessment Analyst’ and ‘Target Network Analyst’ of NCWF
BloomLevel	Application
Learning Objectives	L1, as the learner has to know how to discover a host before she scans its ports
NCWF KSA	It is based on the S0081 skill of NCWF ‘Skill in the use of penetration testing tools and techniques’ and the S0051 skill of NCWF ‘Skill in using network analysis tools to identify vulnerabilities. (e.g., fuzzing, Nmap, etc.)’
Teaching Content	The material is adopted from the ‘*Port Scanning*’ attack pattern of CAPEC (https://capec.mitre.org/data/definitions/300.html)
Grade Scheme	The Grade scheme is an array of GradeScheme objects as the objects described below: GradeScheme1. “times assessed=‘0-1’, hints=‘0’, time=‘1-50’, actions=‘0-3’, score=‘100’, grade=‘100’ ”. GradeScheme2. “times assessed=‘4-6’, hints=‘1-4’, time=‘1-50’, actions=‘0-3’, score=‘-10’, grade=‘1’ ” The GradeScheme1 object denotes that the first or second time that the LO is achieved the learner will have a grade 100%, if no hints are taken, if the time taken from the last goal is 1 to 50 in seconds and if the learner performs 0 to 3 actions (excluding the task that exercised the LO). The score of 100 denotes that the achievement of the LO adds 34 points (i.e., ‘*Degree*’ attribute of LO object) to the corresponding skill in the profile of the learner (i.e., ‘*LO Degree*’ attribute of role object). The GradeScheme2 object denotes that the learner will have a penalty score of 10 if the 4th to 6th time that the LO is assessed, she will acquire hint(s).
Hints	Hint 1: Text = ‘Identify the open ports of the targets’, time=‘*120*’ seconds

Hint 2: ‘Use a port scanner tool that sends probes to an IP address/port and determines the status of the port’, time=‘*120*’seconds

Hint 3: ‘Scan your target(s) using a port scanner (e.g., Nmap) and the appropriate scan option (e.g., TCP Scan option)’ , time=‘*120*’ seconds

Hint 4: ‘In the terminal issue the command: *nmap –sS |target|*’ , time=‘*120*’seconds

---

8.6. Chapter Conclusion

In this chapter, the design of the HackLearn scenario-based game is demonstrated by utilizing the COFELET framework and the COFELET game life-cycle as guides. The chapter also presents the analysis of HackLearn into its components by employing the methodology of the ATMSG model and the *il Segreto di Arlecchino* prototype scenario along with the scenario’s key features (e.g., scenarios rational, learners’ tasks). Additionally, the chapter presented HackLearn’s gameplay by providing details on the learners’ experiences, the challenges learners face, and the manner they have to utilize cyber security tools, techniques and strategies to achieve the game’s objectives.

9. HackLearn's Implementation

9.1. Introduction

HackLearn has been implemented in the Unity 3D game development engine with C# as the programming language. The game's implementation included the creation of 300 key elements (e.g., tasks, conditions, goals, hints, LOs) implemented in XML and more than 15.000 lines of code. HackLearn interacts with a back-end storage facility (i.e., MySQL database) in which it stores learner's details, the game's learning analytics, the learners' answers to the in-game questions. The game addresses SQL queries to the back end by utilizing PHP scripts that communicate with the MySQL database to retrieve and store the game's data.

HackLearn has been developed by a game developer who has worked on the game's implementation for more than a year. HackLearn's implementation process included the development of a prototype scenario, a MySQL database, and a set of PHP scripts for the communication of the game with the database. The design of HackLearn's interface has been elaborated on by a game designer who has worked on the interface for three months. The key elements of HackLearn's attack patterns have been designed by a cybersecurity specialist and the game's scenario has been created with the collaboration of experienced educators. Although HackLearn was initially implemented as a PC standalone application, due to the COVID-19 virus pandemic it was exported from Unity as a WebGL web application to run in any web browser, anytime and anywhere.

This chapter provides implementation details of the HackLearn game, focusing on the COFELET ontology primary elements (i.e., *Tasks*, *Goals* and *Conditions*), the COFELET ontology key elements (e.g., *LOs*, *Hints*, *Teaching contents*, *SEFs*) and the COFELET *scenarios*. The chapter 9 also covers some challenging implementation features such as the polymorphism of entities and the inheritance in the primary elements and entities; it finally presents implementation details of the back-end storage facility and the PHP scripts which communicate with it.

9.2. Primary Elements XML Nodes

The tasks, the conditions and the goals are called Primary Elements (PEs) as they are the fundamental ingredients of the HackLearn game from which the rest of the elements are constructed (Katsantonis & Mavridis, 2019). The task PEs are in-game actions executed towards the unleashing of a cyber-attack and the fulfillment of the in-game goals (goal PEs). The execution of tasks is only possible when the appropriate

conditions hold (condition PEs). HackLearn interprets PEs as quintuple statements in the form `<subject entity, property, object entity, source entity, destination entity>`. A property denotes that an entity acts on another entity or that an entity has a property. HackLearn stores PEs in XML files (*PEs repositories*) in the form of XML nodes (*PEs nodes*) as displayed in *Figure 9-1*. *Figure 9-1* depicts the ICMP Echo Ping host discovery task (*ping command task*) in the form of an XML node representing the launch of a ping command in the HackLearn's terminal. The ping command task node belongs to the *HostDiscover_cmds* TaskGroup and it contains 5 XML child nodes that are four entities and a property. Specifically, the ping command task XML node contains the entity *Agent* of *Learner* type that denotes the task is initiated by the learner; the *Cmd* entity of type *ICMPEchoPingHostDiscoveryCmd* that specifies the category of the command; the source entity *Host* that represents the learner's host; the destination entity *Network* that indicates the target and its type; and the *entersCmd* property. The ping command task XML node, as well as all the XML nodes of HackLearn, also includes the attributes *id*, *name* and *label*.

```
<?xml version="1.0" encoding="utf-8"?>
<Tasks>
  <TasksGroup id="#" name="HostDiscovery_PortScaning_tasks" label="Host discovery tasks">
    <Task id="13533" name="ICMPEchoPingHostsDiscoveryCmdTask" label="Host scanner's discovery command">
      <Subj>
        <Ent type="Learner" value="@runtime">Agent</Ent>
      </Subj>
      <Property>entersCmd</Property>
      <Obj>
        <Ent type="ICMPEchoPingHostsDiscoveryCmd" value="@runtime">Cmd</Ent>
      </Obj>
      <Src>
        <Ent type="LearnerHost" value="@runtime">Host</Ent>
      </Src>
      <Dst>
        <Ent type="TargetNetwork" value="@runtime">Network</Ent>
      </Dst>
    </Task>
```

***Figure 9-1**. Task node*

HackLearn uses conditions to describe the occurring conditions in the cyberspace of HackLearn, while the set of HackLearn conditions form the cyberspace of the game. *Figure 9-2* depicts the *HostScanner_is_locked* condition node, which indicates that the ping has the property *isLocked* (i.e., ping is locked). Condition nodes are stored in the Conditions.xml repository.

```
<Condition id="36488" name="HostScanner_is_locked" label="Host scanner is locked">
  <Subj>
    <Ent type="HostScanner" value="ping">Tool</Ent>
  </Subj>
  <Property>isLocked</Property>
  <Obj />
  <Src />
  <Dst />
</Condition>
```

***Figure 9-2**. Condition node*

Goal nodes are represented by quintuple statements denoting the achievement that the subject has to accomplish. *Figure 9-3* shows the *PortScanGoal* goal node indicating that the agent finds information such as the state of the target's ports, the services running, the versions of running services etc.

```
<Goal id="20997" name="PortScanGoal" label="Port scan goal" capecID="300" apID="0">
  <Subj>Agent</Subj>
  <Property>finds</Property>
  <Obj>
    <Ent type="TargetPortsInfo" value="@runtime">Info</Ent>
  </Obj>
  <Source />
  <Destination />
  <Relation />
</Goal>
```

***Figure 9-3**. Port Scan Goal node*

The goal nodes differ from the task nodes and the condition nodes, as they include the *capecID* and *apID* attributes and the *Relation* child node. The *capecID* attribute specifies the id of the CAPEC's attack pattern from which the goal is derived. The *apID* specifies the id of the Scenario Execution Flow attack pattern with which the goal is associated. In cases that the goal is not associated with a CAPEC attack pattern or a specific Scenario Execution Flow attack pattern the capecID and apID attributes are set to '0'. The Relation child node is set when the goal node is utilized outside the goals repository (e.g., in scenarios and scenario execution flows) and it specifies the task that makes the goal achieved.

9.3. Entities and Polymorphism

The Ent XML nodes (*entity nodes*) represent the entities that lie in the game context. In HackLearn an entity object instantiates the class specified by the entity node. For example, in the ping command task XML node (depicted in *Figure 9-1*) the *Subj* entity is an entity object that instantiates the *Learner* class. The *type* attribute of the entity node designates the class of the instantiated object. In the presented example, the Learner class is a subclass of the Agent class indicated in the inner text of the entity node. An entity node contains the *value* attribute, a unique value that designates a specific instance for the entity it indicates. The value attribute *@runtime* designates that the entity will have a specific instance during the game-play of the HackLearn. For instance, the *Agent* entity will take its value after the learner logs in the game, while the *Cmd* entity will get its value after the learner enters the command represented by the XML node. In such a way, COFELET games allow describing entities that will be put into effect under different forms and in various contexts implementing a type of ***polymorphism***, a critical feature for the development of effective COFELET games.

The entity classes defined in COFELET games vary in size and complexity. Some classes are simple and small having only a few attributes and limited functionality, whereas some entity classes are extended containing multiple attributes and routines. *Figure 9-4* depicts the *Learner* class, a simple class containing attributes of the learner's details that is instantiated when the learner logs into the system:

```
public class Learner : Agent {
  public ushort ID;
  public Role Role;
  public ushort RoleID;
  public string Profile;

  public Learner(ushort id, Role role, ushort roleid, string nm, string lbl) {
      this.ID = id;
      this.Role = role;
      this.RoleID = roleid;
      this.Name = nm;
      this.Label = lbl;
  }
}
```

***Figure 9-4**. Learner class*

The *Learner* class inherits from the *Agent* class the *Name* and the *Label* attributes. The Name attribute is a unique name in the game's context, and it is related to the value attribute of the corresponding entity node, whereas the Label attribute is a user-friendly name appropriate for displaying in the game's interface.

On the contrary, the *Host* class (*Figure 9-5*) is a complex class containing numerous attributes and methods, and providing advanced functionalities in HackLearn (*Figure 9-5* for brevity depicts only the attributes of the *Host* class). The *Host* class defines and inherits from its superclass more than 60 methods providing various functionalities regarding the network interfaces, the host's file system, the user management and the users rights, the network services and connections and the host's firewall. Moreover, the *LearnerHost* class (i.e., a subclass of the *Host* class) is associated with the *Terminal* class, which encapsulates the appropriate methods for the interpretation and execution of the learner's commands.

```
public class Host : NwEnt {
   public string Arch { get; protected set; }
   public string Platform { get; protected set; }
   public string KernelName { get; set; }
   public string KernelRelease { get; protected set; }
   public string KernelVersion { get; protected set; }
   public string OS { get; protected set; }
   public List<eFile> FileSystem;
   public FileMode[] FileDefaultPerms = new FileMode[] { FileMode.Read_Write, FileMode.Read_Write, FileMode.Read_Write };
   public FileMode[] DirDefaultPerms = new FileMode[] { FileMode.All, FileMode.All, FileMode.All };
   public string UMask = "022";
   public string UMaskSymbolic = "-rw-r--r--";
   public int CureFile { get; protected set; }//holds the current node/path
   protected List<HostUser> Users = new List<HostUser>();
   protected List<HostGroup> Groups = new List<HostGroup>();
   protected HostFirewall Firewall = null;
   protected bool FWExists = false;
   protected List<Connection> Connections = new List<Connection>() { };
   protected List<string[]> Connections_serialized = new List<string[]>() { };
   private List<string[]> LoggedUsersDetails = new List<string[]>() { };
   private byte LoggedUserIndex = 0;
   protected Dictionary<int, Service> OpenPorts = new Dictionary<int, Service>();
   public DateTime Started { get; protected set; }
   protected string BEHostsDir = "";
```

***Figure 9-5**. The Host class*

9.4. Inheritance

HackLearn maintains a large repository of conditions that describe in detail the entities that it employs. *Figure 9-6* displays the *AcceptTCPPacketAll* (*AcceptAll* for brevity) condition XML node (condition node), a condition that represents a loose rule of a firewall that allows all the TCP packets to pass.

```
<ConditionGroup id="360107" name="FWTCPConds" label="Conditions related to Firewalls and TCP">
  <Condition id="36100" name="AcceptTCPPacketAll" label="Firewall Accepts All TCP Packets">
    <Subj>
      <Ent type="All" value="@runtime">Firewall</Ent>
    </Subj>
    <Property>accepts</Property>
    <Obj>
      <Ent type="TCPPacket">
        <Class>Packet</Class>
        <Value>Packet_TCP_All</Value>
      </Ent>
    </Obj>
    <Src>
      <Ent type="All" value="@runtime">Host</Ent>
    </Src>
    <Dst>
      <Ent type="All" value="@runtime">Target</Ent>
    </Dst>
```

***Figure 9-6**. AcceptAll Condition node*

Although the AcceptAll condition node has a similar form with the ping command task node presented in section 9.2 (*Figure 9-1*), it has some differences that is the use of the *All* type and the nesting of the *Class* and the *Value*. The Obj entity of the *AcceptAll* condition (depicted in *Figure 9-6*) is of class *Packet* and it has the *Packet_TCP_All* value indicating that the condition allows all the TCP packets to pass. The type *All* is a wildcard for entity types. In the firewall entity, the attribute *All* indicates that any kind of firewall (e.g., a network firewall or a host firewall such as the iptables in Linux) can make this condition hold, whereas the attribute *All* in the *Src* entity indicate that the host can be any host (i.e., the learner's host or a target host). On the other hand, the nesting in the *Packet* entity is used to facilitate ***inheritance*** in the conditions' repository. A condition inheritance example is demonstrated in *Figure 9-7*, which shows two child conditions of *AcceptAll* condition, the *AcceptTCPPacketAllO2I* (*AcceptAllO2I* for brevity) and the *AcceptTCPSynPacketO2I* (*AcceptSynO2I* for brevity).

```
<Condition id="36101" name="AcceptTCPPacketAllO2I" label="Firewall Accepts All TCP
Packets from outside to inside">
  <Src>
    <Ent type="LearnerHost" value="@runtime">Host</Ent>
  </Src>
  <!--Dst>
    <Ent type="All" value="@runtime">Target</Ent>
  </Dst-->

  <Condition id="36102" name="AcceptTCPSynPacketO2I" label="Firewall Accepts TCP
  SYN Packets from outside to inside">
    <Obj>
      <Ent type="TCPPacket">
        <Class>Packet</Class>
        <Value>Packet_TCP_Flag_SYN</Value>
        <Flag>SYN</Flag>
      </Ent>
    </Obj>
  </Condition>
```

***Figure 9-7**. AcceptAllO2I and AcceptSynO2I Condition nodes*

The former inherits from the *AcceptAll* condition the *Subj* entity, the *Obj* entity, the *Dst* entity and the property *accepts* and it hides (i.e., overrides) the *Src* entity by indicating that it has the *LearnerHost* type. In such a way, the *AcceptAllO2I* condition implements more limited applicability than *AcceptAll* condition, as it specifies that the direction of the TCP packets is initiated from the learner's host (a firewall considers this flow of packets as 'out') towards the system that the firewall operates (i.e., a firewall considers this flow of packets 'in'). The condition inherits the *Dst* entity of *Target* class designating that the destination of the packet can be any network device such as a host, a network or the firewall (e.g., in missions that the learner has to inspect the configuration of a network firewall). The *AcceptSynO2I* condition inherits all the parts of the quintuple from the *AcceptAllO2I* conditions except from the *Obj* entity, which overrides the value of its parent condition by indicating that the *Packet* entity it employs has the *Packet_TCP_Flag_SYN* value (i.e., designated by the Value attribute of the *TCPPacket* object). The *Obj* entity of the *AcceptSynO2I* additionally defines the *Flag* attribute by nesting it in the entity node. The *Flag* attribute indicates that the packet has the SYN Flag set. Game developers and instructors can nest in an entity node any number of additional properties in order to realistically describe the properties of the entities.

The Info entity (shown in *Figure 9-3*) instantiates the *TargetPortsInfo* class (*Figure 9-8*) encapsulating the information acquired by the learner when performing a port scan attack pattern.

```
public class TargetPortsInfo : Info {
    public string HostName;
    public string HostIPv4;
    public string HostHwAddr;
    public new List<TargetPortInfo> Information;

    public TargetPortsInfo() { }

    public TargetPortsInfo(string hn, string ip, string hwa) {
        this.Name = hn + " " + ip + " " + hwa;
        this.Label = "Hostname:'" + hn + "' IP:'" + ip + "' HardwareAddress:'" + hwa + "'";
        Information = new List<TargetPortInfo>();
    }
}
```

***Figure 9-8**. TargetPortsInfo class*

The *TargetPortsInfo* class hides (i.e., overrides) the *Information* attribute which is inherited from the *Info* super class. The Information attribute is a list of *TargetPortInfo* entities in which each element corresponds to an open port in the target of the attack.

9.5. Learning Objectives

HackLearn stores LOs in XML files (LOs repositories) as XML nodes, called LO XML Nodes (LO nodes). *Figure 9-9* presents the "*Learner - abuses - authentication to escalate privileges by following exploit's directions*" LO, illustrating the properties of the LO.

```
<LO id="40707" ord="9" tksa_type="S" name="ApplicationAthenticationAbusePrivileges01" label=
"Authentication abuse to escalate privileges">
  <Quintuple>
    <Subj>Learner</Subj>
    <Property>
      <Imperative>Abuse</Imperative>
      <ThirdPerson>abuses</ThirdPerson>
    </Property>
    <Obj>authentication to escalate privileges by following exploit's directions</Obj>
  </Quintuple>
  <Roles>
    <Role role_id="54405">Penetration Tester</Role>
  </Roles>
  <BloomLevel>Application</BloomLevel>
  <Keywords>authentication abuse, privilege escalation, exploit</Keywords>
  <!-- S0293: Skill in using tools, techniques, and procedures to remotely exploit and establish
  persistence on a target. -->
  <!-- K0177: Knowledge of cyber attack stages (e.g., reconnaissance, scanning, enumeration, gaining
  access, escalation of privileges, maintaining access, network exploitation, covering tracks). -->
  <CSWFKSAIDs>S0293,K0177</CSWFKSAIDs>
  <Prerequisites>
    <!-- LO:40675 Learner explains the concepts of vulnerabilities and exploits -->
    <LO lo_id="40675" ord="1">mandatory</LO>
    <!-- LO:40503 Learner utilizes exploit-db search tool -->
    <LO lo_id="40503" ord="2">optional</LO>
  </Prerequisites>
  <DegreeOfAchievement />
  <DegreeOfAchievementFactor />
</LO>
```

***Figure 9-9**. Learning Objective (LO) node*

The LO has type Skill (denoted by the letter 'S'), it lies in the Application level of the Bloom taxonomy, it is associated with the *Penetration Tester* role and it has a mandatory prerequisite LO and an optional one. The *DegreeOfAchievement* and the *DegreeOfAchievementFactor* are metrics stored in the back end (i.e., HackLearn's database) and in the scenario respectively and for this reason they don't have values in the LO XML node stored in the LOs repository.

The *Authentication abuse to escalate privileges* LO is based on two NCWF KSAs, the "*knowledge of cyber-attack stages (e.g., reconnaissance, scanning, enumeration, gaining access, escalation of privileges, maintaining access, network exploitation, covering tracks)*" and the "*skill in using tools, techniques, and procedures to remotely exploit and establish persistence on a target*". Comparing the presented LO and the related NCWF KSAs, it is noticed that the 'Authentication Abuse to escalate privileges' LO is only merely related to the stated NCWF KSAs. The reason for this is that the presented NCWF KSAs do not have the appropriate form to be adopted in an effective educational approach as they are general and lengthy statements including several cyber security concepts. Besides, an instructor cannot measure the degree to which this knowledge and skills are acquired by the learner. On the contrary in the HackLearn game, the instructor has to efficiently define measurable LOs (Allen & Straub, 2015)

(Katsantonis et al., 2019) that have a clear purpose (Nagarajan et al., 2012), (Compte et al., 2015).

9.6. Teaching Content

Teaching contents are the materials (e.g., text, figures and videos) aiding the learner to reinforce KSAs or assimilate the new knowledge and the new competencies. The HackLearn game has a repository of materials in which teaching content is stored in the form of XML nodes (material nodes). *Figure 9-10* shows the *Authentication Abuse* material associated with the *Authentication abuse to escalate privileges* LO.

```
<TeachingContent id="44705" name="AuthenticationAbuseTC01" label="Authentication
abuse teaching content">
  <Text>
    An attacker obtains unauthorized access to an application, service or device
    either through knowledge of the inherent weaknesses of an authentication
    mechanism, or by exploiting a flaw in the authentication scheme's
    implementation. In such an attack an authentication mechanism is functioning
    but a carefully controlled sequence of events causes the mechanism to grant
    access to the attacker. This attack may exploit assumptions made by the
    target's authentication procedures, such as assumptions regarding trust
    relationships or assumptions regarding the generation of secret values.
  </Text>
  <References>
    <Reference ord="1">CAPEC Authentication Abuse
    (https://capec.mitre.org/data/definitions/114.html)</Reference>
  </References>
  <LO lo_id="40707"></LO>
</TeachingContent>
```

***Figure 9-10**. Authentication Abuse Material node*

The *Authentication Abuse* material provides a short description of the homonymous CAPEC's AP and it provides a reference to the CAPEC's website.

9.7. Scenario Execution Flows

Scenario execution flows (SEFs) contain attack patterns in a form of XML nodes representing the APs as sequences of tasks along with the related conditions and goals. The HackLearn game combines the PEs, the LOs and the hints to form AP elements. HackLearn stores the SEFs in an XML file (i.e., the SEFs repository) as XML nodes (*AP nodes*) organized in groups with respect to the goals they contain. The SEFs repository along with the high-level repositories are developed and maintained by the HackLearn game developers.

```
<AttackPattern id="285" capecID="285" name="ICMP_Echo_Request_Ping" url=
"https://capec.mitre.org/data/definitions/285.html">
  <Label>ICMP Echo Request Ping</Label>
  <Description>An attacker sends out an ICMP Type 8 Echo Request and gets a ICMP Type 0 Echo
  Reply datagram.</Description>

  <Goals>
    <Goal goal_id="20403" capecID="285" name="Host Discovery Goal" label="find alive host(s)">
      <Subj><Ent type="Learner" value="@runtime">Agent</Ent></Subj>
      <Property>finds</Property>
      <Obj><Ent type="All" value="@runtime">Target</Ent></Obj>
      <Src />
      <Dst />
      <Relation>
        <!-- HostScanner, gets, Packet_ICMP_Type_0, Destination -->
        <Task task_id="16904">true</Task>
      </Relation>
    </Goal>
  </Goals>

  <Tasks> . . . </Tasks>
  <Conditions> . . . </Conditions>

  <LOs><LO lo_id="40047" content_id="" type="sef">
      <!-- Learner, utilizes, network analysis tools to identify alive hosts in a network-->
      <DegreeOfAchievementFactor>25</DegreeOfAchievementFactor>
    </LO></LOs>

  <Hints>
    <Hint hint_id="45707" order="1" timer="60">Identify alive hosts on the network.</Hint>
    <Hint hint_id="45708" order="2" timer="60">Use a tool that sends probes to an IP address
    to determine if the host is alive. Your goal is to send a packet through to the IP
    address and get a response from the host. You can start with a range of IP addresses
    belonging to a target network.</Hint>
    <Hint hint_id="45709" order="3" timer="60">Use a tool that sends ICMP datagrams (e.g.,
    Type 8 Echo Request) and solicits response datagrams (e.g., ICMP Type 0 Echo Reply)</Hint>
    <Hint hint_id="45710" order="4" timer="60">Ping your target (e.g., 'ping -b |target|' or
    'nmap -PE |target|')</Hint>
  </Hints>
</AttackPattern>

<AttackPattern id="299" capecID="299" name="TCP_SYN_Ping" url=
"https://capec.mitre.org/data/definitions/299.html">...<AttackPattern>
<AttackPattern id="297" capecID="297" name="TCP_ACK_Ping" url=
"https://capec.mitre.org/data/definitions/297.html">...<AttackPattern>
```

***Figure 9-11**. ICMP Echo Request Ping AP node*

Figure 9-11 displays the *ICMP Echo Request Ping* (Ping) AP node of the host discovery APs group. The host discovery APs group also contains the ping AP node's sibling attack patterns, the *TCP SYN Ping*, and the *TCP ACK Ping*. Sibling APs are the APs belonging to the same group of attack patterns that share the same goals. The SEF's AP node contains a set of properties and child nodes.

The goal node of a SEF contains the Relation attribute which designates that the goal is related with the task *<the host scanner tool, gets, a Packet_ICMP_Type_0, from the destination*. The execution of this task makes the SEF's goal to be considered achieved by HackLearn. The Relation nodes include the *id* attribute that holds the id value of the associated task.

The *Tasks* node contains a set of TaskNodes (stated as Tasks for brevity) which describe the AP's tasks (*Figure 9-12*). TaskNodes relate to condition nodes to indicate which condition is the prerequisite for the completion of the task. TaskNode's Relation nodes

nest a condition node containing the id of the condition and the value *true*, specifying that the condition must hold, or the word *false* specifying that the condition must not hold. The TaskNode node also includes the *nextID*, the *nextOrd*, the *type*, and the *interval* attributes (*Figure 9-12*). The *order* attribute denotes the position of the TaskNode in the sequence of tasks in the AP, the *nextID* attribute specifies the subsequent task, *nextOrd* specifies the order of the subsequent task, the *interval* attribute indicates the time period that a task requires to execute, and the type attribute designates the type of the task such as the console command, the user interface task and auto-task performed by the game's engine. The *nextID* and *nextOrd* are mutually exclusive attributes, as *nextID* points to the next TaskNode with the specified id, whereas the *nextOrd* attribute points to a group of TaskNodes having the specified order.

```
<Tasks>
    <Task task_id="13533  " order="1" nextOrd="2" nextID="10454" duration="0" type="ConsoleCmd">
      <Obj>
        <Ent type="ICMPEchoPingHostsDiscoveryCmd" value="@runtime">Cmd</Ent>
      </Obj>
      <Relation>
        <!-- Learner, knows, NetworkInfo-->
        <Condition cond_id="33597">true</Condition>
        <!--Learner, hasAccess, HostScanner-->
        <Condition cond_id="34618">true</Condition>
      </Relation>
    </Task>

    <!-- HostScanner, crafts, Packet_ICMP_Type_8 -->
    <Task task_id="10454" order="2" nextOrd="3" nextID="10455" duration="1" type="Auto">
    </Task>

    <!--Tool, displays, ConsoleText:"Tool is locked"-->
    <Task task_id="13861" order="2" nextOrd="0" nextID="0" duration="1" type="Auto">...</Task>

    <!-- HostScanner, sends, Packet_ICMP_Type_8, LearnerHost, Target-->
    <Task task_id="10455" order="3" nextOrd="4" nextID="0" duration="1" type="Auto">
      <Dst>
        <Ent type="TargetNetwork" value="@runtime">Network</Ent>
      </Dst>
      <Relation>
        <!-- Firewall, accepts, packet ICMP type 8, All, Target-->
        <Condition cond_id="36004">true</Condition>
      </Relation>
    </Task>

    <!--HostScanner, gets, Packet_ICMP_Type_0, Target, LearnerHost-->
    <Task task_id="16904" order="4" nextOrd="0" nextID="0" duration="3" type="Auto">...</Task>

    <!--HostScanner, gets, Packet_null-->
    <Task task_id="16977" order="4" nextOrd="0" nextID="0" duration="5" type="Auto">...</Task>
  </Tasks>
```

***Figure 9-12**. ICMP Echo Request Ping AP Tasks*

As shown in *Figure 9-12*, AP nodes do not include full definitions of the PEs nodes and the LO nodes they contain. The reason for this is that HackLearn omits the definitions of entity nodes, when they are identical to the entity nodes of the original elements stored in the corresponding repositories (i.e., the PEs repositories and LO repository). For example, in the ping AP presented in *Figure 9-11*, the ping command task node (i.e., the first task of the AP presented in *Figure 9-12*) and the LO node do not contain definitions for the *Subj*, *Obj*, *Src* and *Dst* entities. Besides, the ping command task node

and the LO node instead of the *id* attribute define the *task_id* and the *lo_id* attributes respectively to denote that they point to the original nodes stored in the repositories (i.e., the task repository and LO repository).

The AP nodes include a sequence of Hint XML nodes (hint nodes) containing the necessary information to scaffold learners' efforts. The hint nodes contain the text presented to the user, the order denoting the position of the hint in the sequence, and the timer attribute indicating a time period after which the learner is notified with the hint. The specified time period corresponds to the time elapsed since the presentation of the last hint of the current AP. In cases that the hint to be presented is the first in order, the time period counts from the time elapsed since the achievement of the last goal. Subsequently, in cases the AP to be performed is the first in the current session, the time period counts from the start of the current session.

9.8. Repository of Entities

The HackLearn game contains a repository of entities which is an XML file containing descriptions of entities in terms of their types and attributes. The entities repository contains numerous entities including tools, hosts, network interfaces, files, directories, file-systems, networks, firewalls, services and Metasploit modules. The rationale of the entities' repository is to provide a collection of predefined entities aiding instructors and game developers to build a cyber space easily and efficiently.

Figure 9-13 presents the description of the cat tool entity when utilized as a file printer, which shows the contents of a file on the console.

```
<Tool id="6014" name="cat" type="FilePrinter,FileCreator,FileJoiner" label="File Utility">
  <Description>Displays files' contents on the standard outputs and/or concatenates them.
  </Description>
  <ShortDescription>Displays files contents</ShortDescription>
  <Synopsis aggregation="true">[Options] {eFile}</Synopsis>
  <Input id="1" name="eFile" description="obligatory">eFile</Input>
  <!--Cat_FilePrintingCmds-->
  <Cmds name="Cat_FilePrintingCmds" label="Cat File Printer Cmds">
    <Cmd id="6015" name="Cat_FilePrintingCmd" type="FilePrintingCmd" label="File Printer Cmd">
      <Option id="cat_null" name="cat_null" label="File Printing">
        <Literal>null</Literal>
        <Verbose />
        <InputSpecification id="1" description="obligatory" type="RegularFile" value="@runtime"
        >eFile</InputSpecification>
      </Option>
      <!-- Agent, enters, FilePrintingCmd -->
      <Task task_id="12252">
        <Subj>
          <Ent type="Learner" value="@runtime">Agent</Ent>
        </Subj>
        <Obj>
          <Ent type="FilePrintingCmd" value="Cat_FilePrintingCmd">
            <Class>Cmd</Class>
            <Ent type="FilePrinter" value="cat">Tool</Ent>
            <Ent type="FileNameInfo" value="@runtime">Info</Ent>
            <Ent type="RegularFile" value="@runtime">eFile</Ent>
            <Ent type="FileContentsInfo" value="@runtime">Info</Ent>
          </Ent>
        </Obj>
      </Task>
    </Cmd>
  </Cmds>
  <!--Cat_FileCreationCmds-->
  <Cmds name="Cat_FileCreationCmds" label="Cat File Creation Cmds">
    <Cmd id="6020" name="Cat_FileCreationCmd" type="FileCreationCmd" label="File Creation Cmd">
      <Option id="cat_null" name="cat_null" label="File creation option">
```

***Figure 9-13**. The cat tool entity*

The cat tool description contains an explanation of cat's operation and syntaxis, the number and the type of inputs it takes, the commands it issues, the tasks associated with its commands and the entities description of the associated tasks. The tasks' attributes of the entities repository override the attributes of the tasks defined in the tasks repository. For example, *Figure 9-13* shows that the cat file printing command is associated with the <Agents, enters, FilePrintingCmd> task, which includes multiple entities including the *tool* entity with its value set to *cat* and the *Cmd* entity with its value set to *Cat_FilePrintingCmd*. The *Info* and the *eFile* entities are the input and output entities HackLearn will manipulate to simulate the operation of the cat tool. Specifically, the *FileNameInfo* entity is the name of the file learner enters in the console, the *eFile* entity is the actual file related with the name provided and the *FileContentsInfo* is the list of the file's contents outputted in the game's console.

9.9. Scenarios

Scenarios are developed by the instructors in cooperation with the COFELET game developers and they are stored in XML files. Scenarios contain the appropriate information for the setup and run of a game session, and they consist of three main parts.

The first part contains the scenario's details such as the name, the label, the mission's description, the difficulty level, the objectives, and the creators (*Figure 9-14*).

```
<?xml version="1.0" encoding="utf-8"?>
<Scenarios xmlns:S="http://www.w3.org/TR/html4/">
  <Scenario id="1" name="HarlequinsSecret_A" label="il Segreto di Arlecchino">
    <Description>Description: In the ArtComedy scenario you are provided with a terminal and tools. Your
    objectives include the discovery of potential targets and their vulnerabilities and the exploitation of
    these vulnerabilities to discover the secret flag. In the ArtComedy scenario you will have to apply the
    attack patters of 'Host Discovery', 'Port Scanning', 'Password Recovery Exploitation', 'Authentication
    Abuse', 'Using Malicious Files' and 'Malicious Logic Insertion'. You will also need to search exploits in
    the exploit-db, search google and create weaponized files.</Description>
    <Objective>Find the secret of Harlequin in the scenario's flag.
    </Objective>
    <Reward>Harlequin's Secret</Reward>
    <DifficultyLevel>Harlequin's Millions</DifficultyLevel>
    <Creator>Menelaos Katsantonis, Ioannis Mavridis</Creator>
```

***Figure 9-14**. Scenario's Details node*

The second part (i.e., cyberspace) contains a set of entities and conditions (scenario's preconditions) describing the cyberspace of the game. The cyberspace contains a description of the learner's host entity (i.e., the in-game host used by the learner), a definition of the game's network(s) and the network's components including target hosts, network devices, DNS servers, etc. (*Figure 9-15*).

```
<Entities>
  <LearnerHost preset_id="8734">. . .</LearnerHost>
  <Network preset_id="1857" name="VictoryBall">
    <IPv4 rand_field3="21-201">192.168.77.0</IPv4>
    <Mask>24</Mask>
    <BCast>192.168.77.255</BCast>
    <NetworkFirewall preset_id="7897" name="VictoryBall Firewall">
      <IPv4>192.168.77.17</IPv4>
    </NetworkFirewall>
    <TargetHost preset_id="1553">
      <name>Harlequin</name>
      <Interface preset_id="723">. . .</Interface>
      <HostGroup id="55050">. . .</HostGroup>
      <HostUser id="55200" state="logged">
        <Username>Harlequin</Username>
        . . .
      </HostUser>
      <HostUser id="55250">
        <Username>Calmo</Username>
        . . .
      </HostUser>
      <FileSystem>
        <UMask>066</UMask>
        <Directories>
          <Directory name="kitchen" parent="/">. . .</Directory>
        </Directories>
        <Files>. . .</Files>
      </FileSystem>
      <Service preset_id="56592">. . .</Service>
      <Service preset_id="56862">. . .</Service>
    </TargetHost>
    <Host preset_id="1553">. . .</Host>
  </Network>
  <DNS>. . .</DNS>
  <Tools>. . .</Tools>
  <MetasploitModules>. . .</MetasploitModules>
</Entities>
```

***Figure 9-15**. Scenario's Entities node*

The third part contains a sequence of steps corresponding to the stages of the mission (*Figure 9-16*).

```
<Step id="1" name="Local_IP_Discovery" label="Local IP discovery" order="1" group_index="1">. . .</Step>
<Step id="2" name="HostDiscovery_Step" label="Host Discovery" order="2" group_index="2">. . .</Step>
<Step id="3" name="PortScan_step" order="3" group_index="3">. . .</Step>
<Step id="4" name="FindExploits" order="4" group_index="4">. . .</Step>
<Step id="5" name="FindPasswordRecoveryHintInfo" order="4" group_index="5">. . .</Step>
<Step id="6" name="AbuseAuthenticationFtpMechanism" order="5" group_index="6">. . .</Step>
<Step id="7" name="CreateWeapon" order="6" group_index="7">. . .</Step>
<Step id="8" name="UploadPayloadFile" order="7" group_index="8">. . .</Step>
<Step id="9" name="Backdoor_utilization" order="8" group index="9" >. . .</Step>
```

***Figure 9-16**. Scenario's Steps node*

Figure 9-17 shows the *LearnerHost* XML node (learner host node). Learner hosts are sub-entities of the hosts. Host nodes are composite XML structures encapsulating several entity nodes such as the Interface, the HostUser, the HostGroup, Directories, and Files.

```
<LearnerHost preset_id="8734">
  <name>Magnifico</name>
  <Interface preset_id="723">
    <IPv4>192.168.77.23</IPv4>
    <Mask>24</Mask>
  </Interface>
  <HostGroup preset_id="1020">
    <Groupname>learners</Groupname>
     . . .
  </HostGroup>
  <HostUser state="logged">
    <Username>patrick</Username>
     . . .
  </HostUser>
  <FileSystem>
    <UMask>002</UMask>
    <Directories />
    <Files>
      <File name="template.exe" parent="/">
        <BackEndDir>Magnifico</BackEndDir>
        <FileType>ExcutableFile</FileType>
        <OwnerHostUserID>55101</OwnerHostUserID>
        <Size>789</Size>
        <FilesCreationUMask>002</FilesCreationUMask>
        <Contents>
          <Line id="1">□□□`□□1□d□P0□R</Line>
        </Contents>
      </File>
    </Files>
  </FileSystem>
</LearnerHost>
```

***Figure 9-17**. Learner's Host node*

The learner host presented in *Figure 9-17* node utilizes the preset host node depicted in *Figure 9-18*. Preset nodes are predefined nodes stored in the entities' repository utilized by HackLearn's game engine along with the scenario's nodes to describe the game's entities. Preset nodes make the scenario creation process more efficient and convenient, as the game developer does not have to define entities from scratch.

```
<Host id="8734" name="presetHostName8734" label="Host8734">
  <name>snf-8734</name>
  <Arch>x64</Arch>
  <Platform>x86_64</Platform>
  <KernelName>Edux</KernelName>
  <KernelRelease>0.17.0-37-generic</KernelRelease>
  <KernelVersion>#50~16.04.1-COFELET SMP</KernelVersion>
  <OS>GNU/Linux</OS>
  <HostGroup id="55002"> . . . </HostGroup>
  <HostUser id="55103"> . . . </HostUser>
  <path>path preset</path>
  <Interface id="723">
    <Ethn>eth0</Ethn>
    <LinkEncap>Ethernet</LinkEncap>
    <Mac>aa-0c-f4-90-9c-ef</Mac>
    <IPv4>83.212.102.165</IPv4>
    <BCast>83.212.102.255</BCast>
    <Mask>24</Mask>
    <IPv6>2001:648:2ffc:1225:a800:4ff:fe3c:32a</IPv6>
    <Mtu>1500</Mtu>
    <Scope>Global</Scope>
    <NwAddr>83.212.102.0/24</NwAddr>
  </Interface>
  <FileSystem> . . . </FileSystem>
</Host>
```

***Figure 9-18**. Preset Host node*

HackLearn's scenarios also define the network nodes, an example of which is the VictoryBall network node presented in *Figure 9-19*. The VictoryBall network node is a composite node encapsulating several entities such as hosts, network firewalls and servers (e.g., DNS server). When defining a scenario, the instructor has the option to utilize the randomization feature game's entities. For example, the instructor can randomize the IP address of the network by specifying the *rand_field* attribute in the IPv4 child node of the network node. The network node displayed in *Figure 9-19*, contains the *rand_field3* attribute denoting that the last part of the network's IP address takes a random value from the range of 20 το 200. Randomization can also be used in the file names, user ids, group ids, etc.

```
<Network preset_id="1857" name="VictoryBall">
  <IPv4 rand_field3="20-200">192.168.77.0</IPv4>
  <Mask>24</Mask>
  <BCast>192.168.77.255</BCast>
  <NetworkFirewall preset_id="7897" name="VictoryBall Firewall">
    <IPv4>192.168.77.17</IPv4>
  </NetworkFirewall>
   <TargetHost preset_id="1553">. . .<TargetHost>
  <Host preset_id="1553">. . .</Host>
</Network>
```

***Figure 9-19**. Scenario's Network node*

HackLearn's scenario sets the tools and the tools' commands that will be available to the learner by specifying the tool nodes (*Figure 9-20*). The nodes specified in this part reference the tool and the command entities stored in the entity repository. When learners try to use a tool that is not listed in the game's scenario, they will get a warning message informing the learner that the requested tool is not available in the current scenario.

```
<Tools>
  <Tool tool_id="6840" tool_name="nmap">
    <Cmds type="HostScanner">
      <Cmd cmd_id="6841" cmd_name="Nmap_ICMPEchoPingHostDiscoveryCmd"></Cmd><!-- PE -->
      <Cmd cmd_id="6844" cmd_name="Nmap_TCPSynPingHostDiscoveryCmd"></Cmd><!-- PS -->
      <Cmd cmd_id="6845" cmd_name="Nmap_TCPAckPingHostDiscoveryCmd"></Cmd><!-- PA -->
    </Cmds>
    <Cmds type="PortScanner">
      <Cmd cmd_id="6853" cmd_name="Nmap_TCPSynScanPortScanCmd"></Cmd>
    </Cmds>
  </Tool>
  . . .
</Tools>
```

***Figure 9-20**. Scenario's Tools node*

The third part of the scenario consists of a sequence of consecutive steps. Each step contains the *order* attribute, the *group_index* attribute, a goal node, and a list of condition nodes (i.e., step's preconditions and step's post-conditions). The *order* attribute denotes the step's position in the sequence of steps of the scenario, whereas the *group_index* attribute specifies the *id* of the group that the step belongs to. Steps having the same *order* value and different *group_index* values can be executed in any order. On the contrary, steps having the same *order* and *group_index* values are disjoint,

as the learner can execute one step of the group and then proceed to the next group of steps (i.e., the steps with the subsequent *order* value). In such a manner the HackLearn provides the feature to implement steps' conjunction and disjunction and create sequences of steps in which learners will follow different paths. To accomplish the scenario's mission, the learner has to carry out the scenario's steps until the step with the most advanced *order* value (*Figure 9-16*).

Steps are fulfilled by achieving their goals. To achieve a step's goal, learners have to evaluate the occurring conditions (i.e., scenario and step preconditions). Subsequently, the learner utilizes the available tools to perform the proper sequence of tasks, as prescribed in the correlated AP(s) defined in the SEFs repository. An AP is correlated with a step when it has the same goal as the scenario's step. During the game-play, HackLearn's game engine adopts all the elements of the correlated AP(s), apart from the conditions (i.e., the tasks, the LOs, and the hints). The conditions of a correlated AP are defined by the instructor in the scenario's preconditions, so that the AP will be achievable.

HackLearn provides several capabilities in tuning scenarios' attributes to define an effective learning approach. In case the instructor needs to define additional tasks for the fulfillment of a goal, HackLearn provides the convenience of defining genuine AP nodes in scenario's steps with the appropriate nodes (i.e., PEs, LOs, and hints). Moreover, HackLearn allows the instructor to define new LOs and hints that will complement or override the corresponding nodes in the correlated AP(s). Besides, the instructor can define new XML nodes that will trigger new functionalities. For example, the instructor can define a Question XML node to make the game prompt a question when the learner enters or finishes a step (*Figure 9-21*). Question nodes contain the *type* attribute denoting the question's type (i.e., short answer, multiple-choice or true/false), the *lo_ids* attribute denoting the association of the question with the related LO(s) and the *emergence* attribute denoting the time of the question will emerge (i.e., the beginning or the end of a step).

```
<Questions>
  <Question quest_id="43002" lo_id="40760" type="ShortAnswer" emergence="pre">Did you try to ICMP ping your target (e.g., with
  ICMP Type 8 Echo Request datagrams) with an appropriate tool and option (e.g., 'ping -b |target|' or 'nmap -PE |target|')?
  Did it work or do you thing it would have worked? Explain why.</Question>
</Questions>
```

***Figure 9-21**. Question node*

Moreover, the instructor can specify a message entity, which represents the email message that the learner will receive in his/her in-game inbox (*Figure 9-22*).

```
<Message>
  <Sender>Mentor</Sender>
  <Subject>Take a look at this</Subject>
  <Datetime>12/2/2021</Datetime>
  <Body>Please find your details in the www.hacklearn.org</Body>
</Message>
```

***Figure 9-22**. Message Node*

A message consists of the message's body, the sender details, the subject and the date of sending. The instructor has the capability of using HTML-like tags to markup the message's body text or enter links.

The instructor can also specify a grading scheme (presented in *6.9*) according to which the HackLearn game assesses the learner's efforts (*Figure 9-23*).

```
<LOs>
  <LO lo_id="40426">
    <DegreeOfAchievementFactor>50</DegreeOfAchievementFactor>
    <GradeSchemes>
      <GradeScheme assessed="0-1" hints="0" time="1-50" actions="0-3" math="Add" score="100">100</GradeScheme>
      <GradeScheme assessed="0-1" hints="1-3" time="1-50" actions="0-3" math="Add" score="80">80</GradeScheme>
      <GradeScheme assessed="2-100" hints="0" time="1-50" actions="0-3" math="Add" score="50">50</GradeScheme>
      <GradeScheme assessed="2-100" hints="1-3" time="1-50" actions="0-3" math="Subtract" score="10">30</GradeScheme>
    </GradeSchemes>
  </LO>
</LOs>
```

***Figure 9-23**. Grading scheme node*

HackLearn's grading scheme is defined as an XML node (grading scheme node) nested in the LO nodes of steps along with the *DegreeOfAchievementFactor* metric (*Figure 9-23*). The grading scheme consists of a list of rubric nodes that utilize the properties listed in *Table 8-3. L3.1 description*.

9.10. Tools

COFELET games contain tools entities (tools) used by the learners as means to achieve the game's goals. Tools imitate the operations of real tools and for this reason, they contain advanced functionalities. HackLearn implements several tools counterfeiting the operations of linux-like tools that are accessible through the game's terminal. When the learner enters a command in the terminal, the corresponding homonymous function is called. The tool function is attached to the learner host and it handles the learners' commands as follows:

1. Checks the availability of the tool and the command entities in cyberspace. In case that any of these entities is not available, the learner is informed, and the function exits.
2. Reads the inputs and options of the command, it verifies the options and the command's syntax, and it identifies the class and the type of the input(s).
3. Verifies the command's input(s). For example, if the input specifies the IP of a host, the tool function checks that the IP is valid and that the host exists in the cyberspace. If the input is not valid or it does not exist in the cyberspace then the tool function provides feedback to the learner, it reports the learner's action to the *Instructor* component (depicted in *Figure 7-1*) and it ceases its operation.

4. Forms the requested task (i.e., TaskNode), identifies the involved entities in the cyberspace, and passes the task to the *Task Engine* component (depicted in *Figure 7-1*).
5. Gets the result from the *Task Engine* component. If the result of the task's execution is successful, it performs the task in the cyberspace. Otherwise, it provides feedback to the learner, it reports the learner's action to the *Instructor* component, and it ceases its operation.
6. Checks the result of the task execution in the cyberspace and provides feedback to the learner.

9.11. Back-end

COFELET games use a back-end facility to store the information about the learners (e.g., profile details) and all the evidence to assist the *Instructor* component in assessing the learner's performance. The HackLearn game uses a MySQL database (the HackLearn database) as the back-end facility. The HackLearn database primarily defines two sets of tables regarding the actions of the learners and the details of the learners. The first set of tables includes the tables *user_actions*, *task_traces* and *lo_assessed*. The table *user_actions* (*Figure 9-24*) store all the learner's actions regardless of whether they provoke execution of a task in the Task Engine component. The *Instruction* column of the *user_actions* table stores the learner's command in the terminal or a description of the user actions in the game's interface, whereas the *Result* columns store the result of the learner's action (e.g., 'Task Initiated', 'Goal Reached', 'Session Started' etc.).

	#	Όνομα	Τύπος	Σύνθεση	Χαρακτηριστικά	Κενό	Προεπιλογή	Πρόσθετα
☐	1	**ID** 🔑	mediumint(8)		UNSIGNED	Όχι	*Καμία*	AUTO_INCREMENT
☐	2	**UserID**	smallint(5)		UNSIGNED	Όχι	*Καμία*	
☐	3	**SessionID**	varchar(11)	utf8_unicode_ci		Όχι	*Καμία*	
☐	4	**Instruction**	varchar(255)	utf8_unicode_ci		Όχι	*Καμία*	
☐	5	**Datetime**	datetime			Όχι	*Καμία*	
☐	6	**Result**	varchar(192)	utf8_unicode_ci		Όχι	*Καμία*	

***Figure 9-24**. User_actions SQL table*

The table *task_traces* stores the tasks initiated by the learner's actions identified by the foreign key *UserActionsID* (i.e., the field that matches the primary key of the *user_actions* table). The table *task_traces* stores all the information regarding the task, the SEF, and the scenario of the task, the tool utilized to perform the task and exercised LOs.

#	Όνομα	Τύπος	Σύνθεση	Χαρακτηριστικά	Κενό	Προεπιλογή	Πρόσθετα
1	**ID**	mediumint(8)		UNSIGNED	Όχι	*Καμία*	AUTO_INCREMENT
2	**UsersActionsID**	mediumint(8)			Όχι	*Καμία*	
3	**TaskID**	smallint(5)		UNSIGNED	Όχι	*Καμία*	
4	**SefID**	smallint(5)		UNSIGNED	Όχι	*Καμία*	
5	**ScenarioID**	smallint(5)		UNSIGNED	Όχι	*Καμία*	
6	**ToolID**	smallint(5)		UNSIGNED	Όχι	*Καμία*	
7	**ToolOptions**	varchar(128)	utf8_unicode_ci		Όχι	*Καμία*	
8	**ExercisedLoIDs**	varchar(128)	utf8_unicode_ci		Όχι	*Καμία*	

***Figure 9-25**. Task_traces SQL table*

The *lo_assessed* table is related to the *task_traces* table and stores the grade which denotes the degree that the exercised LOs have been reached. The *lo_assessed* table also stores the details related to the grade such as the number of hints required to help the learner to perform the task, the available hints, the time period taken to complete the specified task, the number of actions the learner performed since the last assessment or the start of the session.

#	Όνομα	Τύπος	Σύνθεση	Χαρακτηριστικά	Κενό	Προεπιλογή	Πρόσθετα
1	**ID**	mediumint(9)			Όχι	*Καμία*	AUTO_INCREMENT
2	**TaskTraceID**	mediumint(8)			Όχι	*Καμία*	
3	**LoID**	smallint(5)		UNSIGNED	Όχι	*Καμία*	
4	**HintsTaken**	tinyint(1)		UNSIGNED	Όχι	0	
5	**HintsAvailable**	tinyint(1)		UNSIGNED	Όχι	4	
6	**TimeFromPreviousTask**	smallint(5)		UNSIGNED	Όχι	0	
7	**NumOfActionFromPreviousSuccess**	smallint(5)		UNSIGNED	Όχι	0	
8	**Grade**	tinyint(3)			Όχι	*Καμία*	

***Figure 9-26**. Lo_assessed SQL table*

The second set of tables include the tables *users* and *users_roles_los* (*Figure 9-27*). The table *users* contains the details of the learner (e.g., username, name, year of birth, password, etc.), whereas the *users_roles_los* table matches the LOs that the learner has to fulfill with the *DegreeOfAchievement* metric included in the table. The *users_roles_los* contains the foreign keys *UserID*, *RoleID*, and *LoID* and the *LastUpdate* field denoting the timestamp of the last time that the *DegreeOfAchievement* metric was informed.

#	Όνομα	Τύπος	Σύνθεση	Χαρακτηριστικά	Κενό	Προεπιλογή	Πρόσθετα
1	**UserID**	smallint(5)		UNSIGNED	Όχι	*Καμία*	
2	**RoleID**	smallint(5)		UNSIGNED	Όχι	*Καμία*	
3	**LoID**	smallint(5)		UNSIGNED	Όχι	*Καμία*	
4	**DegreeOfAchievement**	smallint(3)		UNSIGNED	Όχι	*Καμία*	
5	**LastUpdate**	datetime			Όχι	*Καμία*	

***Figure 9-27**. Users_roles_los SQL Table*

#	Όνομα	Τύπος	Σύνθεση	Χαρακτηριστικά	Κενό	Προεπιλογή	Πρόσθετα
1	**ID**	mediumint(8)		UNSIGNED	Όχι	*Καμία*	AUTO_INCREMENT
2	**QuestionID**	smallint(5)		UNSIGNED	Όχι	*Καμία*	
3	**Text**	text	utf8_unicode_ci		Όχι	*Καμία*	
4	**Datetime**	datetime			Όχι	*Καμία*	
5	**Result**	tinyint(1)			Ναι	*NULL*	

***Figure 9-28**. Answers SQL table*

To cooperate with the backend, HackLearn creates requests to PHP scripts stored in the game's server. The PHP scripts contain the SQL queries used to cooperate with the HackLearn database. For example, at the end of each step the Instructor component calls the script *get_times_lo_assessed.php* to look up in the learner's history how many times the achieved LO(s) has been assessed. The *get_times_lo_assessed.php* script (*Figure 9-29*) calls the *conn.php* script to connect to the HackLearn database; it reads the input of the game's POST request into the *$UserID*, *$LoID* and *$UserAction_Result* PHP variables; it forms and queries the SQL statement using inner join to retrieve data from the tables *user_actions*, *task_traces* and *lo_assessed*; and finally it outputs the result that is retrieved from the Instructor facade.

```php
<?php
    require_once("conn.php");
    //************ Init Vars ************//
    $row_cnt = 0;
    $UserID = 1;
    $LoID = "40426";
    $UserAction_Result = "Goal Reached"
    //************ Read data ************//
    if ( $_SERVER['REQUEST_METHOD'] == 'POST') {
        $UserID = $_POST["UserID"];
        $LoID = $_POST["LoID"];
        $UserAction_Result = $_POST["UserAction_Result"];
    }
    //************ SQL ************//
        $usersloids_squery_str = "SELECT lo_assessed.ID, lo_assessed.TaskTraceID, lo_assessed.LoID,
        lo_assessed.Grade, task_traces.TaskID, users_actions.Datetime, task_traces.UsersActionsID FROM
        lo_assessed INNER JOIN task_traces ON lo_assessed.TaskTraceID=task_traces.ID INNER JOIN users_actions
        ON task_traces.UsersActionsID=users_actions.ID WHERE lo_assessed.LoID='$LoID' AND
        users_actions.UserID='$UserID' AND users_actions.Result='$UserAction_Result'";

        $usersloids_squery = mysqli_query($dbConnected, $usersloids_squery_str  )
            or die ("<br>Πρόβλημα ανάγνωσης στη ΒΔ: "  . mysqli_error($dbConnected) );

        $row_cnt = $usersloids_squery->num_rows;
    //************ Output ************//
        echo "$row_cnt";
?>
```

***Figure 9-29**. Get_times_lo_assessed script*

9.12. Chapter Conclusion

COFELET game scenarios are composite elements, which combine the COFELET ontology elements, preset entities, and the custom elements defined by the instructors. HackLearn elements are built in a such way that can be shared across multiple games and cyber security learning approaches. Besides, instructors and game developers are facilitated as they do not have to create COFELET scenarios from scratch. However, this feature adds complexity to the implementation of the COFELET games engine. For example, HackLearn's game engine creates the learner's host entities following the subsequent procedure: it loads the LearnerHost node from the scenario and it reads the preset_id attribute. Then, it loads the preset host node from the repository of entities, and it enumerates the host's attributes. For each attribute, HackLearn's game engine inspects the attributes defined in the scenario's LearnerHost node. Each attribute defined in the LearnerHost node overrides the corresponding attributes defined in the preset host node. In this example, the HackLearn's instructor effortlessly defines a target host node including the host's users, user groups, and files (e.g., the 'flag.txt' text file) and defines user rights for this file. Thus, instructors are facilitated in creating scenarios and describing the game's entities in detail to create realistic game scenarios, but at the cost of implementation complexity.

10. HACKLEARN'S EVALUATION

10.1. Introduction

In this chapter, the evaluation of the HackLearn game is presented. Two evaluations were performed during the 2nd and the 3rd iteration of phase 4 (*Figure 1-1. The methodology applied in the* dissertation). In the first evaluation, the design of HackLearn is put on the test of a preliminary evaluation scheme elaborated for the assessment of new cyber security game-based learning approaches and live competitions. The employed preliminary evaluation scheme is based on the evaluation scheme presented in chapter 4 and the key characteristics of cybersecurity game-based learning presented in chapter 5. In the preliminary evaluation, HackLearn is resolved into its elements, and its pedagogical effectiveness is appreciated by comparing HackLearn's characteristics with characteristics of the concept map of cyber-security game-based approaches key elements (*Figure 3-1*), the concept map of live competitions technological and pedagogical characteristics (*Figure 4-1*), and the issues and challenges it tries to confront (presented in *4.4 Identified Problems and Issues* and *5.2 Issues & Challenges of Cyber Security Education*).

The second evaluation focused on assessing the user experience perceived by HackLearn's users in the real educational environment of the University of Macedonia class. The process of adopting HackLearn in a real educational environment based on the didactic framework for simulation games is described along with the evaluation methodology elaborated, which is based on the serious games' quality characteristics framework.

10.2. Preliminary evaluation

The evaluation of the presented HackLearn's design is based on the analysis and evaluation scheme proposed in (Katsantonis et al., 2017a) for conducting preliminary evaluations on new live competition approaches. Specifically, the evaluation scheme employs a concept map of game-based learning approaches key elements (GBL concept map) depicted in *Figure 3-1* and categorization of challenges as an assessment tool for the deduction of assumptions regarding the feasibility and the educational impact of new game-based learning and training approaches as well as the effectiveness of these approaches in coping with the identified challenges.

As HackLearn draws many elements from the domain of live competitions, the evaluation scheme also utilizes the concept map of live competitions' technological and

pedagogical characteristics (CtFs concept map) depicted in *Figure 4-1*. Additionally, it employs the identified problems and issues of the field presented in (Katsantonis et al., 2017a) and in (Katsantonis et al., 2019).

Particularly, the GBL concept map has the main role in the evaluation process as it encompasses the characteristics of cyber security game-based learning approaches found in the literature; the CtFs concept map has a secondary role because only the *Pedagogical Benefits* and the *Assessment* segments (*Figure 4-1*) are utilized as consistent with the COFELET framework.

10.2.1. Results

The results of HackLearn's evaluation are presented in *Table* **10-1**. The segments and the characteristics listed in *Table* **10-1** have subscripts indicating the concept map they are adopted from (i.e., GBL and CTF, accordingly). The column Support specifies whether the characteristic is supported (symbol '✔'), not supported (symbol '✕'), or merely supported ('?'), whereas the column Rational explains the rationale of the Support specification.

***Table 10-1**. HackLearn's Evaluation*

Characteristics	Support	Rational
Segment 1: Pedagogical Considerations$_{GBL}$ & Pedagogical Benefits$_{CTF}$		
Cognitive learning$_{GBL}$	✔	HackLearn is based on the cognitive learning theories, as it constitutes an educational environment where learners can perform actions, experiment, reflect on their deeds, utilize new practices and assimilate new KSAs. Moreover, HackLearn fosters critical thinking and problem-solving capabilities, as the learner appraises the context of the game plans and executes a CKC attack.
Creativity$_{GBL}$	✔	In HackLearn, instructors define scenarios in which learners think outside of the box and exercise new skills. For example, in the step S5 of the prototype scenario the learner has to apply a genuine attack pattern in order to analyze the manner the password recovery mechanism of the target service operates, retrieve the password hint, excavate user's personal information and get the credentials required to proceed to the next step.
Engagement, immersion, motivation & fun$_{GBL}$	?	HackLearn adopts the attack concept of live competitions, an important factor that enhances the motivation and the entertainment factors (Chung and Cohen, 2014). Additionally, it draws elements from role-playing games that reinforce the engagement and immersion characteristics, as learners assume in-game roles and maintain profiles containing collections of KSAs. Unlike live competitions, the fun and motivation factors are affected by the

		employed instructional learning approach, as learners are obliged to follow the game's scenario elaborated by the instructor.
Continuous learning$_{GBL}$	✔	HackLearn implements a continuous learning approach, as the game is 'always-on' providing the means for organizing learning sessions repeatedly. In learning sessions, learners acquire new KSAs or exercise the KSAs they already possess (adopted KSAs). To regularly exercise the adopted KSAs, an instructor can implement a policy of decreasing the *LO Degree* values in the learner's profile for LOs whose possession has not been achieved for a specified period (specified in the *last update* attribute of Role objects). Consequently, the learner has to periodically repeat training sessions that exercise KSAs bound to LOs with low *LO Degree* values, and thus she enters in a continuous lifecycle of learning, updating and reinforcing KSAs. HackLearn provides the opportunities for learners to exercise their adopted KSAs in new ways (Sessa and London, 2015) by altering the narratives, the cyberspaces and the conditions of the sessions and by utilizing randomization in the attributes of the entities (e.g., network's IP address).
Self-directed learning$_{CTF}$	✘	As opposed to live competitions which promote self-directed learning, HackLearn promotes instructional learning.
Exercise of knowledge, skills and abilities$_{CTF}$	✔	In HackLearn, learners exercise techniques and basic skills such as discovering live hosts in a network (in step S1), scanning the target's ports (in step S2), and creating a weapon payload file (in step S6).
Collaboration$_{CTF}$	✘	HackLearn is a single-player game and lacks the promotion of collaboration among learners in the context of the game.

Segment 2: Learning Outcomes$_{GBL}$		
Connection to the game-play	✔	HackLearn infuses the LOs in the game-play and associates the gaming goals with the learning objectives (analysis presented in sub-section 4.3).
Learning outcomes show purpose and they are measurable	✔	HackLearn's LOs are based on the parent KSAs, they are measurable and they have clear purpose.
Assess proficiency and performance	✔	The assessment of the LOs is based on the measurement of the learners' performance as it involves the recording of the tasks' details (e.g., duration, number of repetitions) associated with the LOs. Moreover, HackLearn aims at assessing the LOs in various in-game contexts to ensure the proficiency in exercising the cyber security knowledge and skills under different conditions.
Segment 3: Architecture$_{GBL}$		
Open access	✔	HackLearn provides open access as anyone can use it anytime from anywhere.
Configurable environment	✔	HackLearn allows the full configuration of the environment in which the learner operates, mainly through the specification of the cyberspace and the conditions in the scenarios.
Manage portfolios of learning objects	✔	HackLearn's repositories can be considered as portfolios of cyber security learning objects which can be adopted in various learning and training environments.
Multiplayer	✘	HackLearn only operates in single-player mode.

Modes of operation	?	HackLearn operates in training mode, but it does not support certification and competition mode. Thus, it embraces one of the three modes of operations.
Incorporation of various games	✗	HackLearn is not a game suite and it does not incorporate a collection of different genre games with different user interfaces and characteristics.
Automation of red team and white team activities	✔	HackLearn requires learners to perform red team activities and it can automate white team activities.

Segment 4: Design and game mechanics$_{GBL}$

Orientation to genre	✔	HackLearn is a hacking simulation game (justified in sub-section 5.1).
Team training	✗	HackLearn does not support team training
Focus on learning	✔	HackLearn complies with the ATMSG model that facilitates the assimilation of the learning aspect in the game's design.
Realism	?	HackLearn does not exhibit the realism of live competitions that run in real settings. However, it involves a certain degree of realism specified by the instructors in the game's scenario through the definition of the cyberspace including entities that imitate the behavior of real devices.
Narrative	✔	HackLearn has a narrative defined by the instructor in the *Description* attribute of the scenario object.
Progression	✔	HackLearn supports real-time progression in the game, as a single-player game. In single player games conflicting and simultaneous actions (Nagarajan et al., 2012) do not occur.

Player's identity	✔	Learners have a role and a personal profile they maintain.
Player's view	✔	The view of the game in single-player is definite and exclusive for the learner.
Interaction	✘	HackLearn does not provide interaction with players and non-playable characters
Segment 5: Adaptability$_{GBL}$		
Complexity adjustment and tuning of stress levels	✔	HackLearn's adaptability facet involves the adjustment of complexity and the tuning of the stress levels in order to optimize the game's effectiveness. To implement game sessions of varying complexities, instructors define a collection of scenarios referring to diverse subjects and associated with various LOs. The scenarios evolve in terms of the number of steps specified, the number of conditions, and the number of entities included. To increase or loose the stress levels, the instructors define in the grading schemes the properties related to the time provided to the learners to perform their tasks, the number of actions they have to perform and the support provided by the game. For instance, the presented prototype scenario refers to learners that have a degree in computer science aiming at following a career in cyber security. For this reason, the scenario's complexity is tuned high in order to motivate and challenge the learners. However, the learners are considered inexperienced CtF participants, and thus the scenario has loose time limits and provides strong support to the learners through the provision of hints and teaching materials.
Learning history	✔	HackLearn stores the learners' learning history in the back-end storage facility (stated in the sub-section 4.3)

Participant's analysis and available time	✔	The instructor considers the learner's characteristics (e.g., background, retention, expectations etc.) and the educational context (e.g., available time, budget, presence of an instructor etc.), and forms the appropriate scenarios for the learner.
Segment 6: Assessment$_{\text{GBL and CTF}}$		
Feedback$_{GBL}$	✔	HackLearn provides feedback to learners through the textual responses of the terminal, the use of visualizations and the providence of a score leaderboard. Besides, HackLearn displays in the learner's profile the *LO Degree* and *Degree* metrics, associated with the achieved LOs possession.
Victory conditions$_{GBL}$	✔	HackLearn considers victory conditions in terms of speed (associated with the time passed since the last action), duration (associated with the time passed since the last SEF) and accuracy (associated with the number of actions since the last task).
Points$_{GBL}$	✔	HackLearn counts scores and grades
Incentives for good practices and disciplinary actions for repeated mistakes$_{GBL}$	✔	HackLearn's instructors define the grading scheme to reward good practices and to penalize unjustified details and repeated errors.
Mayer's methodology$_{GBL}$	✘	HackLearn does not employ the Mayer's methodology (Mayer, 2012)
Formative and summative assessment$_{CTF}$	✔	HackLearn performs a formative assessment, as it counts and displays the score and informs the learner when a goal is achieved. HackLearn also performs summative assessment, as it records the learning history of learners that are available to the instructor.

Assessment features $_{CTF}$	✔	HackLearn's assessment is fair, objective and comprehensive.
Segment 7: Issues and Challenges		
Demands $_{CTF}$	✔	HackLearn demands include the cost of development and the need for cyber security specialists, game developers and instructors. After the creation of the game and the scenarios, the HackLearn learning, and training sessions have minimum demands.
Frequency of events $_{CTF}$	✔	Learning and training sessions can be repeated very often.
Aims $_{CTF}$	?	As opposed to live competitions, HackLearn aims at forming an organized environment that provides possibilities and guidance to learners to adapt by acquiring new KSAs. However, HackLearn is a hacking simulation game that does not take into account operational and maintenance issues such as operational costs of the systems, updates and upgrades, implementation of disaster-recovery policies, backup schemes etc.
Diversity of topics $_{CTF}$	✔	Although, the prototype scenario presented in this study is a penetration testing scenario aiming at fostering vulnerability analysis KSAs, the HackLearn can embrace scenarios from different areas of the cyber security domain (e.g., cryptography, cyber threat intelligence etc.).
Partial credit $_{CTF}$	✔	HackLearn assessment provides partial credit to the learners even when they do not accomplish a mission, but they make some progress towards the scenario's goal (i.e., the capture of the flag).

Table **10-1** shows that HackLearn embraces 68 out of 78 characteristics (i.e., that is 87%) of the GBL and CtF concept maps, from which 5 characteristics (i.e., *engagement*, *immersion*, *motivation* and *fun* of segment 1 and *realism* of segment 4) are merely supported. On the other hand, from the 10 characteristics not supported by HackLearn, 5 characteristics are associated with the lack of multiplayer support (i.e., the *collaboration* of segment 1, *multiplayer* and *competition mode* of segment 3, *team training* and *player interaction* of segment 4). HackLearn embraces 14 of the 17 characteristics of the *Pedagogical Benefits* and *Assessment* segments (i.e., 82%), though it does not support the *self-directed learning* characteristic and the *collaboration* and *teams* characteristics of the *Pedagogical Benefits* segment that are also included in the GBL concept map. On the contrary, HackLearn supports all the characteristics of the *Assessment* segment, yet it does not require the learner to write up reports or the presence of a supervisor to perform the summative assessment. Finally, the *Issues and Challenges* segment shows that HackLearn can confront all the challenges and issues identified in the live competitions field, apart from the challenge that it does not include and realistically present the views of systems associated with the operational costs, the update and back up policies of systems, etc.

10.2.2. Discussion

The results of the evaluation presented in section 6 allow a good appreciation of HackLearn's learning and training effectiveness, as most of the key elements proposed in (Katsantonis et al.,2017b) and in (Katsantonis et al.,2017a) are embraced. Specifically, HackLearn embraces several pedagogical characteristics (listed and analyzed in segments 1 and 5 of Table 7) including the conformance with modern learning theories (presented in sub-section 2.2) that verify its effectiveness. HackLearn's design is based on the activity theory (through the conformance with the ATMSG model) and it additionally supports a repertory of learning theories, from behaviorism (e.g., when learners have to improve adopted KSAs in terms of speed and accuracy) to constructivism (e.g., when instructors foster creativity, problem-solving and critical thinking capabilities).

Moreover, the evaluation of HackLearn's design shows a valuable perspective in the assessment of learners' efforts. That is because HackLearn assimilates well known cyber security models and standards such as CAPEC and CKC to interpret learners' actions and strategies towards unleashing cyber-attacks. The assimilation of these standards is a determining factor in creating an organized, and parameterized environment where learners' actions are monitored, recorded, and dynamically assessed. Subsequently, HackLearn provides the instructors the capability to tune the complexity of the upcoming learning and training sessions by increasing the size of the cyberspace

and the number of steps or by making stricter the grade schemes (presented in segment 5 of *Table* **10-1**). In such a way, the training and learning sessions created in HackLearn can be adapted to the participants' needs and capabilities.

Besides, HackLearn's characteristics listed in the *Issues and Challenges* segment allow a preliminary estimation that HackLearn provides hands-on cyber security learning and training approaches with lower preparation and running costs compared to live competitions. Once a HackLearn is developed and a collection of scenarios is created, the COFELET compliant cyber security learning and training approaches will have minimum demands. Although the development of scenarios includes a certain degree of logical complexity, the formation of scenarios is facilitated through the description of the scenarios' structural elements in the COFELET ontology, and the reuse of objects stored in the repositories.

HackLearn has an always-on architecture that allows learners to use it anytime and anywhere. Nevertheless, HackLearn is a game-based learning approach that is more likely to motivate young learners to engage in cyber security, increasing the chances to motivate them to chase a career in cyber security.

On the other hand, limitations in the pedagogical effectiveness of HackLearn result from the lack of multiplayer support as in single-player games learners do not have the chance to work as members of a team, communicate with their teammates, cooperate or compete. In the primary analysis of the presented work, the multiplayer support feature was in the plans of the HackLearn development. However, in the first iteration of the study, the inclusion of the multiplayer feature was considered infeasible because it raises very much the complexity of the game's design and the creation of scenarios.

Another issue revealed by the evaluation of HackLearn is that it is a single-mode game and it only operates under the umbrella of the hacking simulation game genre. In particular, learner mostly interacts with the game's terminal by entering text-based commands. On the contrary, a cyber security game suite including a collection of different genre games, multiple UIs and multiple modes of operation (e.g., certification and competition modes) promises to offer better effectiveness and pedagogical benefits (e.g., enhanced motivation and immersion factors) than HackLearn does.

10.3. Evaluation in real settings

10.3.1. Methodology

HackLearn's evaluation methodology (*Figure 10-1*) adopts several aspects of the models and frameworks presented in section *2.3.7* (represented in *Figure 10-1* by circles):

- *Didactic framework*: HackLearn's evaluation methodology adopts the flow process of the didactic framework by embracing its stages (i.e., *Preparation*, *Introduction*, *Interactions*, and *Conclusion*) and the phases of assessment (i.e., *pre-game assessment*, *in-game assessment*, *post-game assessment*).
- *COFELET framework*: In the *Interactions* stage learners use HackLearn, which is a simulation game based on the COFELET framework and the COFELET ontology presented in Section 6. The utilization of the COFELET framework helps in developing and running a hacking simulation game. Additionally, it aids in the design and performance of the *in-game assessment*, as it analytically describes the components that have to be included in such games and the elements that have to be assessed.
- *CKC model & APs*: they are demonstrated as teaching materials in the *Introduction* stage and then they are utilized by learners in the *Interactions* stage to plan and perform their mission. The CKC and the APs utilized in this study are associated with the *il Segreto di Arlecchino* scenario and the SEFs. Learners use the CKC model in the *il Segreto di Arlecchino* scenario as a blueprint for planning their strategy and the APs as patterns for applying hacking techniques. However, in other COFELET scenarios different models can be considered such as the Diamond model (Caltagirone, et al., 2013) and the ATT&CK (Strom et al., 2018) model for strategy planning and attacking.
- *QC framework*: aided in selecting the characteristics on which the questionnaire of the post-game assessment focuses.

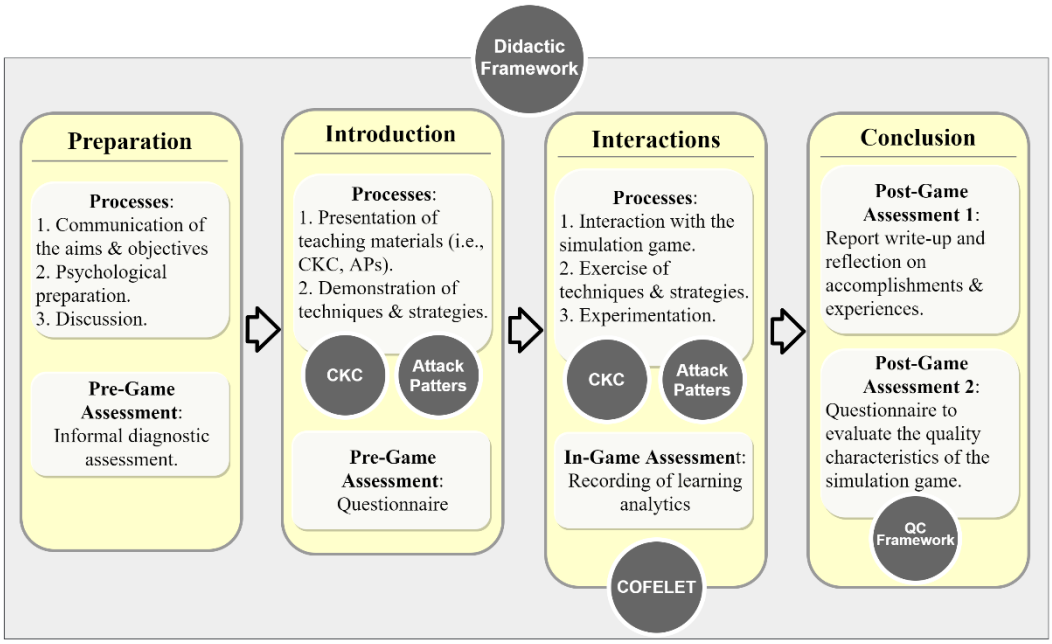


***Figure 10-1**. Evaluation Methodology*

In the remainder of this section, the aspects of HackLearn's evaluation methodology are analytically presented such as the analysis of the questionnaire used in the post-game assessment 2 of the Conclusion stage and the manner that the quality characteristics of HackLearn were assessed.

10.3.2. Experiment

The HackLearn COFELET game was evaluated in the context of the Networks and Web Applications Security course of the Department of Applied Informatics at the University of Macedonia in Thessaloniki, Greece. In the experiment participated 103 fourth-year (i.e., final-year) undergraduate students.

For the evaluation of HackLearn, the didactic framework presented in section *2.3.7* was adopted. Although the didactic framework proposes a flow of processes for business simulation games, the framework was also applied in the evaluation of HackLearn that is a cybersecurity simulation game. The *Preparation* and *Introduction* stages have been conducted in an introductory lecture, which followed a penetration testing lecture wherein the execution of the HackLearn sessions happened. The lecture was delivered on-line through the Zoom platform due to the Covid-19 pandemic. The *Interactions* stage included the HackLearn game sessions the learners participated in and performed as homework outside the regular class period. In the *Conclusion* stage learners answered a questionnaire and wrote a short report post to the execution of the game

sessions. In the remainder of this section, the stages of HackLearn's evaluation process are presented in more detail.

10.3.2.1. Preparation stage

In the first part of the introductory lecture, the students were informed of the aims and objectives of the penetration testing part of the course. Specifically, it was made known to the students that they will learn penetration testing concepts and that they will practice cyber-attack techniques and strategies. The students were also informed that they will interact with a learning environment that provides the opportunities to experiment safely with cyber-attack techniques and it will scaffold their efforts. Additionally, the concept of ethical hacking and the techniques of penetration testing were discussed, and the necessity of ethical hacking was pointed out.

10.3.2.2. Introduction stage

The introductory lecture was delivered to the students presenting the Cyber Kill Chain model and the attack patterns of host discovery, port scanning, password recovery exploitation, and authentication abuse. Subsequently, the introductory lecture was followed by the penetration testing lecture in which the usage and the syntax of the nmap, ftp, ssh, searchsploit, msfvenom, metasploit, ifconfig and base64 tools were presented. Additionally, the HackLearn game was introduced to the students and a demo scenario was explained, in which a host discovery attack pattern (i.e., the ICMP Echo Request Ping attack pattern) and a port scanning attack pattern were presented along with the decoding of base64 encoded text. During the demonstration, students were informed that HackLearn counts participants' scores based on an advanced assessment facility (Katsantonis et al., 2021) according to which the assessment facility grades participants' efforts by keeping track of their times, the number of actions they perform, the hints they acquire and the number of times they play the game. The top 10 scores are presented on the game's leaderboard.

10.3.2.3. Interactions stage

After the penetration testing lecture, students had one week to perform the interactions stage. Students initially created an account as penetration testers. Then, they entered the game and they followed the interactive tutorial (*Figure 8-7*) and they answered the five (5) multiple-choice questions of a self-report questionnaire, in which they declared their prior knowledge and experiences in the lecture's topics (*Figure 10-2*).

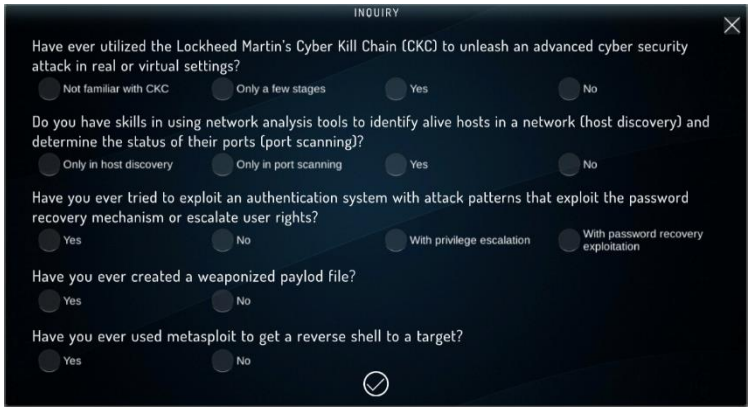


***Figure 10-2**. The pre-game questionnaire*

Students could to play the il Segreto di Arlecchino scenario several times to achieve the scenario's goal. To do so, students had to develop and implement a strategy that adopts the stages of the CKC model. During the game session, students performed actions and interacted with the game's entities (e.g., network, host, firewall, file system, service) that simulate the behavior of real devices. The student's actions were always followed by the game's feedback as a result of students' activities. The feedback was delivered in textual form through the game's terminal and interface (i.e., score and progress in the progress bar). Therefore, students had the opportunity to refine failed techniques and strategies and to try different approaches. For example, in step S2 of the il Segreto di Arlecchino scenario (section *8.5.5*) the students had to change the host discovery ICMP Echo Request Ping attack pattern they initially adopted to find the network's hosts because the game's firewall dropped the ICMP packets. The Instructors chose to demonstrate the ICMP Echo Request Ping attack pattern in the Introduction stage because it fails in the context of the il Segreto di Arlecchino scenario. Thus, the students were led to a cognitive conflict (Mischel, 1971).

10.3.2.4. Conclusion stage

In the Conclusion stage, students answered the post-game assessment questionnaire containing Likert scale and multiple-choice types of questions and wrote a report. In the report students described their actions, the strategy they employed, their achievements, the pitfalls they identified in the game, the comments on their experiences with HackLearn, and suggestions for the improvement of the game. The purpose of the report was to make students reflect on their actions and experiences with HackLearn and to express their opinion on the game in a more open-ended way than they did with the questionnaire and make suggestions.

10.3.3. Assessment phases

HackLearn's evaluation is mainly performed during the *in-game* and *post-game* assessment phases, depicted in *Figure 2-13. The Didactic Framework associated with three assessment phases*. Previously, a multiple-choice pre-game questionnaire was used (depicted in *Figure 10-2*), in order to record the students' prior knowledge in penetration testing and cyber-attack strategies.

An informal diagnostic assessment was performed when the instructor asked questions and discussed the concepts of penetration testing during the lecture of the Preparation stage to appraise in situ the prior knowledge of students.
The *in-game* assessment was carried out during the *Interactions* stage, in which students were involved in the *il Segreto di Arlecchino* scenario with the gaming objective of capturing the file flag.txt. During the *in-game* assessment phase, HackLearn collected learning analytics that provide insights on the students' efforts and achievements in the game's environment and the scaffolding they required. The *post-game assessment* phase was carried out during the *Conclusion* stage in which students answered the post-game questionnaire (presented in section 10.3.2.4) to evaluate HackLearn and wrote a short report to explain their in-game activities and to express their views on the HackLearn.

10.3.4. The post-game assessment questionnaire

The design of the post-game assessment questionnaire was based on the quality characteristics of the framework presented in section 2.3. *Table 10-2* lists the questions of the questionnaire along with the assessed quality characteristics of HackLearn (third column) and a question code (first column). Specifically, question Q1 aims at assessing the students' perceptions on how effective (*effectiveness* characteristic) the HackLearn is in teaching the topics of the penetration testing module of their course. The questions Q2 and Q5 refer to the *engagement* and the *motivation* characteristics as they aim to assess the degree to which the students were challenged by the mission of the *il Segreto di Arlecchino* scenario and the HackLearn's leaderboard feature. Question Q3 aims at examining how interesting and motivating (*motivation* characteristic) the HackLearn game is. Question Q4 refers to the *usefulness* and *acceptance* characteristics of HackLearn as it aims at assessing how much students like the adoption of serious games, such as HackLearn, in the university course materials. Concludingly, the question Q4 implicitly refers to the *engagement* characteristic, as the acceptance is linked to the *engagement*. The questions Q6 and Q7 refer to the *effectiveness* characteristic, as they aim to assess the degree to which students believe that HackLearn's scaffolding features enhanced their performance (Davis, 1989). Finally, the questions Q8 to Q10 refer to

HackLearn's *usability* and *user satisfaction* characteristics, as questions Q8 and Q9 aim at assessing the game design aspects (e.g., background, colors, icons), whereas question Q10 aims at assessing how usable and understandable (*understandability* characteristics) the HackLearn's interface is and how much *user satisfaction* it provides.

***Table 10-2**. The post-game assessment questionnaire*

Id	Question	Quality characteristics
Q1	The utilization of the HackLearn hacking simulator game helped me to comprehend the Cyber Kill Chain model and the attack patterns hackers use to unleash cyber-attacks.	Effectiveness
Q2	The Harlequin mission of the HackLearn game was a challenging assignment.	Engagement, motivation
Q3	I am interested to have more missions in Harlequin.	Motivation
Q4	I would like other courses and subjects to use serious games with simulations (e.g., networks, programming, management, business).	Usefulness, acceptance, engagement
Q5	I would like the top 10 leaderboards to present the scoring of all my colleagues.	Engagement, motivation
Q6	The hints assist me to complete the mission of the game.	Effectiveness
Q7	The teaching contents assist the players to recall and/or comprehend some aspects of the game (e.g., tools' usage, description of attack patterns).	Effectiveness
Q8	I liked the colors and the background of the HackLearn game.	User satisfaction, game design
Q9	I liked the icons of the HackLearn game.	User satisfaction, game design
Q10	It is easy to understand how the game interface works to carry out the mission.	Usability understandability, user satisfaction

10.3.5. Evaluation parameters

HackLearn's evaluation strategy involved the definition of the evaluation metrics used to measure HackLearn's quality characteristics. The evaluation of HackLearn's *effectiveness* was performed with respect to the students' prior knowledge on the topics of the penetration testing. For the evaluation of *effectiveness*, the following parameters were considered:

i. The recorded number of steps students performed.

ii. The number of in-game questions students answered satisfactorily (i.e., graded over 60%).
iii. How much do students think that the utilization of HackLearn helped them to comprehend the topics of the penetration testing lecture (i.e., answers to question Q1).
iv. The recorded number of hints they acquired per step during the game sessions (*hints per step*). Since students had the possibility to play multiple sessions, the *hints per step* were calculated by considering the maximum number of hints per step from all the sessions students played. For example, if a student requested 4 hints in step 2 of his/her first session and 1 hint in step 2 of the proceeding session, it was considered that the student requested 4 hints in step 2.
v. How much students valued the support they had from the game's hints on the Likert scale of the questionnaire they answered in the *post-game* assessment phase (i.e., answers to question Q6).
vi. How much students valued the support they had from the game's teaching contents on the Likert scale of the questionnaire they answered in the *post-game* assessment phase (i.e., answers to question Q7).
vii. Any comments and suggestions made regarding the effectiveness of the game in the *post-game* assessment phase report.

The evaluation of HackLearn's *engagement* and *motivation* characteristics was combined, as the engagement is based on the motivation characteristic (Dele-Ajayi, 2016). Thus, for their combined evaluation, the following parameters were considered:

i. The number of sessions the students performed.
ii. The total time they spent in the game and the average time they spent per session.
iii. The number of actions they performed in the game.
iv. How much interesting and motivating (i.e., question Q3), challenging (i.e., questions Q2 and Q5) and useful (i.e., question Q4) students valued their experience with HackLearn on the Likert scale in the questionnaire the filled in the *post-game* assessment phase.

The *usability* characteristic was associated with how much students valued the easiness of use and the understandability of the user interface (i.e., question Q10 which also related to the *understandability* characteristic), whereas the *user satisfaction* and the *usability* were associated with how much students appreciated the design of the game's interface (i.e., questions Q8 and Q9). The *usefulness* and *acceptance* characteristics were associated with how much students would like the adoption of serious games with simulations in the university courses (i.e., question Q4). The characteristics of *user experience*, *usability* and *user satisfaction* were also associated with the related

comments and suggestions students made in the report of the *post-game* assessment phase.

10.3.6. Results

10.3.6.1. Effectiveness:

In the pre-game questionnaire, students were asked to answer the questions depicted in Figure 8, based on their prior experiences in applying the CKC and cybersecurity attack patterns and techniques. Instructors chose to ask the students to declare their prior knowledge and not to test it, as it was expected that only a minor percentage of students would have prior knowledge in penetration testing. Besides, instructors had the possibility to preliminary appreciate the students' prior knowledge in the *Preparation* stage. *Figure 10-2* shows the results of the *pre-game* inquiry according to which only a minor percentage of students had experiences and knowledge in the penetration testing topics.

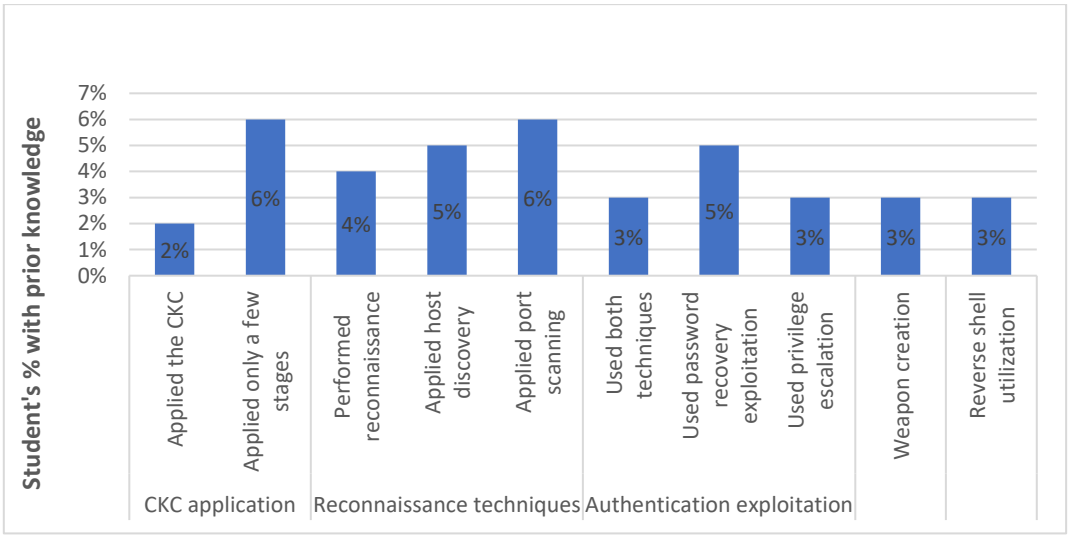


***Figure 10-3**. Pre-game inquiry results*

In the *Interactions* stage, 51 students managed to capture the file flag.txt, whereas from the 11 students who declared that they had prior experience in penetration testing, 7 students captured the file flag.txt. Almost 66% of the students achieved at least 5 out of the 9 mission steps (*Figure 10-4*).

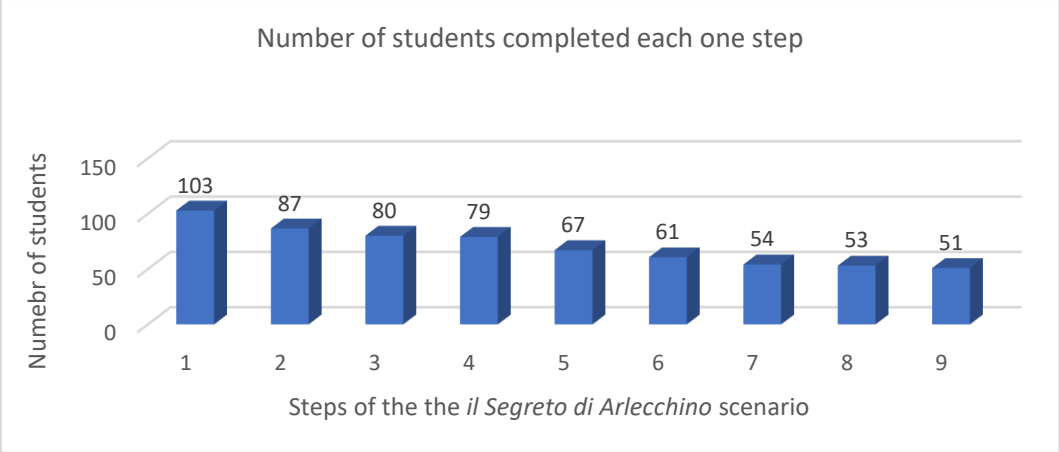


***Figure 10-4**. Number of students per reached step of the il Segreto di Arlecchino scenario*

On average students completed 6.17 steps per se with a standard deviation of 3.18. Each student answered satisfactorily on average 4 questions with a standard deviation of 2.66. In question Q1 of the post-game assessment questionnaire, students showed that they appreciated the usefulness of HackLearn in comprehending the CKC model and the cyber-security attack patterns (*Figure 10-5*).

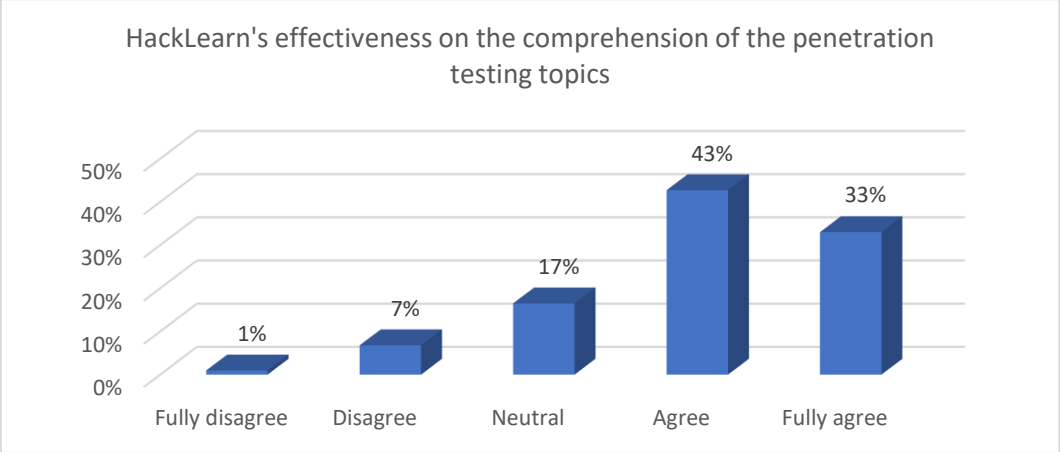


***Figure 10-5**. Percentage breakdown of students' answers to Q1*

Moreover, students showed that they generally appreciated the help they had from the game's scaffolding facilities. Specifically, 69% of the students agreed or fully agreed that game's teaching contents helped them recall and/or comprehend some aspects of the game (i.e., question Q7), whereas 54% agreed or fully agreed that the hints effectively supported them to complete the mission of the game (i.e., question Q6). Though, a considerable percentage of 31% answered that they feel neutral on the support they had from the hints of the game, whereas 16% of the students stated that they disagree or fully disagree that hints helped them to accomplish the mission.

According to the game's analytics each student requested 1.49 hints per step with a standard deviation of 1.45, whereas the 33% of the students that completed up to the step 4 requested on average 0.92 hints per step.

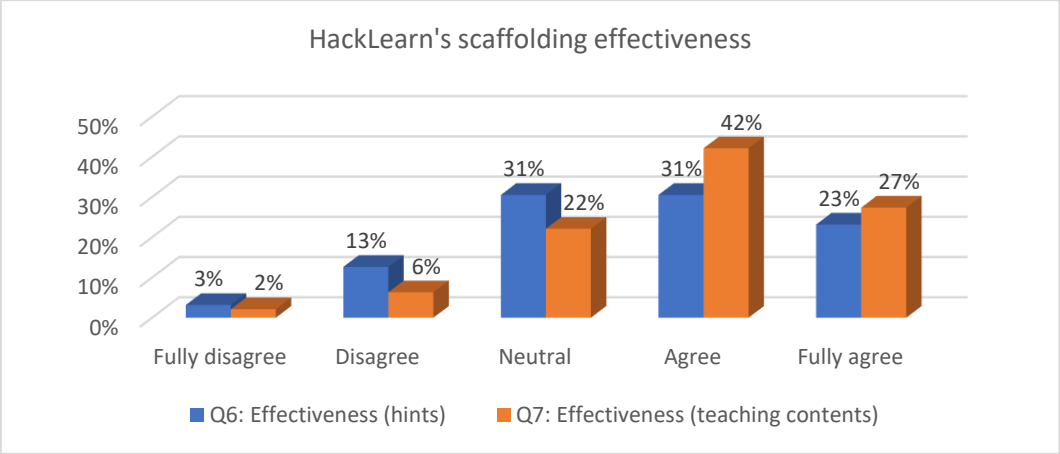


***Figure 10-6**. Percentage breakdowns of students' answers to Q6 and Q7*

In the report of the *post-game* assessment phase, more than 60% of the students stated that the game was an efficient and interactive way to learn the topics of the penetration testing module of their course. Some students also stated that their experience with HackLearn raised their awareness of the security policies applied nowadays (e.g., in the creation of passwords, the protection of accounts). A suggestion that worth's mentioning proposed an enhancement of the game's scaffolding by improving the help option of the *in-game tools* (e.g., `nmap -help`) to provide details on the tool's usage, syntax, etc.

10.3.6.2. Engagement & Motivation

During the *Interactions* stage, 448 sessions were performed and stored in HackLearn's database. Learners performed an average of 4.36 sessions per se with standard deviation 2.70. On average each user spent approximately 56 minutes in the game (3.396 seconds) with a standard deviation of approximately 40 minutes (2.452 seconds), and an average time of 13 minutes per session. Students performed on average 16.78 actions per session with a standard deviation 8.71. Moreover, from the 51 students that captured the flag 34 students (i.e., approximately 65%) replayed the mission possibly to improve their records and scores. In the *post-game* assessment phase, 86% of the students found HackLearn a challenging assignment (i.e., Q2), 66% of the students are interested in playing more scenarios, and 92% of the students would like to use simulation games in university's courses. Though, only 45% of the students were interested in finding out through the leaderboard how their colleagues performed in the game.

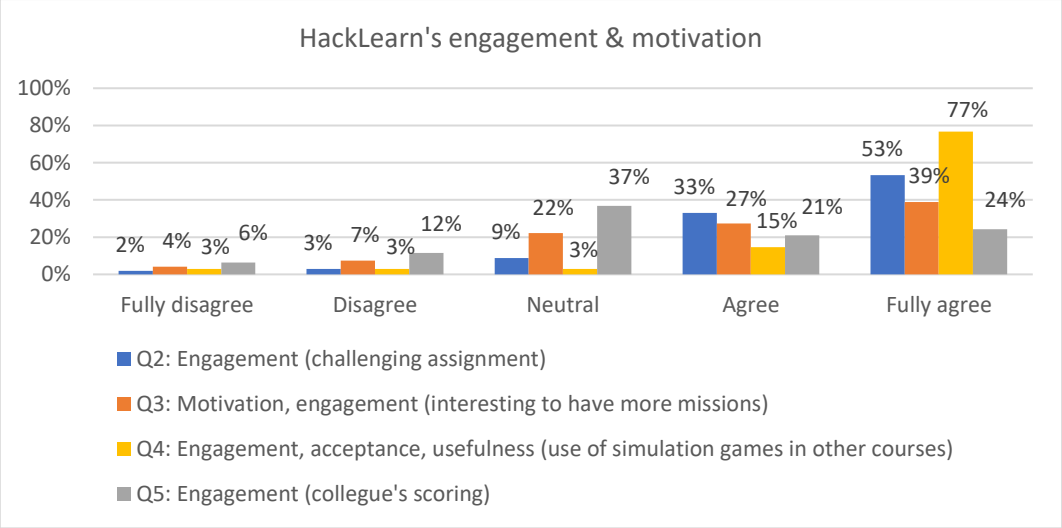


Figure 10-7*. Percentage breakdowns of students' answers to Q2, Q3, Q4 and Q5*

In the report of the *post-game* assessment phase, most of the students stated that the game was a challenging and interesting experience with clever challenges and they really enjoyed that they learned new topics in such a practical and efficient manner.

10.3.6.3. Usability & user satisfaction

Students showed in the *post-game* assessment questionnaire that they were satisfied with the usability and the game design aspect of HackLearn. Specifically, 70% of the students answered that they found the colors and the background of HackLearn usable, 72% that they understood quickly the meaning of the game's icons (*Figure 10-8*) and 70% stated that it was easy to adopt the manner that HackLearn's works (*Figure 10-9*). However, 27% of the students stated in their report of the *post-game* assessment phase that they experienced connection problems while playing the game and they had to replay the game several times from the first step. At this point, it should be noted that the sessions terminated due to connection problems (terminated sessions) were spotted and excluded from the evaluation process. Additionally, students stated that it was frustrating that the game kept asking answers for the in-game questions, even though students had provided answers in preceding sessions. Students suggested that the game should have a save facility that will save a game session's state and the learners' progress and a load facility that will allow learners to restore their game session and continue from their last checkpoint.

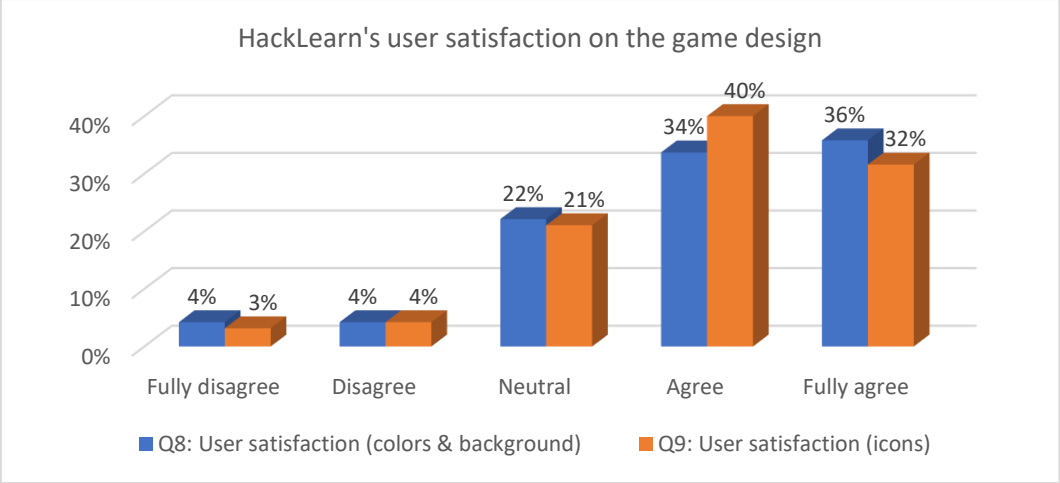


***Figure 10-8**. Percentage breakdowns of students' answers to Q8 and Q9*

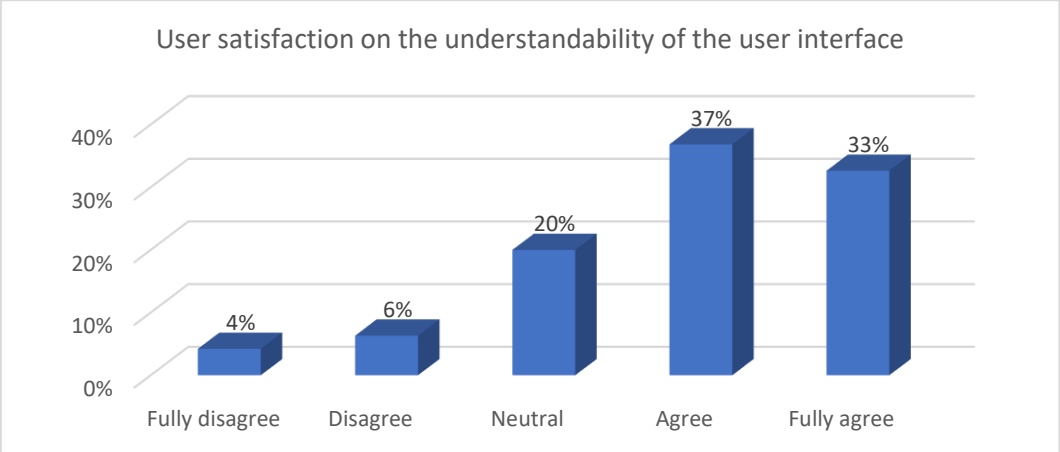


***Figure 10-9**. Percentage breakdowns of students' values on the understandability of HackLearn's user interface*

10.3.7. Discussion

The COFELET framework foresees the improvement of cybersecurity education impact, through the development of proper means to deliver effective cybersecurity learning and training. HackLearn is an innovative COFELET game based on modern learning theories and well-known cyber security standards aiming at teaching cyber security concepts while providing hands-on experiences to learners. As HackLearn is the first game of its genre (Katsantonis et al., 2021), its impact cannot be compared with the impact of other cyber security serious games. However, the results of the presented evaluation can aid in coming to some deductions on HackLearn's impact.

HackLearn has been adopted successfully in a learning approach of a real educational environment, it enhanced a didactic process with many learning benefits, and thus it

can be part of the university's course materials. Specifically, according to HackLearn's analytics, a high percentage of the students were engaged in a game that they played as homework, outside a regular class period. In fact, many students replayed the game several times to achieve the gaming goals or their personal goals (i.e., to increase their scores and make a leaderboard record). The students declared in the *post-game* assessment phase that they considered HackLearn effective in comprehending the module's topics, interesting, challenging, useful, and motivating. Many students particularly commented that they enjoyed the HackLearn sessions because it was a challenging task that required critical thinking. Additionally, it is notable that 92% of the students stated that they would like to use serious games with simulations in university's courses (i.e., 77% fully agreed and 15% agreed), a characteristic that shows that students prefer to be active learners instead of passive receivers of information as with traditional teaching methods.

A high percentage of the students stated in the *post-game* assessment phase that the teaching contents and the hints of the game helped them to carry out the mission (i.e., 69% and 54% respectively). However, a considerable percentage of students stated that they did not appreciate the support they had from the game's hints. Besides, the learning analytics show that the students that did not do well (i.e., the students that reached up to step 4) only requested on average 0.92 per step, whereas one would have expected that they should had used all the support they could get from the game. Thus, more efficient strategies have to be considered for the provision of hints to the learners and especially for learners that find it difficult to function well in the game. Such strategies are the provision of free hints (i.e., hints without score impact) and the formation of attractive and more efficient hints.

In the user satisfaction aspect, although the game has a simple 2D design, most of the students stated that they liked the game design, and they used the user interface without difficulties. However, a considerable percentage of the students experienced the session termination problem due to connection problems, as the game could not communicate with the database to store the sessions' analytics and the students' answers to the in-game questions. The connection problem was an intense problem probably due to the instability of the internet during the Covid-19 virus pandemic and in many cases happened due to the students' unstable connection. However, apart from the save and load features suggested by the students, HackLearn can improve the user experience aspect by incorporating a connection examination mechanism and a buffer mechanism. The connection examination mechanism will constantly test the quality of the participants' connection and the buffer mechanism will occasionally store the game's data when temporary connection problems exist. When the connection is stable the buffer mechanism will query its data to the database.

10.3.8. Chapter Conclusions

This preliminary evaluation showed that HackLearn has the potential to deliver effective cybersecurity education services with advanced scaffolding and assessment capabilities. Besides, a preliminary estimation of the cost shows that HackLearn has lower preparation and running costs than live competitions, as it is considered cheaper to create game scenarios based on reusable elements than organizing and running live competitions (e.g., Capture the Flag - CtF). Though, the employed evaluation in (Katsantonis et. al, 2021) aimed at assessing the game's feasibility in the design phase, thus an evaluation of HackLearn user experience in real settings is necessary to measure its educational effectiveness and to come to safe deductions on the game's impact.

In this section, we presented the evaluation of the HackLearn COFELET game user experience. More specifically, we described the manner that HackLearn can be adopted in a real educational setting by adopting the didactic framework (Utesch, 2016) and we analyzed the methodology we followed to evaluate HackLearn's impact. Specifically, in the presented evaluation process we assessed the game's effectiveness in teaching the CKC model and the attack patterns hackers apply to unleash their attacks, and we assessed how engaged, motivated, and satisfied the learners were by HackLearn. The results of our evaluation show that such approaches are very promising since HackLearn was a beneficiary addition in a university's class.

The results show that the COFELET framework facilitates feasible and effective solutions and reveals the limitations of the HackLearn game.

11. CONCLUSIONS AND FUTURE WORK

This section summarizes and discusses the presented study by comparing the original research questions and the problem statement presented in the Introduction of this thesis. The section also states the limitations of the presented study and some considerations for future work.

Cyber security education is an emergent and complicated domain facing many challenges. In this study, we examined the domain of cyber security education and we identified and analyzed the strengths, the problems and the key issues (i.e., Objective 1 which answers the RQ1). As live competitions are an important part of cyber security education, we analyzed the domain of live competitions utilized in educational environments and we identified the pros and cons. The results of our study include a concept map of live competitions' key characteristics and a categorization of live competitions problems which downgrade the impact of live competitions when utilized in educational settings.

The direction that we employed in the current study foresaw the enhancement of cyber security education by the mitigation of the identified problems through the utilization of emergent techniques and methodologies based upon the foundation of modern learning theories. Educational games are increasingly used in recent years and they are proved to have a significant impact in many areas. They have been used as an innovative strategy for teaching specific subjects, practicing skills and abilities, and changing attitudes in a wide range of areas such as healthcare, etc. Thus, our research interest focused on the possibilities educational games provide to support cyber security learning and training. We reviewed serious games design frameworks, we analyzed the current cyber security game-based learning and training approaches, and we identified the lack of common methodologies and empirical studies. To tackle this problem, we proposed the COFELET framework as a means for the development of cyber security game-based learning and training (i.e., Objective 2 which answers the RQ3). The COFELET framework envisages the exploitation of the strengths of cyber security education by embracing well-known methodologies and models of the cyber security domain and by considering the key characteristics of live competitions. Moreover, COFELET integrates the key design concepts of existing frameworks and models such as the Mechanics Dynamics Aesthetics (MDA) framework, the Design Play Experience (DPE) framework, and the Activity Theory Model for Serious Games (ATMSG) focusing on the pedagogy aspect with respect to cyber security learning. To provide support for the development of COFELET compliant approaches, the COFELET ontology has been proposed aiming at providing coherent descriptions of the COFELET key elements and their relationships (i.e., the Objective 3 which tackles the RQ3). The

COFELET ontology elements are employed in the COFELET games life-cycle, a blueprint for developing COFELET games. The COFELET games life-cycle specifies the main components COFELET games contain and the manner the COFELET ontology elements are organized in the structure of such games (i.e., the Objective 3 which tackles the RQ3). To test our approach, we put into effect the COFELET framework, the COFELET ontology, and the COFELET games life-cycle to design and implement the HackLearn hacking simulation game. We assessed how engaging, motivating, and satisfying HackLearn is by incorporating it into the real educational setting of a university class. HackLearn's evaluation showed that HackLearn was a beneficiary addition in a university's class as students were motivated in learning the topics of a cyber security class in a more active and creative way. To determine HackLearn's impact, we examined HackLearn's effectiveness, the degree by which learners were engaged and motivated, and the user satisfaction perceived by the learners. Consequently, the results support the perspective that serious games can improve the effectiveness of cyber security education as they have the potential to transform the learning process from passive and boring to active, motivating, and engaging (addresses the RQ2). Besides, HackLearn is a hacking simulation game that models and interprets the complex system of cyberspace in which cyber-attacks take place. Thus, the presented study provides a proof of concept that any real system can be modeled and interpreted in an organized and parameterized learning environment (e.g., serious game), no matter how complex it is. Such environments are safe for practicing and they allow learners to acquire knowledge and skills while they are entertained.

Nevertheless, the evaluations presented in *chapter 10* revealed two important limitations of this study, which must be highlighted to hint at the paths for future research. Initially, the lack of multilayer support is a restrain, which affects the effectiveness of HackLearn and COFELET based approaches. As stated in *10.2.2* the multiplayer was omitted from the scope of this research as it increased the complexity of the study and made it infeasible. However, after the elaboration of the COFELET framework and the COFELET ontology under the single-player perspective, the study of multiplayer COFELET-based approaches seems viable. Additionally, according to HackLearn's evaluation results, HackLearn's scaffolding façade did not succeed in scaffolding the students' efforts and especially of students who did not do well (discussed in *10.3.7*). Thus, the COFELET framework has to consider more sophisticated scaffolding strategies towards the motivation of learners and the optimization of their performance.

The future work of this study is multi-faced. HackLearn has to be furtherly tested and evaluated in a large scale and in terms of its actual learning effectiveness. Additionally, it has to be extended through the elaboration of scenarios for different learners' roles,

the enrichment of the repository of COFELET ontology key elements, and the upgrade of the scaffolding system with more features. The COFELET framework has to be further examined through the development and evaluation of more COFELET compliant games of various genres (e.g., simulation games, card games) and under the perspectives of adding multiplayer support and enhancing the scaffolding façade. The adaptation characteristic of COFELET games can be furtherly analyzed and studied as well as the degree to which the reuse by which the game foundations of the COFELET games life-cycle (*presented in 7.4*) can facilitate the development of new COFELET games.

PUBLICATIONS

International Journals

I. Katsantonis MN, Mavridis I, Gritzalis D. “Design and Evaluation of COFELET-based Approaches for Cyber Security Learning and Training”. Computers & Security, 105, 102263. Impact Factor: 4.438

II. Katsantonis MN, Mavridis I. “Evaluation of HackLearn COFELET Game User Experience for Cybersecurity Education”, International Journal of Serious Games, Volume 8, Issue 3, September 2021. http://dx.doi.org/10.17083/ijsg.v8i3.437 (accepted for publication)

International Conferences

III. Katsantonis M, Fouliras P, Mavridis I, “Conceptual analysis of cyber security education based on live competitions”. 2017 IEEE Global Engineering Education Conference (EDUCON), Athens, 2017, pp. 771-779. https://doi.org/10.1109/EDUCON.2017.7942934.

IV. Katsantonis MN, Fouliras P, Mavridis I. “Conceptualization of Game Based Approaches for Learning and Training on Cyber Security”. Proceedings of the 21st Pan-Hellenic Conference on Informatics. 2017. https://doi.org/10.1145/3139367.3139415.

V. Katsantonis MN, Kotini I, Mavridis I. “An Innovative Teaching Approach in E-safety Education”. In Proceedings of 9th Conference on Informatics in Education (9th CIE2017), Piraeus, Greece, 2017.

VI. Katsantonis MN, Mavridis I. “Ontology-Based Modelling for Cyber Security E-Learning and Training”. In: Herzog M, Kubincová Z, Han P, Temperini M (eds) Advances in Web-Based Learning - ICWL 2019. ICWL 2019. Lecture Notes in Computer Science, Springer, Cham. vol 11841. https://doi.org/10.1007/978-3-030-35758-0_2.

VII. Katsantonis MN, Kotini I, Fouliras P, Mavridis I. “Conceptual Framework for Developing Cyber Security Serious Games”. In 2019 IEEE Global Engineering Education Conference (EDUCON), 2019 IEEE, pp. 872-881., Dubai, United Arab Emirates, 2019. https://ieeexplore.ieee.org/document/8725061/

REFERENCES

(Abdellatif et al., 2018) A.J. Abdellatif, B. McCollum, P. McMullan. "Serious games: Quality characteristics evaluation framework and case study", 2018 IEEE Integrated STEM Education Conference (ISEC). IEEE, 2018, https://doi.org/10.1109/ISECon.2018.8340460.

(Abt, 1970) Abt, C. C. (1970). "Serious games". New York: Viking Press.

(Adams & Makramalla, 2015) Adams, M., & Makramalla, M. 2015. "*Cybersecurity Skills Training: An Attacker-Centric Gamified Approach*". Technology Innovation Management Review, 5(1): 5-14.

(Allen & Straub, 2015) Allen PD, Straub KA. "Using Games to Enrich Continuous Cyber Training". Johns Hopkins APL Technical Digest. 2015; 33.2.

(Almond et al., 2002) Almond, Russell, Linda Steinberg, and Robert Mislevy. "Enhancing the design and delivery of assessment systems: A four-process architecture." The Journal of Technology, Learning and Assessment 1.5 (2002).

(Amorim et al., 2013) J. A. Amorim, M. Hendrix, S. F. Andler and P. M. Gustavsson. 2013. "Gamified training for cyber defence: Methods and automated tools for situation and threat assessment". In NATO Modelling and Simulation Group (MSG) Annual Conference 2013 (MSG-111).

(Adrion, 1993) W. R. Adrion. "Research methodology in software enginering: summary of the Dagstuhl workshop on future directions on software engineering". SIGSoft Software Engineering Notes, 18(1):36{37, 1993.

(Anneta, 2010) L. A. Annetta. 2010. "The 'I's' have it: A framework for serious educational game design". Review of General Psychology 14.2: 105.

(Arnab et al., 2015) S. Arnab, T. Lim, M. B. Carvalho, F. Bellotti, S. De Freitas, S. Louchart, N. Suttie, R. Berta and A. De Gloria. 2015. "Mapping learning and game mechanics for serious games analysis". British Journal of Educational Technology, 46(2), 391-411.

(Ausubel, 2000) Ausubel DP. The Acquisition and Retention of Knowledge: A Cognitive View. Dordrecht: Springer - science + business media: 2000.

(Bedwell et al., 2012) Bedwell, Wendy L., et al. "Toward a taxonomy linking game attributes to learning an empirical study." Simulation & Gaming 43.6 (2012): 729-760.

(Behrens et al., 2007) Behrens, J. T., Frezzo, D. C., Mislevy, R. J., Kroopnick, M., & Wise, D. (2007). "Structural, functional, and semiotic symmetries in simulation-based

games and assessments". Assessment of problem solving using simulations, 59-80. New York: Erlbaum.

(Bratosin, 2014) B.A. Bratosin, "Cyber Defense Exercises and their Role in Cyber Warfare", Journal of Mobile, Embedded and Distributed Systems 6, no. 2, 2014, 70-76.

(Breuer & Bente, 2010) Breuer, J., & Bente, G. (2010). "Why so serious? On the relation of serious games and learning". Journal for Computer Game Culture, 4, 7-24.

(Buckley & Caple, 2009) R. Buckley and J. Caple. 2009. "The theory and practice of training". Kogan Page Publishers, 2009.

(CAPEC, 2021a) The MITRE Corporation, Common Attack Pattern Enumeration and Classification (CAPEC), https://capec.mitre.org/, Retrieved 27 August 2021.

(CAPEC, 2021b) The MITRE Corporation, CAPEC and ATT&CK Comparison https://capec.mitre.org/about/attack_comparison.html, Retrieved 27 August 2021.

(Carvalho et al., 2015) M. B. Carvalho, F. Bellotti, R. Berta, A. De Gloria, C. I. Sedano, J. B. Hauge, J. Hu and M. Rauterberg, "An activity theory-based model for serious games analysis and conceptual design", Computers & education, 87, 166-181, 2015.

(Carlisle et al., 2015) M. Carlisle, M. Chiaramonte and D. Caswell, "Using CTFs for an Undergraduate Cyber Education". In 2015 USENIX Summit on Gaming, Games, and Gamification in Security Education (3GSE 15). 2015.

(Catuogno & De Santis, 2008) L. Catuogno and A. De Santis, "An internet role-game for the laboratory of network security course". In ACM SIGCSE Bulletin, vol. 40, no. 3, pp. 240-244, ACM, 2008.

(Čeleda et al., 2015) P. Čeleda, J. Čegan, J. Vykopal and D. Tovarňák. 2015. "KYPO–A Platform for Cyber Defence Exercises". M&S Support to Operational Tasks Including War Gaming, Logistics, Cyber Defence. NATO Science and Technology Organization.

(Caltagirone et al., 2013) S. Caltagirone, A. Pendergast, and Christopher Betz. "The diamond model of intrusion analysis". Center For Cyber Intelligence Analysis and Threat Research Hanover Md, 2013.

(Childers et al., 2010) Childers, N., Boe, B., Cavallaro, L., Cavedon, L., Cova, M., Egele, M., and Vigna, G., "Organizing large scale hacking competitions". In International Conf. on Detection of Intrusions and Malware, and Vulnerability Assessment, pp.132-152, Springer Heidelberg, 2010.

(Cheung et al., 2012) R.S. Cheung, J.P Cohen, H.Z. Lo, F. Elia, and V. Carrillo-Marquez, "Effectiveness of cybersecurity competitions". In Proceedings of the International Conference on Security and Management (SAM), p. 1, The Steering Committee of The

World Congress in Computer Science, Computer Engineering and Applied Computing (WorldComp), 2012.

(Chung & Cohen, 2014) K. Chung, and J. Cohen, “Learning Obstacles in the Capture The Flag Model”. In 2014 USENIX Summit on Gaming, Games, and Gamification in Security Education (3GSE 14), 2014.

(Chothia & Novakovic, 2015) T. Chothia and C. Novakovic, “An Offline Capture The Flag-Style Virtual Machine and an Assessment of its Value for Cybersecurity Education”. In 2015 USENIX Summit on Gaming, Games, and Gamification in Security Education (3GSE 15), 2015.

(Csikszentmihalyi, 1997) Csikszentmihalyi, M. (1997). “Finding flow”. New York, NY. Basic Books.

(Caltagirone, et al., 2013) S. Caltagirone, A. Pendergast, and C. Betz, “The Diamond Model of Intrusion Analysis”. Threat Connect, vol. 298, no. 0704, pp. 1–61, 2013.

(Compte et al., 2014) A. Le Compte, T. Watson and D. Elizondo. 2014. “Serious Games: A design methodology from concept to end-user”. In Proceedings of the Sixth International Conference on Virtual Worlds and Games for Serious Applications: VS-Games.

(Compte et al., 2015) A. Le Compte, T. Watson and D. Elizondo. 2015. “A renewed approach to serious games for cyber security”. Cyber Conflict: Architectures in Cyberspace (CyCon), 2015 7th International Conference on. IEEE.

(Ctftime.org, 2016) C.team, “Ctftime.org / all about ctf (capture the flag),” 2016. [Online]. Available: https://ctftime.org/ctf-wtf/

(Dabrowski et al., 2015) A. Dabrowski, M. Kammerstetter, E. Thamm, E. Weippl, and W. Kastner, “Leveraging Competitive Gamification for Sustainable Fun and Profit in Security Education”. In 2015 USENIX Summit on Gaming, Games, and Gamification in Security Education (3GSE 15), 2015.

(Davis, 1989) F.D. Davis, “Perceived usefulness, perceived ease of use, and user acceptance of information technology”. MIS quarterly: 319-340, 1989.

(Davis et al., 2014) A. Davis, T. Leek, M. Zhivich, K. Gwinnup, and William Leonard, “The Fun and future of CTF”. In 2014 USENIX Summit on Gaming, Games, and Gamification in Security Education (3GSE 14), 2014.

(Dele-Ajayi, 2016) O. Dele-Ajayi, J. Sanderson, R. Strachan, A. Pickard, “Learning mathematics through serious games: An engagement framework”. In 2016 IEEE Frontiers in Education Conference (FIE) (pp. 1-5). IEEE, October 2016

(Dewey, 1933) Dewey J. “How We Think: A Restatement of the Relation of Reflective Thinking to the Educative Process”. Boston, Massachusetts: D.C. Heath & Co Publishers: 1933.

(Doupé et al., 2011) A. Doupé, M. Egele, B. Callait, G. Stringhini, G. Yakin, A. Zand, L. Cavedon, G. Vigna, “Hit'em where it hurts: a live security exercise on cyber situational awareness”. Proceedings of the 27th Annual Computer Security Applications Conference, ACM, 2011.

(Fallon and Brown, 2016) Fallon, C., and Brown, S. “E-learning standards a guide to purchasing, developing, and deploying standards-conformant e-learning”. CRC Press LLC, U.S. (19/4/2016).

(De Castell & Jenson, 2003) De Castell, S., & Jenson, J. (2003). “OP‐ED Serious play”. Journal of Curriculum Studies, 35(6), 649‐665. https://doi.org/10.1080/0022027032000145552

(De Freitas & M. Oliver, 2006) S. De Freitas and M. Oliver. 2006. “How can exploratory learning with games and simulations within the curriculum be most effectively evaluated?”. Computers & education 46.3 (2006): 249-264.

(De Freitas & Liarokapis, 2011) De Freitas, Sara, and Fotis Liarokapis. “Serious Games: A New Paradigm for Education?”. Serious games and edutainment applications. Springer London, 2011. 9-23.

(Elborji & Mohamed, 2014) Elborji, Yassine, and Mohamed Khaldi. “An IEEE LOM Application Profile to Describe Serious Games «SG-LOM»”. International Journal of Compute Applications 86.13 (2014): 1-8.

(Garris et al., 2002) Garris, R., Ahlers, R., & Driskell, J. (2002). “Games, motivation, and learning: A research and practice model”. Simulation & Gaming, 33(4), 441–467.

(Glass, 1995) R. L. Glass. “A structure-based critique of contemporary computing research”. Journal of Systems and Software, 28(1):3-7, 1995

(Greitzer et al., 2007) F. L. Greitzer, O. A. Kuchar, and K. Huston, “Cognitive science implications for enhancing training effectiveness in a serious gaming context”, Journal on Educational Resources in Computing (JERIC) 7.3 (2007): 2, 2007.

(Haney & Lutters, 2017) J. M. Haney & W. G. Lutters, “Skills and characteristics of successful cybersecurity advocates”. In Proc. of the 13th Symposium on Usable Privacy and Security, ser. SOUPS (Vol. 17), 2017.

(Hendrix et al., 2012) Hendrix, Maurice, et al. “Defining a metadata schema for serious games as learning objects”. eLmL 2012, The Fourth International Conference on Mobile, Hybrid, and On-line Learning. 2012.

(Hendrix et al., 2016) M. Hendrix, A. Al-Sherbaz, and V. Bloom. 2016. “Game based cyber security training: are serious games suitable for cyber security training?”. International Journal of Serious Games 3.1: 53-61.

(Hoffman et al., 2005) L.J. Hoffman, T. Rosenberg, R. Dodge and D. Ragsdale, “Exploring a national cybersecurity exercise for universities”. IEEE Security & Privacy 3, no. 5 (2005): 27-33.

(Hunicke et al., 2004) R. Hunicke, M. LeBlanc, and R. Zubek, “MDA: A formal approach to game design and game research,” in Proceedings of the AAAI Workshop on Challenges in Game AI, 2004, vol. 4.

(Johnson et al., 2015) B. Johnson, A. Laszka, and J. Grossklags, “Games of timing for security in dynamic environments”, In International Conference on Decision and Game Theory for Security, pp. 57-73, Springer, Cham, 2015.

(Jonassen & Rohrer-Murphy, 1999) Jonassen DH, Rohrer-Murphy L. “Activity theory as a framework for designing constructivist learning environments”. Educational technology research and development. 1999;47.1:61-79.

(Joosten-ten Brinke et al., 2007) Joosten-ten Brinke, D., Van Bruggen, J., Hermans, H., Burgers, J., Giesbers, B., Koper, R., and Latour, I. “Modeling assessment for re-use of traditional and new types of assessment”. Computers in Human Behavior, 23(6):2721–2741, 2007.

(Kapp, 2012) Kapp, Karl M. “The gamification of learning and instruction: game-based methods and strategies for training and education”. John Wiley & Sons, 2012.

(Katsantonis et al., 2017a) Katsantonis M, Fouliras P, Mavridis I, “Conceptual analysis of cyber security education based on live competitions”. 2017 IEEE Global Engineering Education Conference (EDUCON), Athens, 2017, pp. 771-779. https://doi.org/10.1109/EDUCON.2017.7942934.

(Katsantonis et al., 2017b) Katsantonis MN, Fouliras P, Mavridis I. “Conceptualization of Game Based Approaches for Learning and Training on Cyber Security”. Proceedings of the 21st Pan-Hellenic Conference on Informatics. 2017. https://doi.org/10.1145/3139367.3139415.

(Katsantonis & Mavridis, 2019) Katsantonis MN, Mavridis I. “Ontology-Based Modelling for Cyber Security E-Learning and Training”. In: Herzog M, Kubincová Z, Han P, Temperini M (eds) Advances in Web-Based Learning - ICWL 2019. ICWL 2019. Lecture Notes in Computer Science, Springer, Cham. vol 11841. https://doi.org/10.1007/978-3-030-35758-0_2.

(Katsantonis et al., 2019) Katsantonis MN, Kotini I, Fouliras P, Mavridis I. “Conceptual Framework for Developing Cyber Security Serious Games”. In 2019 IEEE Global

Engineering Education Conference (EDUCON), 2019 IEEE, pp. 872-881., Dubai, United Arab Emirates, 2019. https://ieeexplore.ieee.org/document/8725061/

(Katsantonis et al., 2021) Katsantonis MN, Mavridis I, Gritzalis D. “Design and Evaluation of COFELET-based Approaches for Cyber Security Learning and Training”. Computers & Security, 105, 102263.

(Katsantonis & Mavridis, 2021) Katsantonis MN, Mavridis I. “Evaluation of HackLearn COFELET Game User Experience for Cybersecurity Education”, International Journal of Serious Games 8.3, 2021.

(Konak et al., 2015) A. Konak, T.K. Clark and M. Nasereddin, “Using Kolb's Experiential Learning Cycle to improve student learning in virtual computer laboratories”. Computers & Education 72 (2014): 11-22.

(Koch et al., 2012) S. Koch, J. Schneider and J. Nordholz, “Disturbed Playing: Another Kind of Educational Security Games”. In Proceedings of the 5th USENIX conference on Cyber Security Experimentation and Test, USENIX Association, Berkeley, CA, USA, 2012.

(Lopes & Bidarra, 2011) R. Lopes, and R. Bidarra, “Adaptivity challenges in games and simulations: a survey”. IEEE Transactions on Computational Intelligence and AI in Games, 3(2), 85-99. 2011.

(Martin, 2014) L. Martin, “Cyber Kill Chain”, URL: http://cyber. lockheedmartin. com/hubfs/Gaining_the_Advantage_Cyber_Kill _Chain.pdf, 2014.

(Martini & Choo, 2014) B. Martini and K.K.R. Choo, “Building the Next Generation of Cyber Security Professionals”. Proceedings of Twenty Second European Conference on Information Systems, Tel Aviv, 2014.

(Madsen, 2018) Madsen, Kristina. (2018). “The Gamified Museum - A critical literature review and discussion of gamification in museums”. In Gamescope: The potential for gamification in digital and analogue places, Aalborg Universitetsforlag, 2020.

(Martens & Müller, 2017) Martens, A. and Müller, W. “Gamification”. Handbook of Digital Games and Entertainment Technologies. Springer, 2017.

(Mauer et al., 2012) B. Mauer, W. Stackpole and D. Johnson, “Developing Small Team-based Cyber Security Exercises”. In Proceedings of the International Conference on Security and Management (SAM), p. 1, The Steering Committee of The World Congress in Computer Science, Computer Engineering and Applied Computing (WorldComp), 2012.

(Mayer, 2012) I. Mayer. 2012. “Towards a comprehensive methodology for the research and evaluation of serious games”. Procedia Computer Science 15 (2012): 233-247.

(McClain et al., 2015) J. McClain, A. Silva, G. Emmanuel, B. Anderson, K. Nauer, R. Abbott, and C. Forsythe, “Human performance factors in cyber security forensic analysis”. Procedia Manufacturing 3, 2015, pp.5301-5307.

(Mirkovic & Benzel, 2012) J. Mirkovic and T. Benzel, “Teaching cybersecurity with DeterLab”. IEEE Security & Privacy 10, no. 1, pp. 73-76, 2012.

(Michael & Chen, 2006) Michael, D. and Chen, S. “Serious Games: Games That Educate”. Train and Inform. Boston: Thomson, 2006.

(Mirkovic & Peterson, 2014) J. Mirkovic, & P.A. Peterson, “Class capture-the-flag exercises”. In 2014 USENIX Summit on Gaming, Games, and Gamification in Security Education (3GSE 14), 2014.

(Mirkovic et al., 2015) J. Mirkovic, A. Tabor, S. Woo, and P. Pusey, “Engaging Novices in Cybersecurity Competitions: A Vision and Lessons Learned at ACM Tapia 2015”. In 2015 USENIX Summit on Gaming, Games, and Gamification in Security Education (3GSE 15), 2015.

(Mischel, 1971) T. Mischel, “Piaget: Cognitive conflict and the motivation of thought”. Cognitive development and epistemology: 311-355, 1971.

(Mislevy et al., 2003) Mislevy, R. J., Steinberg, L. S., & Almond, R. G. “On the structure of educational assessment”. Measurement: Interdisciplinary Research and Perspective, 1(1), 3–62, 2003.

(Nagarajan et al., 2012) Nagarajan A, Allbeck JM, Sood A, Janssen TL. “Exploring game design for cybersecurity training”. Cyber Technology in Automation, Control, and Intelligent Systems (CYBER), 2012 IEEE International Conference on. IEEE. 2012.

(Newhouse et al., 2017) Newhouse W, Keith S, Scribner B, Witte G. “National Initiative for Cybersecurity Education (NICE) - Cybersecurity Workforce Framework”. National Institute of Standards and Technology (NIST) Special Publication. 2017; 800, 181.

(Nilüfer et al., 2018) B. Nilüfer, A. Löffler, R. Heininger, M. Utesch, H. Krcmar, “Evaluation Methods for the Effective Assessment of Simulation Games”. In International Conference on Interactive Collaborative Learning, pp. 626-637, Springer, Cham, 2018. https://doi.org/10.1007/978-3-030-11932-4_59.

(Noy & McGuinness, 2001) Noy, N., F., and McGuinness, D., L. “Ontology development 101: A guide to creating your first ontology”, 2001.

(Parker & Becker, 2013) Parker, J. R., & Becker, K. “The simulation-game controversy: What is a ludic simulation?”. International Journal of Gaming and Computer-Mediated Simulations (IJGCMS), 5(1), 1-12, 2013.

(Piaget, 1952) J. Piaget. 1952. “The origins of intelligence”. Madison, CT: International Universities Press.

(Poltrack, 2014) Poltrack, J. “ADL Training & Learning Architecture (TLA)”. http://www.adlnet.gov/wp-content/uploads/2014/07/ADL-Training-and-LearningArchitecture-1.pdf, last accessed 30/5/2019.

(Poortvliet & Darnon, 2010) Poortvliet, P. M., & Darnon, C. (2010). “Toward a more social understanding of achievement goals: The interpersonal effects of mastery and performance goals”. Current Directions in Psychological Science, 19(5), 324–328. https://doi.org/10.1177/0963721410383246

(Protégé, 2021) Protégé Stanford, http://protege.stanford.edu, last accessed 30/09/2021.

(Prensky, 2005) Prensky, Marc. “Computer games and learning: Digital game-based learning”. Handbook of computer game studies 18 (2005): 97-122.

(Pusey et al., 2014) P. Pusey, D. Tobey Sr and R. Soule. “An Argument for Game Balance: Improving Student Engagement by Matching Difficulty Level with Learner Readiness”. In 2014 USENIX Summit on Gaming, Games, and Gamification in Security Education (3GSE 14), 2014.

(Rothwell & Kazanas, 1997) Rothwell, W. J., & Kazanas, H. C. “Mastering the instructional design process: A systematic approach”. San Francisco: Jossey-Bass, 1997.

(Rege, 2015) A. Rege. “Multidisciplinary Experiential Learning for Holistic Cybersecurity Education, Research and Evaluation”. In USENIX Summit on Gaming, Games, and Gamification in Security Education (3GSE 15), 2015.

(Riconscente, 2015) Riconscente, Michelle M., Robert J. Mislevy and Seth Corrigan. "Evidence-Centered Design". In Handbook of Test Development ed. Suzanne Lane , Mark R. Raymond and Thomas M. Haladyna (Abingdon: Routledge, 02 Nov 2015), accessed 09 May 2021 , Routledge Handbooks Online.

(Salen & Zimmerman, 2004) Salen, Katie, and Eric Zimmerman. “Rules of play: Game design fundamentals”. MIT press, 2004.

(Schell, 2008) Schell, J. “The Art of Game Design: A Book of Lenses”. Burlington, MA." Elsevier, 2008.

(Sessa & London, 2015) Sessa VI, London M. “Continuous learning in organizations: Individual, group, and organizational perspectives”. Psychology Press, 2015.

(Silva et al., 2014) A. Silva, J. McClain, T. Reed, B. Anderson, K. Nauer, R. Abbott and C. Forsythe, “Factors impacting performance in competitive cyber exercises”. Proceedings of the Interservice/Interagency Training, Simulation and Education Conference, Orlando FL, 2014.

(Smith et al., 2015) S.P. Smith, K. Blackmore, K. Nesbitt. “A Meta-Analysis of Data Collection in Serious Games Research”. In: Loh C., Sheng Y., Ifenthaler D. (eds) Serious Games Analytic, Advances in Game-Based Learning. Springer, Cham, 2015. https://doi.org/10.1007/978-3-319-05834-4_2.

(Strom et al., 2018) B.E. Strom, A. Applebaum, D.P. Miller, K.C. Nickels, A.G. Pennington, C.B. Thomas, “Mitre ATT&CK: Design and philosophy”, Technical report, 2018.

(Susi et al., 2007) Susi, T., Johannesson, M. and P. Backlund. “Serious Games - An Overview”. Technical Report HS- IKI -TR-07-001, School of Humanities and Informatics, University of Skövde, Sweden (2007).

(Tang et al., 2009) Tang, Stephen, Martin Hanneghan, and Abdennour El Rhalibi. “Introduction to games-based learning”. Games-based learning advancements for multi-sensory human computer interfaces: Techniques and effective practices. IGI Global, 2009. 1-17.

(Taylor, 2014) AS. A. Taylor. “Facilitation matters: A framework for instructor-led serious gaming”. Ph.D. Dissertation. University of Skövde, 2014.

Toulmin, S. E. The uses of argument. Cambridge, UK: Cambridge University Press, 1958.

(Utesch, 2016) M.C. Utesch. “A Successful Approach to Study Skills: Go4C's Projects Strengthen Teamwork”. International Journal of Engineering Pedagogy, 6(1), 2016.

(Vandewaetere et al., 2013) M. Vandewaetere, F. Cornillie, G. Clarebout, P. Desmet, “Adaptivity in Educational Games: Including Player & Gameplay Characteristics”. International Journal of Higher Education, 2(2), 106-114., 2013.

(Vigna et al., 2014) G. Vigna, K. Borgolte, J. Corbetta, A. Doupe, Y. Fratantonio, L. Invernizzi, D. Kirat and Y. Shoshitaishvili. “Ten years of ictf: The good, the bad, and the ugly”. In 2014 USENIX Summit on Gaming, Games, and Gamification in Security Education (3GSE 14), 2014.

(Vykopal & Barták, 2016) J. Vykopal, and M. Barták. “On the Design of Security Games: From Frustrating to Engaging Learning”. In ASE@ USENIX Security Symposium, August 2016.

(Vygotsky, 1978) Vygotsky LS. “Mind in society: The development of higher psychological processes”. Cambridge, Massachusetts: Harvard University Press; 1978.

(Werther et al., 2011) J. Werther, M. Zhivich, T. Leek, and N. Zeldovich. “Experiences in cyber security education, The mit lincoln laboratory capture-the-flag exercise”. In

the 4th Workshop on Cyber Security Experimentation and Test, San Francisco, CA, United states, 2011.

(Wills et al., 2009) Wills, G., Bailey, C., Davis, H., Gilbert, L., Howard, Y., Jeyes, S., Millard, D., Price, J., Sclater, N., Sherratt, R., Tulloch, I., and Young, R. “An E-Learning Framework for Assessment (FREMA)”. Assessment & Evaluation in Higher Education, 34(3): 273–292, 2009.

(Winn, 2008) B. Winn. “The design, play, and experience framework”. Handbook of research on effective electronic gaming in education 3: 1010-1024, 2008.

(Zhang et al., 2004) Zhang, D., Zhao, J. L., Zhou, L., & Nunamaker Jr, J. F. “Can e-learning replace classroom learning?”. Communications of the ACM, 47(5), 75-79, 2004.

(Zyda, 2005) Zyda, M. “From Visual Simulation to Virtual Reality to Games”. Computer, Vol. 38, No. 9, pp 25-32, 2005.