COURSE DESCRIPTION CARD - SYLLABUS

Course name
Malicious Software Analysis [S2Inf1E-CYB>MSA]

Course

Field of study
Computing

Year/Semester
2/3

Area of study (specialization)
Cybersecurity

Profile of study
general academic

Level of study
second-cycle

Course offered in
English

Form of study
full-time

Requirements
compulsory

Number of hours

Lecture
15

Laboratory classes
30

Other
0

Tutorials
0

Projects/seminars
0

Number of credit points

4,00

Coordinators

dr hab. inż. Piotr Zwierzykowski prof. PP
piotr.zwierzykowski@put.poznan.pl

mgr inż. Błażej Nowak
blazej.nowak@put.poznan.pl

Lecturers

Prerequisites

A student entering this subject should have a basic knowledge of computer networks, cryptographic algorithms and Windows and Linux operating systems. He or she should also have the ability to obtain information from the indicated sources and have a willingness to to cooperate as part of a team.

Course objective

Provide students with knowledge in the field of widely understood malware analysis, including methods and tools used for static and dynamic analysis of such software and elements of reverse engineering. As part of the course, the selected methods of static and dynamic malware analysis and reverse engineering used for this purpose will be discussed. As part of the laboratory exercises, the student will get to know the tools to detect malware in practice

Course-related learning outcomes

Knowledge:

has a structured and theoretically founded general knowledge related to key issues in the field of malware analysis.
has advanced detailed knowledge of selected issues in the field of broadly understood malware analysis as well as methods and tools used for static and dynamic analysis and reverse engineering.
has knowledge about development trends and the most important cutting edge achievements in computer science and in the field of malware detection, static and dynamic analysis.
has advanced and detailed knowledge of the processes occurring in systems used for dynamic malware analysis

Skills:
is able to obtain information on methods of malicious software analysis from literature, databases and other sources (both in polish and english), integrate them, interpret and critically evaluate them, draw conclusions and formulate and fully justify opinions
is able to plan and carry out experiments, including computer measurements and simulations, interpret the obtained results and draw conclusions and formulate and verify hypotheses related to malware analysis.
can integrate knowledge from different areas of computer science (and if necessary also knowledge from other scientific disciplines) when formulating and solving engineering tasks related to the detection and analysis of malware.
is able to assess the suitability and the possibility of using new hardware and software solutions for solving engineering tasks consisting in building secure data transmission systems.

Social competences:
understands that in the field of ict security, knowledge and skills quickly become obsolete.
understands the importance of using the latest knowledge in the field of ict security in solving research and practical problems.
is aware of the need to develop professional achievements and comply with the rules of professional ethics

Methods for verifying learning outcomes and assessment criteria

Learning outcomes presented above are verified as follows:

Learning outcomes presented above are verified as follows:
The knowledge acquired during the lecture is verified by an oral and / or written test.
Passing issues, on the basis of which questions are developed, are sent to students by e-mail using the university"s e-mail system, or placed in a subject course in the university"s distance learning system.
Oral and / or written test consists of 3 to 5 questions for which a descriptive answer is expected. Each answer to the question is rated on a scale of 0 to 5 points. Each question is scored equally. Passing threshold: 50% of points.
In the case of an oral test, students draw questions from a set of 30 questions. In the case of a written test, questions are asked by the teacher.
The skills acquired during the laboratory classes are verified on an ongoing basis. At each laboratory class, the correctness of the exercises is assessed on a scale from 2 to 5. The final grade is the average of the grades obtained from each laboratory session. The final grade is the average of the grades obtained from each laboratory session.

Programme content

Lecture topics:
- Introduction to Malware Analysis
- Classification of Malware
- Static Analysis of Malware
- Dynamic Analysis of Malware
- Reverse Engineering in Malware Analysis
- Static and Dynamic Reversing
- Malware Functionalities and Persistence
- Malware Obfuscation and Evasion Techniques

Laboratory topics:
- Basic Static Analysis Techniques,

- Malware Analysis in Virtual Machines,
- Basic Dynamic Analysis Techniques,
- IDA - Interactive Disassembler,
- C Constructions in ASM,
- Malware Analysis in Windows OS,
- OllyDbg - Interactive Debugger,
- Network Signatures,
- Encryption,
- C++ Analysis.

Course topics

Introduction to malware analysis
Malware analysis is the study and understanding of the modus operandi, objectives, and effects of malicious code. The goal is to identify, classify, and develop methods to detect and neutralize threats. Analysis can be conducted at various levels, from superficial identification of threats to detailed decompilation and examination of code.

Classification of malware
Malware can be classified according to various criteria, such as the method of infection, the type of activity, or the target of the attack.

Reverse engineering in malware analysis
Reverse engineering involves decompiling and analyzing malicious code to understand how it works.

Functions and persistence of malware
Malware often contains a variety of functionalities designed to permanently install itself on a system and perform malicious activities.

Obfuscation and detection avoidance techniques
Malware often employs various obfuscation and evasion techniques to make analysis and detection by security systems more difficult.

Basic static analysis techniques
Static analysis involves examining program code without running it. To do this, we analyze binary files, source code (if available), and other resources such as libraries or configuration files. Tools often used in static analysis are the disassembler, binary analysis tools, and source code analysis tools.

Malware analysis in virtual machines
Using virtual machines (VMs) allows malware to be safely run and analyzed without risking infection of the host system. Popular solutions include VMware, VirtualBox and Hyper-V. Analysis in a VM allows you to create an isolated environment where you can monitor the behavior of suspicious software, track changes in the file system, system registry and network traffic.

Basic dynamic analysis techniques
Dynamic analysis involves studying the behavior of a program while it is running. This includes running the program in a controlled environment and monitoring its activity, such as network communications, file operations, and interactions with the operating system. Tools used in dynamic analysis include Process Monitor, Process Explorer, Wireshark, and file system and registry tracking tools.

IDA - Interactive Disassembler
IDA (Interactive Disassembler) is one of the most powerful tools for static analysis of software. It allows you to disassemble binary code into a readable assembler form, making it possible to analyze program structure and logic. IDA offers an interactive environment with the ability to add comments, labels and control flow analysis.

C language constructs in ASM
Understanding how C language constructs (such as loops, conditional statements, functions) are converted into assembler code by the compiler is crucial in analyzing low-level code. This allows for easier tracking of program logic and identification of key code fragments that may be responsible for malicious activity.

Windows malware analysis
Windows is the most commonly attacked operating system, so understanding its architecture and security mechanisms is crucial. Windows malware analysis includes examining registry changes, tracking system processes and services, analyzing executable files and monitoring interactions with the operating system.

OllyDbg - Interactive debugger
OllyDbg is a popular dynamic analysis tool that allows interactive debugging of applications. With OllyDbg, you can perform step-by-step program instructions, monitor variables, track control flow and identify errors and malicious behavior in code. It is particularly useful for analyzing programs without access to the source code.

Network signatures
Network signatures are patterns used to identify malicious network traffic. They can be specific byte sequences, IP addresses, domains, or specific communication patterns used by malware. Network traffic analysis tools, such as Wireshark or IDS (Intrusion Detection System), use these signatures to detect and block suspicious traffic.

Encryption
Encryption is often used by malware to hide its operations or to secure communications with control servers. Understanding encryption techniques, such as AES, RSA, and the ability to decode them, is crucial in malware analysis. Often, analysis of encryption keys or encryption methods can lead to the discovery of information about the malware's operation.

C++ analysis
The analysis of programs written in C++ is more complicated than those written in C, due to the complexity of the language and the use of object-orientation. This includes understanding constructs such as classes, inheritance, polymorphism, and memory management. Disassembling C++ code and analyzing its structure requires advanced tools and techniques, such as analyzing virtual arrays, identifying virtual methods, and reconstructing the structure of classes.

Teaching methods

Lecture: multimedia presentation, illustrated by examples given on the blackboard.
Laboratory: practical exercises in groups or individually, using test environments and tools.

Bibliography

Basic
D. Barker: Malware Analysis Techniques, Packt>, 2021
Additional
1. Alexey Kleymenov, Amr Thabet: Mastering Malware Analysis,Packt>, 2019
2. Reginald Wong: Mastering Reverse Engineering, Packet>, 2018
3. K.A. Monnappa: Learning Malware Analysis, Pack>, 2018
4. M. Skikorski, A. Honing: Practical Malware Analysis: The Hands-On Guide to Dissecting Malicious Software, No Starch Press; 1st edition , 2012
5.O. Or-Meir, Nir Nissim, Yuval Elovici, and Lior Rokach: Dynamic Malware Analysis in the Modern Era—A State of the Art Survey, ACM Computing Surveys, Vol. 52 Issue 5, October 2019, Article No.: 88, pp 1–48, 10.1145.3329786

Breakdown of average student's workload

	Hours	ECTS
Total workload	100	4,00
Classes requiring direct contact with the teacher	45	2,00
Student's own work (literature studies, preparation for laboratory classes/ tutorials, preparation for tests/exam, project preparation)	55	2,00