Prifysgol **Wrecsam**
Wrexham University

Module specification

When printed this becomes an uncontrolled document. Please access the Module Directory for the most up to date version by clicking on the following link: Module directory

Module Code	CONL711
Module Title	Secure Software Development
Level	7
Credit value	15
Faculty	FACE
HECoS Code	100374
Cost Code	GACP

Programmes in which module to be offered

Programme title	Is the module core or option for this programme
MSc Computer Science with Cyber Security	Core
MSc Computer Science with Software Engineering	Core
MSc Computer Science with UX	Core

Pre-requisites

None

Breakdown of module hours

Learning and teaching hours	15 hrs
Placement tutor support	0 hrs
Supervised learning e.g. practical classes, workshops	0 hrs
Project supervision (level 6 projects and dissertation modules only)	0 hrs
Total active learning and teaching hours	**15** hrs
Placement / work based learning	0 hrs
Guided independent study	135 hrs
Module duration (total hours)	150 hrs

For office use only	
Initial approval date	04/09/19
With effect from date	01/01/20

For office use only	
Date and details of revision	27/06/2024 Programme revalidation
Version number	2

Module aims

The module will allow students to understanding and apply the theory and practice of exploiting vulnerabilities in software as well as key skills of design and implementation of secure software. Students will learn the ability to implement secure systems and environments to support software security. Additionally, they will explore the use of secure programming languages and the effects on secure software. The use obfuscation and encryption in the protection of software will also be investigated.

Module Learning Outcomes - at the end of this module, students will be able to:

1	Conduct in-depth research to compare and contrast various approaches to software and system security.
2	Utilise and adapt secure programming techniques effectively. Demonstrate advanced skills in applying security principles to programming, including the ability to modify and enhance existing code to meet high standards of security in complex software environments
3	Critically evaluate different approaches to obfuscation, encryption, and signing within software and security contexts.
4	Identify and critically evaluate weaknesses in computer software and systems with a high level of expertise.
5	Select, justify, and document the most appropriate approaches, methods, and techniques used to secure software.

Assessment

This section outlines the type of assessment task the student will be expected to complete as part of the module. More details will be made available in the relevant academic year module handbook.

Indicative Assessment Tasks:

For Assessment 1 students will be given a case study, and asked to identify, document and present the potential security issues along with planning solutions. This will involve analysing system documentation and evaluating samples. Throughout the module, students will develop an understanding of several aspects of secure software development, including appropriate techniques and design strategies. This will develop their understanding of appropriate practices and code assessment techniques. This understanding will then be tested during Assessment 2 in the form of an in-class test with an indicative length of 90 minutes.

Assessment number	Learning Outcomes to be met	Type of assessment	Weighting (%)
1	2,3,5	Coursework	70%
2	1,4	In-class test	30%

Derogations

None

Learning and Teaching Strategies

The overall learning and teaching strategy is one of guided independent study requiring ongoing student engagement. Online material will provide the foundation of the learning resources, requiring the students to log in and engage regularly throughout the eight weeks of the module. There will be a mix of suggested readings, discussions and interactive content containing embedded digital media and self-checks for students to complete as they work through the material and undertake the assessment tasks. A range of digital tools via the virtual learning environment and additional sources of reading will also be utilised to accommodate learning styles. There is access to a helpline for additional support and chat facilities through Canvas for messaging and responding.

Indicative Syllabus Outline

- Memory models.
- Programming bugs and mistakes that lead to vulnerabilities.
- Secure programming languages and frameworks.
- Attacks against software, and other software related attacks: e.g. XSS attacks, SQL injection, etc.
- Programming for security.
- Software and system protection methods.
  ‘Secure by design’ development.

Indicative Bibliography:

Please note the essential reads and other indicative reading are subject to annual review and update.

Essential Reads

License to access SudoCyber online platform.

Other indicative reading

M. Howard, D. LeBlanc, and J. Viega, *24 Deadly Sins of Software Security: Programming Flaws and How to Fix Them*. New York, NY: McGraw-Hill, 2009.

A. Hoffman, *Web Application Security: Exploitation and Countermeasures for Modern Web Applications*. Sebastopol, CA: O’Reilly Media, 2020.

S. Azad and A. S. K. Pahtan, *Practical Cryptography: Algorithms and Implementations Using C++*. Boca Raton, FL: Taylor & Francis, 2014.

C. Cachin, R. Guerraoui, and L. Rodrigues, *Introduction to Reliable and Secure Distributed Programming*. Berlin, Germany: Springer, 2011.

R. C. Seacord, *Secure Coding in C and C++*. Upper Saddle River, NJ: Addison-Wesley, 2013.

A. Shalloway, S. Bain, K. Pugh, and A. Kolsky, *Essential Skills for the Agile Developer: A Guide to Better Programming and Design*. Boston, MA: Addison-Wesley, 2011.