Національний технічний університет України
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ
імені ІГОРЯ СІКОРСЬКОГО»

Кафедра програмного
забезпечення комп'ютерних
систем

SOFTWARE SECURITY

Working program of the academic discipline (Syllabus)

Details of the educational component

Level of higher education	***First (Bachelor)***
Branch of knowledge	*12 Information Technologies*
Specialty	*121 Software Engineering*
Educational program	*Software Engineering of Multimedia and Information-Retrieval Systems*
Status of the educational component	*Normative*
Form of education	*Full-time*
A year of training	*4th year of training, 7th semester*
The scope of the educational component	*Lectures: 36 hours, laboratory work: 18 hours, self-study: 66 hours.*
Semester control / control measures	*Exam, modular control work, calendar control*
Schedule of classes	*According to the schedule for the autumn semester of the current academic year (http://roz.kpi.ua/*
Language of instructions	*English*
Information about head of the course / teachers	*Lecturer: Ph.D., associate professor, V. V. Tsurkan, v.v.tsurkan@gmail.com* *Laboratory work: Ph.D., associate professor, V.V. Tsurkan, v.v.tsurkan@gmail.com*
Course location	Google classroom: https://classroom.google.com/

Program of academic discipline

1. Description of the educational discipline, its purpose, subject of study and learning outcomes

The study of the discipline "Software Security" allows students to develop the competencies necessary for solving practical tasks of professional activity related to the development of software in terms of its security (primarily confidentiality, integrity, availability).

The purpose of studying the discipline "Software Security" is the formation of students' abilities to independently develop software by defining and implementing its security requirements (primarily confidentiality, integrity, availability).

The subject of the "Software Security" discipline is methods of modeling software security threats.

The study of the discipline "Software Security" forms professional competences (FC) in students, necessary for solving practical tasks of professional activities related to the development, improvement and operation of software:
PC01 *Ability to identify, classify and formulate software requirements.*
PC03 *Ability to develop software systems architectures, modules and components.*
PC06 *Ability to analyze, select and apply methods and tools to ensure information security (including cybersecurity).*

PC08 *Ability to apply fundamental and interdisciplinary knowledge to successfully solve software engineering problems.*
PC14 *Ability to algorithmic and logical thinking.*
PC17 *Ability to develop software for information retrieval systems.*
PC19 *Ability to develop software for multimedia and mulsemedia systems.*

The study of the discipline "Software Security" contributes to the formation in students of the following ***program learning outcomes (PLO)*** *according to the educational program:*
PLO01 *To analyze, purposefully search and select the necessary information and reference resources and knowledge to solve professional problems, taking into account modern advances in science and technology.*
PLO18 *To know and be able to apply information technology of processing, storage and transmission of data.*
PLOH21 *To know the tools, analyze, select, skillfully apply the information security (including cybersecurity) and data integrity means in accordance with the applied tasks and software systems.*
PLO38 *To be able to apply programming technologies for multimedia and information retrieval systems software development.*

2. Pre-requisites and post-requisites of the discipline (place in the structural and logical scheme of training according to the relevant educational program)

The successful study of the discipline "Software Security" is preceded by the study of the disciplines "Components of Software Engineering", "Fundamentals of Programming", "Databases", "Programming" of the training plan for bachelors in the specialty 121 Software Engineering.

The theoretical knowledge and practical skills obtained during the mastering of the "Software Security" discipline ensure the successful implementation of course projects and diploma projects in the specialty 121 Software Engineering.

3. Content of the academic discipline

The discipline "Software Security" involves the study of the following topics:
Topic 1. Introduction to software security
Topic 2. Methods of modeling software security threats
Modular control work
Exam

4. Educational materials and resources

Basic literature:

1. Security of the software environment. Electronic campus of NTUU "KPI named after Igor Sikorsky". Materials from the discipline "Security of the software environment". – Access to registered students.

Additional literature:

2. Shostak A. Threat Modeling: Designing for Security. Indianapolis: Jon Shii & Sons, 2014. 590 p.

3. Tarandach I., Coles M. J. Threat Modeling. A Practical Guide for Development Teams. Sebastopol: O'Reilly Media, 2020, 201 p.

4. Threat Modeling. URL: https://owasp.org/www-community/Threat_Modeling (accessed on: 01.06.2022).

5. Common Vulnerability Scoring System v3.1: Specification Document. URL: https://www.first.org/cvss/v3.1/specification-document (accessed on: 01.06.2022).

6. LINDDUN framework. URL: https://www.linddun.org/linddun (accessed on: 01.06.2022).

7. ISO/IEC 27005:2018. Information technology. Security techniques. Information security risk management. [Valid from 2018-06-10]. URL: https://www.iso.org/standard/75281.html (accessed on: 01.06.2022).

8. MITER ATT&CK. URL: https://attack.mitre.org/ (accessed on: 01.06.2022).

9. Threat Modeling Manifesto. URL: https://www.threatmodelingmanifesto.org/ (accessed on: 01.06.2022).

10. Threat Modeling. URL: https://www.microsoft.com/en-us/securityengineering/sdl/threatmodeling (accessed on: 01.06.2022).

11. Threat Modeling. Process. URL: https://owasp.org/www-community/Threat_ Modeling_Process (accessed on: 01.06.2022).

12. Create a threat model using data-flow diagram elements. URL: https://docs.microsoft.com/en-us/learn/modules/tm-create-a-threat-model-using-foundational-data-flow-diagram-elements/ (accessed on: 01.06.2022).

13. ISO/IEC 27000:2018. Information technology. Security techniques. Information security management systems. Overview and vocabulary. [Valid from 2018-02-07]. URL: https://www.iso.org/standard/73906.html (accessed on: 01.06.2022).

14. DREAD Threat Modeling: An Introduction to Qualitative Risk Analysis. URL: https://www.eccouncil.org/cybersecurity-exchange/threat-intelligence/dread-threat-modeling-intro/ (accessed on: 01.06.2022).

15. Schneier B. Attack Trees. URL: https://www.schneier.com/academic/archives/1999/12/attack_trees.html (accessed on: 01.06.2022).

16. An Alternative: Attack Trees. URL: https://www.oreilly.com/library/view/building-secure-servers/0596002173/ch01s03.html (accessed on: 01.06.2022)

17. Common Vulnerability Scoring System v3.1: Examples. URL: https://www.first.org/cvss/v3.1/examples (accessed on: 01.06.2022).

18. IEC 31010:2019. Risk management. Risk assessment techniques. [Valid from 2019-06-17]. URL: https://www.iso.org/standard/72140.html (accessed on: 01.06.2022).

19. Finding Cyber Threats with ATT&CK™-Based Analytics. URL: https://www.mitre.org/sites/default/files/2021-11/16-3713-finding-cyber-threats-with-attack-based-analytics.pdf (accessed on: 01.06.2022).

Use to master the practical skills of the discipline. The materials are freely available on the Internet.

Educational content

4. Methods of mastering an educational discipline (educational component)

No	*Type of training session*	*Description of the training session*
Topic 1. Introduction to software security		
1	*Lecture 1. Course content, introduction to software security*	*Overview of course content. The concept of software security. Properties of confidentiality, integrity, availability. Life cycle of developing secure software. Approaches to defining software security requirements* *Task on self-study: item 6 No. 1.*

2	*Lecture 2. Modeling software security threats*	*The concept of software security threat modeling. The process of modeling software security threats. Stages of modeling software security threats. Decomposition of software in terms of threats.* *Assignment on self-study: item 6 No. 2.*
3	*Laboratory work 1. Software decomposition in terms of threats*	*Task: to decompose the software in terms of threats.* *Assignment on self-study: item 6 No. 3.*
Topic 2. Methods of modeling software security threats		
4	*Lecture 3. A method of modeling software security threats based on a data flow diagram*	*The concept and characteristics of the data flow diagram. Features of using a flow diagram to model software security threats. Elements of a data flow diagram. The process of building a data flow diagram.* *Tasks on self-study: Item 6 No. 4.*
5	*Lecture 4. Stages of building a data flow diagram*	*Rules for constructing a data flow diagram. Definition of software processes. Definition of software data stores. Definition of software entities. Defining data flows and trust boundaries between software elements.* *Task on self-study: item 6 No. 5.*
6	*Laboratory work 2. Creating a model of software security threats based on a data flow diagram*	*Task: Create a software security threat model based on a data flow diagram.* *Assignment on self-study: item 6 No. 6.*
7	*Lecture 5. STRIDE software security threat modeling method*	*Characteristics of the STRIDE method. Attributes of the STRIDE method: cpufing, spoofing, denial, disclosure, denial of service, privilege escalation. The process of modeling software security threats using the STRIDE method.* *Tasks on self-study: Item 6 No. 7.*
8	*Lecture 6. Stages of modeling software security threats using the STRIDE method*	*Defining the threat of spoofing. Definition of the threat of falsification. Determining the threat of failure. Determination of the threat of information disclosure. Determining the threat of denial of service. Identifying the threat of privilege escalation. Defining software security requirements.* *Tasks on self-study: Item 6 No. 8.*
9	*Laboratory work 3. Creating a model of software security threats using the STRIDE method*	*Task: create a model of software security threats using the STRIDE method.* *Assignment on self-study: item 6 No. 9.*
10	*Lecture 7. DREAD software security threat modeling method*	*Characteristics of the DREAD method. DREAD threat assessment scales. Selection of the threat assessment scale by the DREAD method. The process of modeling software security threats using the DREAD method.* *Assignment on self-study: item 6 No. 10.*
11	*Lecture 8. Stages of modeling software*	*Identifying software security threats. Definition of software security threat assessment scale. Software security threat assessment.*

	security threats using the DREAD method	*Software Security Threat Ranking. Defining software security requirements.* *Task on self-study: item 6 No. 11.*
12	*Laboratory work 4. Creating a software security threat model using the DREAD method*	*Task: create a model of software security threats using the DREAD method.* *Assignment on self-study: item 6 #12.*
13	*Lecture 9. Modeling software security threats by creating an attack tree*	*Characteristics of the attack tree creation method. Ways to use the attack tree. Choosing how to use the attack tree. The process of modeling software security threats using the method of creating an attack tree.* *Assignment on self-study: item 6 No. 13.*
14	*Lecture 10. Stages of modeling software security threats using the method of creating an attack tree*	*Choosing how to display the attack tree. Creating the root node of the attack tree. Creating subnodes of the root node of the attack tree. Checking the completeness of the created attack tree. Pruning the generated attack tree. Checking the generated attack tree.* *Assignment on self-study: item 6 No. 14.*
15	*Laboratory work 5. Creating a model of software security threats using the attack tree method*	*Task: create a model of software security threats using the attack tree method.* *Task on self-study: item 6 No. 15.*
16	*Lecture 11. The method of assessing the severity of software vulnerabilities*	*Characteristics of the method of evaluating the severity of vulnerabilities. Metrics for evaluating the severity of vulnerabilities. A string vector for assessing the severity of vulnerabilities. Vulnerability Severity Calculator.* *Assignment on self-study: item 6 No. 16.*
17	*Lecture 12. Stages of the method of assessing the severity of software vulnerabilities забезпечення*	*Definition of basic metrics. Determination of time metrics. Defining user environment metrics. Definition of the equation for evaluating the severity of software vulnerabilities. Using the Software Vulnerability Severity Calculator.* *Assignment on self-study: item 6 No. 17.*
18	*Laboratory work 6. Evaluation of software vulnerabilities according to the CVSS standard*	*Task: evaluate software vulnerabilities according to the CVSS standard.* *Task on self-study: item 6 No. 18.*
19	*Lecture 13. Modeling software security threats using the LINDDUN method*	*Characteristics of the LINDDUN method. Categories of threats according to the LINDDUN method. Building a software security threat model. Detection of software privacy threats. Managing Software Privacy Threats.* *Assignment on self-study: item 6 No. 19.*
20	*Lecture 14. Detection of software privacy threats using the LINDDUN method*	*Building a software model. Mapping the elements of the data flow diagram in relation to categories of privacy threats. Identifying and documenting privacy threats.* *Task on self-study: item 6 No. 20.*

21	Modular control work. Implementation of software security requirements	Task: implement software security requirements. Task on self-study: item 6 No. 21.
22	Lecture 15. Methods of software security risk assessment	Varieties of software security risk assessment methods. Criteria for choosing software security risk assessment methods. Approaches to choosing software security risk assessment methods. Task on self-study: item 6 No. 22.
23	Lecture 16. Assessment of information security risks using the "Consequences - Probability Matrix" method	Characteristics of the risk assessment method. Risk assessment scales. Choosing a risk assessment scale. Risk acceptability criteria. Stages of using the risk assessment method. Task on self-study: item 6 No. 23.
24	Laboratory work 7. Demonstration of implemented software security requirements	Task: demonstrate implemented software security requirements. Task on self-study: item 6 No. 24.
25	Lecture 17. MITER ATT&CK Knowledge Base on Software Hacker Tactics and Techniques	Structure of the MITER ATT&CK knowledge base. MITER ATT&CK matrix. Categories of structuring knowledge about tactics and techniques of the violator: enterprise, mobile devices, industrial control systems. Task on self-study: item 6 No. 25.
26	Lecture 18. Creating a software security threat model based on the MITER ATT&CK knowledge base	Determination of the offender's behavior. Establishing data to determine the behavior of the offender. Defining analytics based on established data to determine offender behavior. Determination of probable scenarios of the violator's actions. Assessment of likely actions of the violator. Task on self-study: item 6 No. 26.

5. Independent work of a student/graduate student

The "Software Security" discipline is based on independent preparations for classroom classes on theoretical and practical topics.

№ z/ p	The name of the topic submitted for independent processing	Number of hours	literature
1	Preparation for the lecture 1	1	1; 2; 3; 4; 9–11
2	Preparation for the lecture 2	1	1; 2; 3; 4; 9–11
3	Preparation for laboratory work 1	2	1; 2; 3; 4; 9–11
4	Preparation for the lecture 3	1	1; 2; 3; 4; 9–11
5	Preparation for the lecture 4	1	1; 2; 3; 4; 9–11
6	Preparation for laboratory work 2	2	1; 2; 3; 4; 9–11
7	Preparation for the lecture 5	1	1; 2; 3; 4; 9–11; 13

8	*Preparation for the lecture 6*	*1*	*1; 2; 3; 4; 9–11; 13*
9	*Preparation for laboratory work 3*	*2*	*1; 2; 3; 4; 9–11; 13*
10	*Preparation for the lecture 7*	*1*	*1; 2; 3; 4; 9–11; 14*
11	*Preparation for the lecture 8*	*1*	*1; 2; 3; 4; 9–11; 14*
12	*Preparation for laboratory work 4*	*2*	*1; 2; 3; 4; 9–11; 14*
13	*Preparation for the lecture 9*	*1*	*1; 2; 3; 4; 9–11; 15; 16*
14	*Preparation for the lecture 10*	*1*	*1; 2; 3; 4; 9–11; 15; 16*
15	*Preparation for laboratory work 5*	*2*	*1; 2; 3; 4; 9–11; 15; 16*
16	*Preparation for the lecture 11*	*1*	*1; 2; 3; 4; 5; 9–11; 17*
17	*Preparation for the lecture 12*	*1*	*1; 2; 3; 4; 5; 9–11; 17*
18	*Preparation for laboratory work 6*	*2*	*1; 2; 3; 4; 5; 9–11; 17*
19	*Preparation for the lecture 13*	*1*	*1; 2; 3; 4; 6; 9–11*
20	*Preparation for the lecture 14*	*1*	*1; 2; 3; 4; 6; 9–11*
21	*Preparation for modular control work*	*4*	*1; 2; 3; 4; 9–16*
22	*Preparation for the lecture 15*	*1*	*1; 2; 3; 4; 7; 9–11; 18*
23	*Preparation for the lecture 16*	*1*	*1; 2; 3; 4; 7; 9–11; 18*
24	*Preparation for laboratory work 7*	*2*	*1; 2; 3; 4; 9–16*
25	*Preparation for the lecture 17*	*1*	*1; 2; 3; 4; 8–11; 19*
26	*Preparation for the lecture 18*	*1*	*1; 2; 3; 4; 8–11; 19*
27	*Preparation for the exam*	*30*	*1-19*

Preparation for laboratory work

6. Policy of academic discipline (educational component)

Attending lectures is mandatory.

- *Attending laboratory work classes may be occasional and as needed for consultation/protection of laboratory work.*
- *Rules of behavior in classes: activity, respect for those present, turning off phones.*
- *Adherence to the policy of academic integrity.*
- *Rules for the protection of laboratory work: the work must be performed in accordance with the assigned tasks and according to the option chosen by the student.*

7. Types of control and rating system for evaluating learning outcomes (RSO)

During the semester, students perform 7 laboratory works.

The maximum number of points for each laboratory work: *5 points.*
Points are awarded for the quality of performance and protection of laboratory work: 0-5 points.
Criteria for evaluating the quality of performance and protection:
5 points - the work is done qualitatively, in full, the answers are complete, well-argued;
4 points - the work is done qualitatively, in full, but has shortcomings, answers with minor errors;

3 points – the work is done with sufficient quality, in full, but contains significant shortcomings, answers with significant errors;
0 points - the work is not done well, not in full, the answers are either absent or incorrect.
The maximum number of points for performing and defending laboratory work:
5 points × 7 laboratory works = 35 points.
The task of modular control work is to implement software security requirements. The answer is evaluated by 15 points.
Evaluation criteria for modular test work:
14–15 points – the answer is correct, complete, well-argued;
12–13 points – the answer is generally correct, but has flaws;
9–11 points – there are significant errors in the answer;
0 points - there is no answer or the answer is incorrect.
The maximum number of points for a modular control work:

15 points × 1 task = 15 points.

The rating scale for the discipline is equal to:

$R = RS = R_{lab.\ works} + R_{modular\ control\ work} + R_{exam}$ = 35 points + 15 points + 50 points = 100 points.

Calendar control: is conducted twice a semester as a monitoring of the current state of fulfillment of the syllabus requirements.

At the first certification (8th week), the student receives "Passed" if his current rating is at least 10 points (50% of the maximum number of points that the student can receive before the first certification).

At the second certification (14th week), the student receives "Passed" if his current rating is at least 20 points (50% of the maximum number of points that the student can receive before the second certification).

Semester control: exam

Conditions for admission to semester control:

A prerequisite for a student's admission to the exam is a semester rating (RC) of at least 30 points. After passing the exam, a grade is assigned according to the table (Table of correspondence of rating points to grades on the university scale).

The exam task consists of 3 questions - 2 theoretical and 1 practical. The answer to each theory question is worth 15 points, and the answer to a practical question is worth 20 points.

Evaluation criteria for a theoretical question:

14–15 points – the answer is correct, complete, well-argued;

11–13 points – the answer is generally correct, but has flaws;

5–10 points – there are significant errors in the answer;

0 points - there is no answer or the answer is incorrect.

Evaluation criteria for a practical question:

17–20 points – the answer is correct, complete, well-argued;

12–16 points – the answer is generally correct, but has flaws;

5–11 points – there are significant errors in the answer;

0 points - there is no answer or the answer is incorrect.

Table of correspondence of rating points to grades on the university scale:

Scores	*Rating*
100-95	Perfectly

94-85	Very good
84-75	Fine
74-65	Satisfactorily
64-60	Enough
Less 60	Unsatisfactorily
Admission conditions not met	Not allowed

8. Additional information on the discipline (educational component)

The list of questions submitted for semester control is given in Appendix 1.

Working program of the academic discipline (syllabus):

Compiled by Ph.D., Associate Professor V.V. Tsurkan.

Adopted by Computer Systems Software Department (protocol № 12 from 26.04.23)

Approved by the Faculty Board of Methodology (protocol № 10 from 26.05.23)

Appendix 1. List of questions submitted for semester control

1. Describe the concept of software security.

2. To characterize the properties of confidentiality, integrity, and availability of software data.

3. Describe the life cycle of developing secure software.

4. Describe approaches to defining software security requirements.

5. Describe the concept of software security threat modeling.

6. Describe the process of modeling software security threats.

7. Describe the stages of software security threat modeling.

8. To characterize software decomposition in terms of threats.

9. Describe the method of modeling software security threats based on the data flow diagram.

10. Describe the STRIDE software security threat modeling method.

11. Describe the DREAD software security threat modeling method.

12. To characterize the modeling of software security threats by the method of creating an attack tree.

13. Describe the method of assessing the severity of software vulnerabilities.

14. Describe the severity metrics of software vulnerabilities.

15. Describe the simulation of software security threats using the LINDDUN method.

16. Describe approaches to choosing software security risk assessment methods.

17. To characterize the assessment of information security risks using the "Consequences - Probability Matrix" method.

18. Describe the creation of a software security threat model based on the MITER ATT&CK knowledge base.

19. Describe the definition of behavior analytics of a software security violator based on the MITER ATT&CK knowledge base.

20. To characterize the determination of probable scenarios of actions of a software security violator based on the MITER ATT&CK knowledge base.