COURSE DESCRIPTION CARD - SYLLABUS

Course name

Virtualization methods [S1Cybez1>MW]

Course

Field of study Cybersecurity	Year/Semester 2/3
Area of study (specialization) –	Profile of study general academic
Level of study first-cycle	Course offered in Polish
Form of study full-time	Requirements compulsory

Number of hours

Lecture 8	Laboratory classes 24	Other 0
Tutorials 0	Projects/seminars 24	

Number of credit points

4,00

Coordinators	Lecturers
dr inż. Łukasz Kułacz lukasz.kulacz@put.poznan.pl	

Prerequisites

The student should have a basic knowledge of programming, especially the concepts of variables, classes and functions. He should have the ability to implement simple programs and web applications.

Course objective

The purpose of the course is to provide students with basic knowledge of vitualization and containerization. Through virtualization, the student can isolate the designed application or service from the host operating system, so that it is possible to ensure that the code runs independently of hardware, system versions and drivers. In addition, the course will present issues related to containerization, primarily, in the context of scalability and independence of applications designed as so-called microservices.

Course-related learning outcomes

Knowledge:

The student has basic theoretical and practical knowledge of software virtualization, containerization and container orchestration. The student knows the basic tools to create, use and manage containers. The student knows the features and limitations of using containers to deploy a finished solution. [K1_W06][K1_W11]

Skills:
The student is able to create, use and manage virtual operating systems. The student is able to prepare simple applications and then place them in containers, run and update them. The student is able to combine several microservices into a working system and swap system components without starting the whole system from scratch. [K1_U02]

Social competences:
The student is aware of the opportunities and limitations associated with developing software and placing it inside containers. Understands the potential gains associated with scalability and increased reliability of the system while recognizing the risks associated with using such a solution. [K1_K05]

Methods for verifying learning outcomes and assessment criteria

Learning outcomes presented above are verified as follows:

In terms of the project, the verification of the established learning outcomes is carried out by: substantive evaluation of the realization of the individual project.
In terms of the laboratory, verification of the established learning outcomes is realized by: substantive evaluation of individual or group tasks performed in class or in the form of tasks to be performed after class, activity in class.
In terms of the lecture, verification of the established learning outcomes is carried out through a written test, which can include single-choice, multiple-choice and open-ended questions.
In each form of the course assessment, the grade depends on the number of points the student earns relative to the maximum number of required points. Earning at least 50% of the possible points is a prerequisite for passing. The relationship between the grade and the number of points is defined by the Study Regulations. Additionally, the course completion rules and the exact passing thresholds will be communicated to students at the beginning of the semester through the university's electronic systems and during the first class meeting (in each form of classes).

Programme content

1. Vitualization of operating systems.
2. Basics of containerization.
3. Basics of microservices.
4. Basics of container orchestration.
5. Integration of microservices.

Course topics

1. Vitualization of operating systems.
2. Basics of containerization.
3. Basics of microservices.
4. Basics of container orchestration.
5. Integration of microservices.

Teaching methods

Lecture: multimedia presentation, supplemented by examples and additional explanations based on code snippets.
Laboratories: work at the computer, performance of individual tasks, performance of group tasks.
Project: work at the computer, performance of individual tasks.

Bibliography

Basic:
1. Rozwijanie mikrousług w Pythonie. Budowa, testowanie, instalacja i skalowanie. Tarek Ziade
2. Architektura aplikacji w Pythonie. TDD, DDD i rozwój mikrousług reaktywnych. Harry Percival
3. Nauka Dockera w miesiąc. Elton Stoneman
4. Nauka Kubernetesa w miesiąc. Elton Stoneman

Additional:

1. Docker. Niezawodne kontenery produkcyjne. Praktyczne zastosowania. Sean Kane

Breakdown of average student's workload

	Hours	ECTS
Total workload	116	4,00
Classes requiring direct contact with the teacher	56	2,00
Student's own work (literature studies, preparation for laboratory classes/ tutorials, preparation for tests/exam, project preparation)	60	2,00