WM3E9-15 Malware Analysis and Exploit Development

27/28

Department
WMG
Level
Undergraduate Level 3
Module leader
Michael Macaulay
Credit value
15
Module duration
10 weeks
Assessment
100% coursework
Study location
University of Warwick main campus, Coventry

Description

Introductory description

Modern programming approaches use high level constructs which abstract away the system's architecture. These high levels of abstraction use code generation programs such as compilers and assemblers to take the human author's input, and produce code that will execute as output. The modern programmer rarely needs to consider the underlying architecture of the machine that will execute the code.
There are situations where, rather than creating an executable from source, you need to go in the opposite direction; you need to infer what the source code might look like by analysing the executable. Perhaps you have some potential malware; or perhaps you have to analyse and exploit a vulnerability which an executable might have. Either way, you want to know what the program will do, were it to run on your system.
In order to reverse engineer an executable, you need to understand the typical idioms that an operating system, architecture and code generation programs will adopt to convert high level constructs into low level executables.
In addition, if the executable is malware, then it is likely the authors will have used some obfuscation in order to make the analysis more difficult. Under these circumstances you need to understand the typical idioms of obfuscation.

Module aims

The module aims to explore the essential low-level techniques and analysis concepts relevant to identifying malicious code and exploiting vulnerabilities that reside in the binaries.

Outline syllabus

This is an indicative module outline only to give an indication of the sort of topics that may be covered. Actual sessions held may differ.

The content of this module will be taught from a cyber security perspective.

- executable code from a variety of perspectives
- assembly language programming
- machine-level instruction set and organisation
- code generation
- reverse engineering techniques
- de-obfuscation
- common tools for reverse engineering
- anti-debugging mechanisms
- fuzzing

Learning outcomes

By the end of the module, students should be able to:

- Demonstrate a critical understanding of common idioms and patterns used during code transformation and explain the origin and organisation of arbitrary code and/or data fragments within an executable program [CITP 2.1.8, 3.1.2]
- Apply tools and techniques as appropriate to infer the executable's overall high-level function, potentially obfuscated, potentially malicious code [CITP 2.1.3, 2.1.4, 2.1.5, 2.1.8, 2.2.2, 3.1.1, 3.1.2, 3.2.2]
- Perform malicious code analysis, vulnerability identification and evaluation independently from the findings generated by automated analysis tools [CITP 2.1.3, 2.1.4, 2.1.5,2.1.8, 2.2.2, 3.1.1, 3.1.2, 3.2.2]
- Identify system vulnerabilities, and consequently develop and deploy custom exploits [CITP 2.1.3, 2.1.4, 2.1.5,2.1.8,2.2.2, 3.1.1, 3.1.2, 3.2.2]

Indicative reading list

Reading lists can be found in Talis

Specific reading list for the module

Subject specific skills

Code transformation pattern recognition;
Executable code and data fragment analysis;
Advanced tool use for code function inference;
Obfuscated and malicious code analysis.

Transferable skills

Critical thinking, problem solving

Study

Study time

Type	Required
Lectures	10 sessions of (0%)
Supervised practical classes	20 sessions of 1 hour (13%)
Online learning (independent)	20 sessions of 1 hour (13%)
Private study	50 hours (33%)
Assessment	60 hours (40%)
Total	150 hours

Private study description

Independent study time is intended to support student learning through activities such as background reading, reviewing lecture materials, engaging with supplementary resources, and practicing key skills. It is not directly allocated to assessment preparation, which is accounted for separately in the assessment components.

Lecture content is integrated within workshop sessions, providing a blended approach that combines theoretical input with practical application.

Costs

No further costs have been identified for this module.

Assessment

You must pass all assessment components to pass the module.

Assessment group A

	Weighting	Study time	Eligible for self-certification
Assessment component			
Portfolio of Knowledge & Skills	60%	36 hours	Yes (extension)

Students will build a malware analyst’s casebook on a series of malware artefacts, progressing from initial triage to professional reporting. They begin with static and dynamic analysis, then advance to reverse engineering and detection rule creation, culminating in a threat intelligence report mapped to the MITRE ATT&CK framework.

Reassessment component is the same

Assessment component			
Reverse Engineering and Exploitation Project	40%	24 hours	No

This assessment comprises a group project designed to evaluate students’ practical and analytical skills in malware analysis, reverse engineering, and binary exploitation. Peer Marking Process will be adopted in this assessment.

Reassessment component is the same

Feedback on assessment

Formative feedback during lab sessions
Summative feedback on assessments

Availability

Courses

This module is Core optional for:

- UWMA-H651 Undergraduate Cyber Security
  - Year 3 of H651 Cyber Security
  - Year 3 of H651 Cyber Security
  - Year 3 of H651 Cyber Security