Design and Practice Exploration of Software Security Experimental Teaching

Fangfang Xu*, Ziqi Zhu, Feng Gao, Xiao Xie, Yaojie Chen

School of Computer of Science and Technology, Wuhan University of Science and Technology, Wuhan, Hubei, 430081, China

*Corresponding author: xuff@wust.edu.cn

Abstract. In order to cultivate students' comprehensive ability and innovative thinking, meet the requirements of engineering practice ability under the new engineering background, and solve the difficulties faced by software experiment teaching at present, combining the advantages of traditional teaching, this paper puts forward new ideas and new ideas for software safety experiment teaching design and practice. The combination of independent experiment and comprehensive experiment, classification teaching, the introduction of competition mechanism heuristic teaching, using online experimental teaching platform, focusing on process assessment and other aspects are implemented into the experimental teaching process. Finally, the subsequent teaching improvement suggestions are put forward.

Keywords: Software security; Experimental teaching; Information security

1 INTRODUCTION

Currently, society's dependence on information technology is becoming increasingly significant. With the rapid growth of information volume, the complexity of software development and integration is rising, leading to serious vulnerabilities in software products. In the field of information security, software security is a core course, and its experimental teaching component is especially important [1]. At present, many universities have offered this course, which plays a crucial role in helping students understand and master security practices as well as secure programming techniques, enhancing their innovation and practical skills [2,3].

In this context, how to effectively design and implement software security experimental teaching to cultivate students' problem-solving abilities and innovative capabilities has become an important issue faced by educators [4-5]. The purpose of this paper is to explore the design and practice of software security experimental teaching.

In the "Guiding Professional Standards for Information Security," the research directions of the information security discipline are divided into several areas, including cryptography, network security, information system security, and information content security. Software security, as an important component of information system security, is one of the core courses in the information system security program. The main goal of the software security course is to help students understand software security threats and their root causes, as well as to master the basic concepts related to software security, including the detection and protection technologies for malicious code. This course imposes high demands on students' foundational knowledge and practical skills, and due to the rapid pace of technological advancements, the course content also needs to remain current and relevant [6].

However, there are still some shortcomings in the experimental teaching component of this course:

(1) Students Lack the Initiative to Comprehensively Apply the Foundational Knowledge They Have Learned.

In the preliminary foundational courses, only three security-related courses are involved: "Introduction to Cyberspace Security," "Mathematical Foundations of Information Security," and "Cryptography." These courses are less targeted toward software security, leading some students to perceive that the foundational courses have little practical value for software security. Students often adopt a passive attitude in software security experimental teaching, typically receiving tasks assigned by instructors without actively integrating and applying the foundational knowledge they have learned. For example, during the software security experiment on "Buffer Overflow Vulnerability Analysis and Exploitation," students frequently find it challenging to organically combine foundational knowledge such as operating system principles and network protocols with the experimental content, resulting in insufficient understanding and problem-solving abilities during practical operations.Therefore, it is particularly important to design basic software security experiments that help students organize their knowledge of information security and guide them to proactively apply the foundational knowledge they have learned to address software security issues.

(2) Students Lack the Enthusiasm to Leverage Foundational Knowledge to Reverse-engineer Underlying Mechanisms.

In software security experimental teaching, students often focus solely on superficial security vulnerabilities and simple defense measures, lacking a deep understanding of fundamental knowledge such as operating system principles, programming languages, and network protocols. This limitation hinders their ability to effectively reverse-engineer underlying mechanisms. For example, when conducting the "Buffer Overflow Vulnerability Analysis and Exploitation" experiment, if students have not firmly grasped foundational concepts like memory management, function call conventions, and their implementation in compilers, they are unable to identify the root causes of attacks and exploit vulnerabilities effectively. Furthermore, students tend to rely excessively on existing security tools to find and identify vulnerabilities, lacking systematic analytical skills. This restricts their ability to thoroughly analyze potential security risks, resulting in their practical skills remaining at the level of tool usage .

(3) The Evaluation and Assessment Method does not Meet the Needs of the New Engineering Disciplines.

The current evaluation and assessment methods tend to emphasize theoretical knowledge while neglecting practical skills, making it difficult to comprehensively assess students' abilities and performance in practical operations. In experimental teaching, the evaluation often relies solely on a single report format, which fails to adequately reflect students' overall performance and innovative capabilities during experiments. Furthermore, assessments typically focus only on individual performance, not effectively capturing students' teamwork and communication skills. In summary, the existing evaluation and assessment system does not meet the needs for cultivating students' comprehensive abilities and innovative thinking in the context of new engineering disciplines.

2 EXPERIMENTAL TEACHING CONTENT DESIGN

2.1 Basic Objectives of Experimental Teaching Content Design

Through years of software security teaching practice, considering students' actual situations as well as industry talent demands, our software security experimental teaching primarily revolves around three objectives:

(1) Cultivating the Ability to Integrate Foundational Knowledge with Practice. The content of the experimental teaching is closely linked to the theoretical courses, designing foundational experimental projects that are relevant to the curriculum. Additionally, real software security cases are introduced to design comprehensive experimental projects based on actual problems, thereby enhancing students' practical abilities.

(2) Stimulating Students' Initiative and Creativity. By designing exploratory, open-ended, and challenging experimental projects, we provide students with a certain degree of choice and freedom, encouraging them to explore and innovate independently during experiments. This approach aims to foster students’ creative and critical thinking, thereby improving their ability to solve complex problems.

(3) Developing Students' Teamwork and Communication Skills. By designing group collaborative experimental tasks and introducing teamwork dynamics, we guide students to enhance their communication and collaboration skills within a team, helping them learn to work effectively with others.

2.2 Experimental Teaching Content

In alignment with the objectives of software security experimental teaching, the content of software security experiments is categorized into six types: foundational experiments, principle verification experiments, comprehensive exploration experiments, innovative experiments, competition experiments, and practical application experiments.

The specific teaching objectives and content for these experiments are presented in Table 1.

Table 1. Experimental Teaching Content

Experiment Type	Purpose	Content
Foundational Experiments	Understand the basic knowledge of software security.	Analysis of insecure code and writing secure code, buffer overflow experiments, SQL injection experiments, XSS experiments.

Principle Verification Experiments	Understand the core principles of software security.	Encryption algorithm experiments, access control mechanism experiments, intrusion detection system (IDS) experiments, malware analysis experiments.
Comprehensive Exploration Experiments	Cultivate comprehensive analytical and problem-solving abilities.	Double free vulnerability experiments, analysis of vulnerabilities in complex applications, combined dynamic and static analysis experiments, malware family classification experiments.
Innovative Experiments	Cultivate innovative thinking and the ability to develop new technologies.	Exploration of new technologies: Investigating the application of cutting-edge technologies (e.g., blockchain, machine learning) in software security, requiring students to design experiments for validation. Project Development: Collaborative group development of a highly secure software application, requiring assessment of potential security risks and corresponding design measures.
Competition Experiments	Develop practical skills and teamwork abilities.	Capture The Flag (CTF) competitions, penetration testing competitions, defense strategy competitions, attack-defense confrontation experiments, practical drills and simulation competitions, enterprise collaboration competitions.
Practical Application Experiments	Apply theoretical knowledge to real-world scenarios to deepen understanding.	Collaborating with enterprises to conduct experiments related to actual projects, helping students understand the industry's real needs and solutions for software security.

3 IMPLEMENTATION OF EXPERIMENTAL TEACHING

The purpose of experimental teaching is not only to help students understand and master theoretical knowledge but also to focus on cultivating their practical abilities and innovative spirit. To enhance students' learning capabilities, analytical skills, and ability to apply knowledge comprehensively, as well as to stimulate their interest in learning and uncover their potential, a variety of teaching methods are employed during the instructional process, such as combining independent experiments with comprehensive experiments. This approach aims to guide students to value the development of their ability to apply knowledge and their capacity for independent innovation.

3.1 Combining Independent Experiments with Comprehensive Experiments

Independent experiments focus on training specific knowledge, while comprehensive experiments emphasize the integration of multiple knowledge points and the ability to solve practical problems. However, these two should maintain a moderately loose coupling; otherwise, failures in preliminary experiments can negatively impact subsequent experiments, leading to a decline in students' interest and confidence. Therefore, both types of experiments need to be designed holistically to achieve effective integration of independent and comprehensive experiments.

For example, first, design an independent experiment focused on a single topic: "Learning and Using Tools to Analyze the Panda Burning Incense Malware," providing detailed steps and operational guidelines to ensure students can complete it independently. Following that, a comprehensive experiment can be designed: "Thoroughly Analyzing a Complex Malware Sample," which involves static analysis, dynamic analysis, and behavioral detection, and requires students to propose defense strategies.

Years of experimental teaching practice have demonstrated that this approach, moving from simple to complex and from individual to comprehensive, allows students not only to solidify their understanding of foundational knowledge and skills but also to apply these knowledge and skills to complex real-world problems, thereby expanding their ability to apply knowledge and enhancing their problem-solving skills.

3.2 Categorized Teaching

While ensuring the fundamental teaching tasks are met, students are grouped based on their varying levels and learning needs, with corresponding teaching content and methods designed and implemented. Students are divided into three groups according to their levels: the Basic Group, Intermediate Group, and Advanced Group. Based on their learning needs, they are further classified into: the Instructional Group and the Competition Group. Given the limited teaching resources and time, different levels of difficulty and depth of experimental content and resources are provided according to the grouping results.

Teaching is conducted separately based on the grouping results, offering varying levels of difficulty and depth in the experimental content. Personalized guidance is provided for different student groups, with additional support for the Basic Group and encouragement for the Advanced Group and Competition Group to take on more challenging experimental tasks.

The specific categorized teaching strategies are shown in Table 2.

Table 2. Categorized Teaching Strategies

Group	Teaching Content	Guidance	Self-Study	Required	Optional	Experiment Difficulty
Basic Group	Foundational Experiments	√		√		Easy
	Principle Verification Experiments	√		√		Easy
	Comprehensive Exploration Experiments	√			√	Moderate
	Innovative Experiments	√			√	Moderate
	Competition Experiments	√			√	Challenging
	Practical Application Experiments	√			√	Challenging
Intermediate	Foundational Experiments		√	√		Easy

Group	Principle Verification Experiments	√		√		Moderate
	Comprehensive Exploration Experiments	√		√		Moderate
	Innovative Experiments	√			√	Challenging
	Competition Experiments	√			√	Challenging
	Practical Application Experiments	√			√	Very Challenging
Advanced Group	Foundational Experiments		√	√		Easy
	Principle Verification Experiments		√	√		Moderate
	Comprehensive Exploration Experiments	√		√		Challenging
	Innovative Experiments	√		√		Challenging
	Competition Experiments	√		√		Very Challenging
	Practical Application Experiments	√		√		Very Challenging
Instructional Group	Foundational Experiments	√		√		Easy
	Principle Verification Experiments	√		√		Easy
	Comprehensive Exploration Experiments	√		√		Moderate
	Innovative Experiments	√			√	Challenging
	Competition Experiments	√			√	Very Challenging
	Practical Application Experiments	√			√	Very Challenging
Competition Group	Foundational Experiments		√		√	Easy
	Principle Verification Experiments		√		√	Moderate
	Comprehensive Exploration Experiments		√		√	Challenging
	Innovative Experiments	√			√	Very Challenging
	Competition Experiments	√		√		Very Challenging
	Practical Application Experiments	√		√		Very Challenging

Teaching practice has demonstrated that the categorized teaching model can provide personalized learning experiences tailored to students' varying levels and needs, thereby enhancing teaching effectiveness. This approach more effectively stimulates students' interest in learning and initiative, while avoiding the frustration caused by disparities in learning abilities.

3.3 Guided Teaching with the Introduction of a Competitive Mechanism

The knowledge system and technological advancements in software security are rapidly evolving, making it difficult for educational institutions to provide comprehensive instruction. Therefore, by introducing a competitive mechanism into heuristic teaching, the aim is to cultivate students' interest in exploring new technologies and tools, increase the diversity of problem-solving approaches, and enhance students' initiative. In practice, two main teaching methods are employed:First, instructors set different topics based on the experimental content and current hot issues in software security both domestically and internationally, providing relevant reference materials. Students are guided to work in groups to discuss and research these topics. Each group is required to present and submit a research report. A judging panel is established to score the presentations based on their logical coherence and clarity, with the scores contributing to the overall assessment.Second, students are encouraged to actively explore and analyze real-world issues related to software security challenges encountered in their learning and daily environments, such as smart wearable devices and mobile terminals. Students analyze application software to extract vulnerability information, and instructors provide corresponding scores based on the vulnerability analysis reports submitted by the students. Scores can be accumulated, and instructors will regularly update and publicly display the total scores of the students in the class, which will also be factored into the final assessment.

Multiple practices in experimental teaching have shown that introducing a competitive mechanism into heuristic teaching effectively stimulates students' enthusiasm for learning, increases their participation, and enhances their practical skills and innovative thinking. Additionally, this approach fosters teamwork and communication skills among students. Furthermore, in a competitive environment, students also develop their ability to handle pressure and adapt to changing circumstances.

3.4 Online Experimental Teaching Platform

The online experimental teaching platform, Istudy, has been established to provide a virtual laboratory for remote experiments. It also allows for online resource sharing, submission and assessment of experiment reports, and online Q&A sessions.

With the online teaching platform, students can access it at any time, learn at their own pace, and repeat experiments as needed. The online platform provides students with an isolated virtual experiment environment, mitigating the risk of virus transmission and ensuring the safety and control of experiments. Additionally, the platform records the students' experimental processes and results, offering detailed analytical data for teachers to assess student performance in real time. This online platform creates an interactive, open, and efficient learning environment for students, significantly enhancing the effectiveness of the teaching process.

3.5 Reform of Assessment Processes

Compared to traditional assessment methods, our evaluation emphasizes process assessment and places greater importance on students' overall abilities. The final grade consists of three components: attendance, performance during the experiment, and the experimental report:

(1) Attendance accounts for 10% of the final grade and includes students' attendance records to ensure they participate in experimental teaching sessions on time.

(2) Performance during the experiment constitutes 70% of the final grade and encompasses students' analytical abilities in problem-solving during the experiment, group defense scores, and accumulated points for submitted vulnerability information.

(3) The experimental report represents 30% of the final grade and includes the experiment steps, results and analysis, and reflective questions related to the experiment.

Teaching practices have shown that this assessment method not only provides a comprehensive evaluation of students' abilities and encourages continuous learning but also supports students' individualized development.

4 SUBSEQUENT DEVELOPMENT IDEAS

In the subsequent development of the experimental courses, we will first incorporate past competition problems into the planning of experimental projects based on the progress of the theoretical courses, analyzing the problem-solving approaches from these competitions to enhance students' abilities and interest in participating in competitions. At the same time, we will actively apply for and host information security competitions to provide students with opportunities for face-to-face interactions with top talents from across the country.

Beside, we will promote the latest events and security technologies in the field of software security through the online experimental teaching platform. By actively collaborating with companies such as Qihoo 360, Tencent, and Xiaomi Technology, we will introduce industry professionals as instructors and establish corporate partnership projects, thus building a communication bridge between students and the industry.

5 CONCLUSION

Software security experimental teaching plays a crucial role in helping students grasp the fundamental concepts and principles of software security. This paper starts with the existing issues in software security experimental teaching, providing a detailed introduction to the teaching objectives and content design of the course, while also discussing the methods of implementation. After more than five years of practice, we have achieved significant results, notably an improvement in students' ability to engage in autonomous learning.

ACKNOWLEDGMENT

This work is supported by the Natural Science Foundation of Hubei Province Educational Committee (ID: B2019009), and Innovative Talent Cultivation Special Project of Wuhan University of Science and Technology (2023X067&2023X027)".

REFERENCES

1. PENG GJ, ZHANG HY, DAI JM, et al. Research on the practice of attack and defense of software security quality course[J], computer education, 2020, (08): 79-83. DOI:10.16512/j.cnki.jsjjy.2020.08.019.
2. LIU YS, HONG S, LIU JW, et al. Teaching Reform and Practice of combining "Software security" with engineering problems [J]. Industrial and Information Education, 2023, (12):49-53.
3. PENG BT, WANG CJ, LUO HJ. Research on software vulnerability analysis experiment teaching[J]. Journal of Hubei Open Vocational College,2021,34(20):160-162.
4. J. O. B. Smith, A. Johnson, and R. Brown, "Teaching Software Security Through Attack-Defense Scenarios: An Educational Framework," Journal of Information Security Education, vol. 15, no. 3, pp. 200–215, 2022.
5. SU T, JIANG L, FANG M. Network and information security course series of integrated teaching exploration and practice[J]. Computer education, 2023, (06): 76-80. DOI:10.16512/j.cnki.jsjjy.2023.06.017.
6. S. R. Patel, "Developing Security Awareness in Software Engineering Students: A Dual Approach," Journal of Systems and Software, vol. 165, p. 110553, 2020.