GHENT UNIVERSITY

Course Specifications

Valid as from the academic year 2023-2024

Software Hacking and Protection (E017942)

Course size *(nominal values; actual values may depend on programme)*

Credits 6.0 **Study time 180 h**

Course offerings in academic year 2023-2024

A (semester 1)	English	Gent

Lecturers in academic year 2023-2024

Coppens, Bart	TWO6	staff member
De Sutter, Bjorn	TWO6	lecturer-in-charge

Offered in the following programmes in 2023-2024	crdts	offering
Master of Science in Computer Science	6	A
Master of Science in Computer Science Engineering	6	A
Master of Science in Computer Science Engineering	6	A
Master of Science in Information Engineering Technology	6	A
Exchange Programme in Computer Science (master's level)	6	A
Exchange Programme Information Engineering Technology	6	A

Teaching languages

English

Keywords

Software asset risk management, reverse engineering, software tampering, man-at-the-end attacks, software protection, protection evaluation and deployment, physical attacks on software, malware analysis, hardware support for confidentiality and integrity of software

Position of the course

Since the start of the ICT-revolution, companies, organizations, private citizens, governments, and society at large have come to depend more and more on software. Today, software benefits many aspects of our private and public lives. At the same time, it is a key enabler of many business models. Software can also be abused and weaponized in various ways, however, to disrupt private lives, business models, and society at large. There hence exists a strong need to defend (systems running) software against a range of attacks. Doing so requires defensive software protection knowledge and skills, but also offensive training. After all, poachers make the best gamekeepers. Moreover, at the technical level, the required skills largely overlap: many technically neutral methods and tools can be used offensively as well as defensively, the difference solely depending on whether the targeted software is malware or goodware.

In this course, we focus on the technical knowledge and skills, so-called hacking skills, required to attack and protect software assets. These assets are the most valuable and sensitive parts of programs that need to remain confidential in the face of reverse engineering and of which the integrity needs to be maintained in the face of software tampering attempts. We study and practice a wide range of software analysis and tampering techniques, including side channel and fault-injection techniques that rely on physical implementation features, as well as protections such as obfuscations that aim to mitigate unauthorized uses of those techniques to protect assets and their security requirements. We study and practice the complex tasks of deploying good combinations of protections and of evaluating the achieved levels of protection, including the design and execution of empirical experiments in the domain of reverse engineering and software protection. Furthermore, we study and practice how to use a range of software analyses and forensic techniques to analyze and detect malware, as well the techniques that malware authors deploy in an attempt to evade detection.

The attacks and mitigations in the scope of this course are commonly called man-at-the-end

techniques, as they are deployed by parties that have (almost) complete control over the end devices on which they attack and analyze the software, such that they don't need to rely on exploitable vulnerabilities for their attacks and analyses. This course hence complements the courses on Information Security, which focuses on cryptography to counter man-in-the-middle attacks; Secure Software and Systems, which focuses on (often remotely) exploitable flaws and bugs in software designs and implementations; and Network Security, which focuses on attacks with which network infrastructures are penetrated.
This course builds on the knowledge gained in earlier courses on computer architecture, operating systems, and programming in low-level languages (C, C++).

Contents

- Context: man-at-the-end attack model, software assets, security requirements, security economics
- Static reverse engineering tools and techniques: interactive disassemblers, decompilers, pattern matching, etc.
- Dynamic reverse engineering tools and techniques: debuggers, hooking, emulation, tracing, statistical analysis, symbolic and concolic execution, taint analysis, delta analyses, fuzzing, etc.
- Software tampering techniques, static and dynamic
- Software obfuscation techniques (layout, design, code, data)
- Preventive software protection techniques: anti-debugging, anti-tampering, anti-emulation
- Software protection evaluation and validation methodologies, incl. the design, execution, and analysis of empirical experiments with human subjects
- Software asset risk management approaches, decision support for software protection, impact on software development life cycle
- Physical attacks based on side channels and fault injection
- Malware analysis, detection, and classification techniques
- Domain-specific attacks and defense on assets embedded in software: cryptographic assets, machine learning models, and possibly others
- Hardware support for code and data confidentiality and integrity

Initial competences

- Programming in C.
- Basic knowledge of computer architecture, in particular the software-hardware interface including basic x86 and x86-64 assembler knowledge.
- Basic knowledge of operating systems.

Final competences

1 Understanding the man-at-the-end attack model, including its relevant constructs, models, and methods.
2 Knowing how to devise appropriate man-at-the-end attack strategies based on knowledge about the targeted assets.
3 Knowledge of, a deeper understanding of, and experience with the tools and techniques commonly deployed in man-at-the-end software attacks.
4 Capacity to execute a range of man-at-the-end attack steps on software (reverse engineering, software tampering) as part of such strategies.
5 Extended knowledge of the techniques used for software protection against man-at-the-end attacks, understanding their limitations and the need to combine and layer multiple techniques to obtain useful protection.
6 Basic experience in using software protection tools for deploying those techniques.
7 Understanding strategies for and the complexities of deploying, evaluating, and validating software protections, including applicable risk management approaches.
8 Knowing how to design, execute, and analyze empirical experiments involving human subjects for expanding the knowledge in software protection and man-at-the-end attack models.
9 Knowing the different types of malware and their core features.
10 Understanding the most used malware detection, classification, and analysis methods, and how malware tries to evade those.
11 Being able to deploy forensic malware analysis techniques and tools.
12 Communicating and presenting domain-specific knowledge in a correct and clear manner, with the appropriate language skills, incl. the use of correct terminology.

Conditions for credit contract

Access to this course unit via a credit contract is determined after successful competences assessment

Conditions for exam contract

This course unit cannot be taken via an exam contract

Teaching methods

Lecture, Practical, Independent work, Peer teaching

Extra information on the teaching methods

- Online knowledge clips, demo clips, quizzes and assignments (flipping the classroom)
- Lectures: some classical lectures, mostly Q&A lectures (response colleges)
- Practicums mostly on own laptops (which must be able to run x86 virtual machines), possibly also on provided experimentation boards.
- Peer presentations with guided preparation are done in small groups.

Learning materials and price

- Course website
- Annotated slides
- Online videos
- Online exercises (including quizzes) and tasks
- Handouts of labs, their assignments and preparatory material
- Academic publications (accessible/provided for free)
- Overview of content to be studied for the exam
- Example exam questions
- Course notes, content summaries, and slides (in English), provided for free through the electronic learning platform

References

- Surreptitious Software: Obfuscation, Watermarking, and Tamperproofing for Software Protection, by Christian Collberg & Jasvir Nagra, 2009, ISBN-13: 978-0321549259

Course content-related study coaching

- Interactive support and coaching through the electronic learning platform appointments
- Teachers and assistants are available for extra help before and after contact hours, and via chat and conf calls
- Use of flipping the classroom and blended learning techniques
- Intensive guidance for preparation of peer presentations

Assessment moments

end-of-term and continuous assessment

Examination methods in case of periodic assessment during the first examination period

Written assessment with open-ended questions

Examination methods in case of periodic assessment during the second examination period

Written assessment with open-ended questions

Examination methods in case of permanent assessment

Oral assessment, Participation, Peer and/or self assessment, Assignment

Possibilities of retake in case of permanent assessment

examination during the second examination period is possible in modified form

Extra information on the examination methods

- Periodic evaluation: written examination with open and closed questions on theory and practice (exercises), with closed-book
- During semester: graded lab sessions (written reports and possibly oral evaluation); participation 'flipping the classroom' activities, evaluation (including peer assessment) of peer presentations

Calculation of the examination mark

- 50% on permanent evaluation, 50% on periodic exam.
- Lack of participation in permanent evaluation for no valid reason results in a zero for that part.
- In the case of group assignments, the students in a group get the same score by default. Only when there is a clear difference in contribution, the students will be given different scores.
- The student must pass (>=10/20) both parts to pass the whole course. If they fail for one part while still scoring >=10/20 on average, the final score becomes 9/20.

Facilities for Working Students

Option to be freed from presence in labs with alternative assignment after consultation with the responsible teacher. Option for oral exam with written preparation at another time in the academic year. Option for feedback by appointment during and after business hours.