H8SAP: Secure Application Programming

Module Code:		H8SAP					
Long Title		Secure Application Programming APPROVED					
Title		Secure Application Programming					
Module Level:		LEVEL 8					
EQF Level:		6					
EHEA Level:		First Cycle					
Credits:		10					
Module Coordinator:							
Module Author:		Alex Courtney					
Departments:		School of Computing					
Specifications of the qualifications and experience required of staff		aster's degree/PhD in Computing or cognate discipline.					
Learning Outco	mes						
On successful co	mpletion of this modu	ule the learner will be able to:					
#	Learning Outcome	ning Outcome Description					
LO1	Identify and analyse vulnerabilities.	entify and analyse common software vulnerabilities and investigate counter-measures to mitigate the threats to applications resulting from such Inerabilities.					
LO2	Evaluate, develop ar practices/standards,	te, develop and implement programming solutions for securing software applications using relevant programming solutions, secure coding es/standards, programming languages and applying secure software development lifecycle processes.					
LO3	Appraise trade-offs in	ise trade-offs in performance, usability, and other quality attributes that must be balanced when developing secure code.					
LO4	Identify, analyse and testing.	dentify, analyse and evaluate the ethical effects and impacts of design decision, the ethical issues in disclosing vulnerabilities and the ethics of thorough esting.					
Dependencies							
Module Recommendations							
No recommendations listed							
Co-requisite Modules							
No Co-requisite modules listed							
Entry requirements		Learners should have attained the knowledge, skills and competence gained from stage 3 of the BSc (Hons) in Computing.					

H8SAP: Secure Application Programming

Module Content & Assessment								
Indicative Content								
Review of Secure Software Development Lifecycle. Principles of Secure Design Review of Secure Software Development Lifecycle – include waterfall model, agile model and security; . Principles of Secure Design (least privilege, fail-safe, complete mediation, separation, minimize trust, economy of mechanism, principles of least astonishment – Usable Security, etc.)								
Introduction to Secure Coding Security support for programming languages. Type safety and its importance. Intro to Secure Coding – secure coding principles, standards, etc Seven Pernicious Kingdoms.								
Basic Web Security Model Same origin policy. HTTP/HTTPS & security extensions. JavaScript security								
Authentication, Authorization & Session Management Authentication and Authorization vulnerabilities & good secure practices. Secure session lifecycle; Session related vulnerabilities (e.g. session fixation, hijacking, etc.)								
Secure Coding: Validation of the input and its representation Input validation and data sanitization. Examples of input validation and data sanitization errors (e.g XSS vulnerability. SQL/NoSQL injection. Integer overflow. Buffer overflow, heap overflow. Format string attacks. Other injection attacks (e.g. OS command injection). XML vulnerabilities.								
OS Exploit Mitigation Data Execution Prevention/Non-Executable	OS Exploit Mitigation Data Execution Prevention/Non-Executable Stack. Return-to-libc and Return Oriented programming. Address Space Layout Randomisation							
Time and State Race Condition and TOCTOU. Defences for	or Race Conditions							
Security Testing Code review. Static and Dynamic Analysis								
Ethics in software development, testing and vulnerability disclosure. code reuse (licensing), professional responsibility, codes of ethics such as the ACM/IEEE-CS Software Engineering Code of Ethics and Professional Practice. Consequences and implications of poor or non-secure programming practices. How to disclose, to whom to disclose and when to disclose vulnerabilities. What, when and why to test – ethical implications of testing								
Assessment Breakdown			%					
Coursework			100.00%					
Assessments								
Full Time								
Coursework								
Assessment Type:	Formative Assessment	% of total:	Non-Marked					
Assessment Date:	n/a	Outcome addressed:	1,2,3,4					
Non-Marked:	Yes							
Assessment Description: Students will be required to perform variou requires in many cases that they should p	us tasks, including hacking games, ain ut their black hat on). Code review exe	ning to support them in understanding bett prcises with immediate feedback from the	ter how security testing should be done (as this lecturer, case studies, etc.					
Assessment Type:	Continuous Assessment	% of total:	40					
Assessment Date:	n/a	Outcome addressed:	1,2					
Non-Marked:	No							
Assessment Description: Practical work will be conducted throughout the semester to assess the learner's skills in terms of vulnerability identification and exploit, fixing code vulnerabilities and secure application development.								
Assessment Type:	Project	% of total:	60					
Assessment Date:	n/a	Outcome addressed:	2,3,4					
Non-Marked:	No							
Assessment Description: Learners are to develop an application from scratch employing a secure development lifecycle model. The project will have also a collaborative component as students will pair for the code review testing of their projects. Learners must also compile an associated report detailing the development process and how security characteristics have been incorporated into the working application.								
No End of Module Assessment								
No Workplace Assessment								
Reassessment Requirement								
Repeat examination Reassessment of this module will consist of a repeat examination. It is possible that there will also be a requirement to be reassessed in a coursework element.								
Reassessment Description Coursework Only This module is reassessed solely on the basis of re-submitted coursework. There is no repeat written examination.								

H8SAP: Secure Application Programming

Module Workload								
Module Target Workload Hours 0 Hours								
Workload: Full Time								
Workload Type	Workload Description	Hours	Frequency	Average Weekly Learner Workload				
Lecture	Classroom & Demonstrations (hours)	24	Every Week	24.00				
Tutorial	Other hours (Practical/Tutorial)	24	Every Week	24.00				
Independent Learning	Independent learning (hours)	202	Every Week	202.00				
Total Weekly Contact Hours								

Module Resources						
Recommended Book Resources						
Laura Bell, Michael Brunton-Spall, Rich Smith. (2016), Agile Application Security, O'Reilly Media, p.300, [ISBN: 978-1491938843].						
Jim Manico,August Detlefsen. (2014), Iron-Clad Java, McGraw Hill Professional, p.304, [ISBN: 978-0-07-183589-3].						
Matt Bishop. (2018), Computer Security, Addison-Wesley Professional, p.1440, [ISBN: 978-0-321-71233-2].						
Article/Paper List.						
Туре.						
ltem.						
Disselkoen, C., Renner, J., Watt, C., Garfinkel, T., Levy, A. and Stefan, D. (0), Position Paper: Progressive Memory Safety for WebAssembly, Proceedings of the 8th International Workshop on Hardware and Architectural Support for Security and Privacy, n/a,.						
Roemer, R., Buchanan, E., Shacham, H. and Savage, S (2012), , Return-oriented programming: Systems, languages, and applications, ACM Transactions Information and System Security (TISSEC), 15(1), p, 2, n/a.						
This module does not have any article/paper resources						
This module does not have any other resources						
Discussion Note:						