

**MODULE SPECIFICATION**

|  |
| --- |
| **Part 1: Information** |
| **Module Title** | Operating systems security and defensive programming |
| **Module Code** | CY202 | **Level** | 5  |
| **For implementation from** | September 2020  |
| **UWE Credit Rating** | 30 | **ECTS Credit Rating** | 15 |
| **Faculty** | Environment and Technology | **Field** |  |
| **Department** | Computer Science and Creative Technologies |
| **Contributes towards**  | BSc (Hons) Cyber Security Technical Professional Compulsory |
| **Module type:**  | Standard  |
| **Pre-requisites**  | None |
| **Excluded Combinations**  | None  |
| **Co- requisites**  | None  |
| **Module Entry requirements** | None |
|  |
| **Part 2: Description**  |
| This module introduces apprentices to the tasks of operating systems such as controlling and allocating memory, prioritising system requests, controlling input and output devices, facilitating data networking and managing files, including security and protection. Apprentices will learn the concepts of security protection in operating systems, such as hierarchical protection domains, and how they are employed to resist malware threats. A design pattern is a description of how to solve a problem that can be used in many different situations and can help deepen the understanding of object-orientated programming and help improve software design and reusability. They can also be used for secure programming and apprentices will learn how to apply them along with other methods and tools. Lecture sessions cover the technical knowledge required. Designated practical work is included to ensure that apprentices have absorbed and understood the key principles involved.This module will be based on ensuring that apprentice’s practical skills and knowledge gained in the block release sessions are carried into the workplace to inform their employment and generation of evidence of competency. You will cover:* security and protection
* kernel security and protection
* typical OS security features and how they may be exploited
* approaches to defensive programming, for example input validation, least privilege, defence in depth, data sanitization, etc
* resistance to malware techniques such as memory corruption, code injection, user/kernel space vulnerabilities, privilege escalation, etc.
* design patterns for developing secure software
* use of compiler features to support the creation of secure code
* static and dynamic code analysis techniques
* sources of secure programming practices, including employer or software development organisation, for different types of software systems (e.g., OWASP, CERT, etc.)
* at least 1 formal method that may be applied to software development and its strengths and weaknesses when applied to development of software with security properties
 |
| **Part 3: Assessment**  |
| This module is assessed by a combination of techniques: a report (3,000 words) and a practical.Component AApprentices will write a 3,000-word report that will require them to research the most prevalent threats to operating systems. They will then describe how operating system security features protect against these threats. Finally, they will show how operating systems should be configured to take the most advantage of these features.Component BApprentices will be given a specification from which they have to produce a solution. This must incorporate the use of secure design patterns. They must show the research they have done to select the secure programming practices they will employ. As well as demonstrating the solution, apprentices will have to produce evidence of the design, development, implementation, test and debug.  |
|  |
| Identify final timetabled piece of assessment (component and element) | **Component B1** |
| **% weighting between components A and B** (Standard modules only) | **A:**  | **B**:  |
| **40%** | **60%** |
|  |
| **First Sit** |
| **Component A** **Description of each element** | **Element weighting****(as % of component)** |
| 1. Report (3,000 words)
 | 100% |
| **Component B** **Description of each element** | **Element weighting****(as % of component)** |
| 1. Researched and documented practical solution
 | 100% |
| **Resit (further attendance at taught classes is not required)** |
| **Component A** **Description of each element** | **Element weighting(as % of component)** |
| 1. Report (3,000 words) | 100% |
| **Component B Description of each element** | **Element weighting(as % of component)** |
| 1. Researched and documented practical solution | 100% |
|  |
| **Part 4: Learning Outcomes & KIS Data** |
| **Learning Outcomes** | On successful completion of this module students will be able to:Configure an Operating System in accordance with security policy. (Component A)Identify threats and the features that mitigate the threats (Component A)Apply secure programming principles and design patterns to address security issues (Component B)Research sources of secure programming practices and apply them (Component B) |
| **Key Information Sets Information (KIS)****Contact Hours****Total Assessment** |  The table below indicates as a percentage the total assessment of the module which constitutes a;**Written Exam**: Unseen or open book written exam**Coursework**: Written assignment or essay, report, dissertation, portfolio, project or in class test **Practical Exam**: Oral Assessment and/or presentation, practical skills assessment, practical exam (i.e. an exam determining mastery of a technique)  |
| **Reading List** | Reading list to be added |

***FOR OFFICE USE ONLY***

|  |  |
| --- | --- |
| **First Approval Date (and panel type)** | *Date of first {panel} approval*  |
| **Revision ASQC Approval Date** *Update this row each time a change goes to ASQC* |  | **Version**  | *1* | *Link to RIA*  |
|  |  |  |
|  |  |  |
|  |  |  |