OTHM LEVEL 4 DIPLOMA IN CYBER SECURITY

Qualification Number: 610/5103/2

OTHM LEVEL 5 DIPLOMA IN CYBER SECURITY

Qualification Number: 610/5104/4

OTHM LEVEL 5 EXTENDED DIPLOMA IN CYBER SECURITY

Qualification Number: 610/5102/0

NOVEMBER 2024

TABLE OF CONTENTS

QUALIFICATION OBJECTIVES

The objective of the OTHM Level 4, Level 5 and Level 5 Extended Diploma in Cyber Security is to give learners a solid background in the theoretical and practical skills necessary to understand the range of cyber security risks that face individuals and organisations in the modern day.

Learners will gain a background in fundamental topics in computer science and mathematics that will help them to competently explain and reason about topics in the domain of security. After completing the qualification, learners will have a broad understanding of information technology, computer architecture and networks, and operating systems. They will also have acquired a wealth of practical skills that enable them to develop essential digital applications and software such as databases and deep neural networks.

The qualification provides learners with a grasp of essential digital security practices and develops within them an understanding of the mindset of malicious threat actors. Learners will gain an appreciation for the common security problems that are exploited, as well as the skills to understand, analyse, mitigate, prevent, and recover from modern cyber security attacks.

Successful completion of this qualification will allow learners to take on a variety of roles within the field of cyber security with confidence, and to progress to further study in the fields of computer science or security.

QUALITY, STANDARDS AND RECOGNITIONS

OTHM Qualifications are approved and regulated by Ofqual (Office of Qualifications and Examinations Regulation). Visit the Register of Regulated Qualifications.

OTHM has progression arrangements with several UK universities that acknowledges the ability of learners after studying Level 3-7 qualifications to be considered for advanced entry into corresponding degree year/top up and Master's/top-up programmes.

REGULATORY INFORMATION

Qualification Titles	OTHM Level 4 Diploma in Cyber Security OTHM Level 5 Diploma in Cyber Security OTHM Level 5 Extended Diploma in Cyber Security
Ofqual Qualification Numbers	OTHM Level 4 Diploma in Cyber Security - 610/5103/2 OTHM Level 5 Diploma in Cyber Security - 610/5104/4 OTHM Level 5 Extended Diploma in Cyber Security - 610/5102/0
Regulation Start Date	29/11/2024
Operational Start Date	29/11/2024
Duration	OTHM Level 4 Diploma in Cyber Security - 1 Year OTHM Level 5 Diploma in Cyber Security - 1 Year OTHM Level 5 Extended Diploma in Cyber Security - 2 Years
Total Credit Value	OTHM Level 4 Diploma in Cyber Security - 120 Credits OTHM Level 5 Diploma in Cyber Security - 120 Credits OTHM Level 5 Extended Diploma in Cyber Security - 240 Credits

Total Qualification Time (TQT)	OTHM Level 4 Diploma in Cyber Security - 1200 Hours OTHM Level 5 Diploma in Cyber Security - 1200 Hours OTHM Level 5 Extended Diploma in Cyber Security - 2400 Hours
Guided Learning Hours (GLH)	OTHM Level 4 Diploma in Cyber Security - 600 Hours OTHM Level 5 Diploma in Cyber Security - 600 Hours OTHM Level 5 Extended Diploma in Cyber Security - 1200 Hours
Sector Subject Area (SSA)	06.1 Digital Technology (practitioners)
Overall Grading Type	Pass / Fail
Assessment Methods	Coursework
Language of Assessment	English

EQUIVALENCES

OTHM Level 4 and 5 or Level 5 Extended Diploma qualifications represent practical knowledge, skills, capabilities and competences that are assessed in academic terms as being equivalent to Higher National Diplomas (HND) and Year 2 of a three-year UK Bachelor's degree programme.

QUALIFICATION STRUCTURE

OTHM Level 4 Diploma in Cyber Security qualification consists of 6 mandatory units, 120 credits, 1200 hours Total Qualification Time (TQT) and the recommended Guided Learning Hours (GLH) for this qualification is a minimum of 600 hours.

OTHM Level 5 Diploma in Cyber Security qualification consists of 6 mandatory units, 120 credits, 1200 hours Total Qualification Time (TQT) and the recommended Guided Learning Hours (GLH) for this qualification is a minimum of 600 hours.

OTHM Level 5 Extended Diploma in Cyber Security qualification consists of 12 mandatory units, 240 credits, 2400 hours Total Qualification Time (TQT) and the recommended Guided Learning Hours (GLH) for this qualification is a minimum of 1200 hours.

All units are mandatory.

Unit Ref No	Unit Title	Level	Credit	GLH	TQT
A/651/4218	Information Technology Security	4	20	100	200
D/651/4219	Principles of Computer Science	4	20	100	200
J/651/4220	Algorithms and Data Structures	4	20	100	200
K/651/4221	Computer Networks	4	20	100	200
L/651/4222	Mathematics for Computer Science	4	20	100	200
M/651/4223	Operating Systems	4	20	100	200
R/651/4224	Security Testing	5	20	100	200
T/651/4225	Artificial Intelligence	5	20	100	200
Y/651/4226	Databases	5	20	100	200
A/651/4227	Digital Forensics	5	20	100	200
D/651/4228	Ethical Hacking	5	20	100	200

F/651/4229	Malware Analysis	5	20	100	200

DEFINITIONS

Total Qualification Time (TQT) is the number of notional hours which represents an estimate of the total amount of time that could be expected to be required in order for a learner to achieve and demonstrate the achievement of the level of attainment necessary for the award of a qualification.

Total Qualification Time is comprised of the following two elements –

a) the number of hours which an awarding organisation has assigned to a qualification for Guided Learning, and

b) an estimate of the number of hours a Learner will reasonably be likely to spend in preparation, study or any other form of participation in education or training, including assessment, which takes place as directed by – but, unlike Guided Learning, not under the Immediate Guidance or Supervision of – a lecturer, supervisor, tutor or other appropriate provider of education or training.

(Ofqual 15/5775 September 2015)

Guided Learning Hours (GLH) are defined as the hours that a teacher, lecturer or other member of staff is available to provide immediate teaching support or supervision to a learner working towards a qualification.

Credit value is defined as being the number of credits that may be awarded to a learner for the successful achievement of the learning outcomes of a unit. One credit is equal to 10 hours of TQT.

ENTRY REQUIREMENTS

These qualifications are designed for learners who are typically aged 18 and above.

The entry profile for learners is likely to include at least one of the following:

- Relevant Level 3 Diploma qualification or equivalent qualification
- GCE Advanced level in 3 subjects or equivalent qualification
- Mature learners (over 21) with relevant management experience (learners must check with the delivery centre regarding this experience prior to registering for the programme)

English requirements: If a learner is not from a majority English-speaking country, they must provide evidence of English language competency. For more information visit the English Language Expectations page on the OTHM website.

PROGRESSION

Successful completion of the OTHM Level 4, Level 5 or Level 5 Extended Diploma in Early Cyber Security provides learners with the opportunity to access a wide range of academic progression.

As this qualification is approved and regulated by Ofqual (Office of the Qualifications and Examinations Regulation), learners are eligible to gain direct entry into a UK Bachelor's

degree programme. For more information visit the University Progressions page on the OTHM website.

DELIVERY OF OTHM QUALIFICATIONS

OTHM do not specify the mode of delivery for its qualifications, therefore OTHM centres are free to deliver this qualification using any mode of delivery that meets the needs of their learners. However, OTHM centres should consider the learners' complete learning experience when designing the delivery of programmes.

It is important that centres develop an effective delivery method to teaching and learning that supports the progression and stretch of learners.

OTHM Centres must ensure that the chosen mode of delivery does not unlawfully or unfairly discriminate, whether directly or indirectly, and that equality of opportunity is promoted. Where it is reasonable and practicable to do so, it will take steps to address identified inequalities or barriers that may arise.

Guided Learning Hours (GLH) which are listed in each unit gives centres the number of hours of teacher-supervised or direct study time likely to be required to teach that unit.

ASSESSMENT AND VERIFICATION

All units within this qualification are assessed and internally quality assured by the centre and externally verified by OTHM. The qualifications are Criteria referenced, based on the achievement of all the specified learning outcomes.

To achieve a 'pass' for a unit, learners must provide evidence to demonstrate that they have fulfilled all the learning outcomes and meet the standards specified by all assessment criteria. Judgement that the learners have successfully fulfilled the assessment criteria is made by the assessor.

Specific assessment guidance and relevant marking criteria for each unit are made available in the Assignment Brief document. These are made available to centres immediately after registration of one or more learners.

The assessor should provide an audit trail showing how the judgement of the learners' overall achievement has been arrived at.

Assessment Tracking and Recording Learner Progress

It is necessary to track and record learner achievement throughout the delivery period of the Diploma and this should not be left until the end of the course.

This will include requalification gular review of learner work through formative and summative assessment and internal quality assurance at planned intervals during the programme:

- before decisions have been made on any unit
- sampling evidence once one or two of the units or assignments are completed

Tracking learner progress, recording the achievement of each learner per criteria on a unit-by-unit basis ensures:

- the assessment evidence is clearly measured against national standards

- learner progress is accurately tracked
- the assessment process can be reliably verified
- evidence is valid, authentic and reliable for the safety of certification
- identification of which assessments are outstanding
- internal verification is timely
- samples for standards verification and other external audits can be made available as required
- up to date, securely stored assessment records help to minimise the risk of assessment malpractice and potential issues; maintaining the integrity of the qualification.

Tutors/Assessors should provide learners with formative and summative feedback to aid development during their studies.

Formative Assessment

Formative assessment is an integral part of the assessment process, involving both the Tutor/Assessor and the learner about their progress during the course of study.
Formative assessment takes place prior to summative assessment and focuses on helping the learner to reflect on their learning and improve their performance and does not confirm achievement of grades at this stage.

The main function of formative assessment is to provide feedback to enable the learner to make improvements to their work. This feedback should be prompt so it has meaning and context for the learner and time must be given following the feedback for actions to be complete. Feedback on formative assessment must be constructive and provide clear guidance and actions for improvement.

All records should be available for auditing purposes, as we may choose to check records of formative assessment as part of our ongoing quality assurance.

Summative Assessment

Summative assessment is used to evaluate learner competence and progression at the end of a unit or component. Summative assessment should take place when the assessor deems that the learner is at a stage where competence can be demonstrated.

Learners should be made aware that summative assessment outcomes are subject to confirmation by the Internal Verifier and External Quality Assurer (EQA) and thus is provisional and can be overridden. Assessors should annotate on the learner work where the evidence supports their decisions against the assessment criteria. Learners will need to be familiar with the assessment and grading criteria so that they can understand the quality of what is required.

Evidence of both formative and summative assessment MUST be made available at the time of external quality assurance – EQA.

RECOGNITION OF PRIOR LEARNING AND ACHIEVEMENT

Recognition of Prior Learning (RPL) is a method of assessment that considers whether learners can demonstrate that they can meet the assessment requirements for a unit through knowledge, understanding or skills they already possess and do not need to develop through a course of learning.

RPL policies and procedures have been developed over time, which has led to the use of a number of terms to describe the process. Among the most common are:

- Accreditation of Prior Learning (APL)
- Accreditation of Prior Experiential Learning (APEL)
- Accreditation of Prior Achievement (APA)
- Accreditation of Prior Learning and Achievement (APLA)

All evidence must be evaluated with reference to the stipulated learning outcomes and assessment criteria against the respective unit(s). The assessor must be satisfied that the evidence produced by the learner meets the assessment standard established by the learning outcome and its related assessment criteria at that particular level.

Most often RPL will be used for units. It is not acceptable to claim for an entire qualification through RPL. Where evidence is assessed to be only sufficient to cover one or more learning outcomes, or to partly meet the need of a learning outcome, then additional assessment methods should be used to generate sufficient evidence to be able to award the learning outcome(s) for the whole unit. This may include a combination of units where applicable.

EQUALITY AND DIVERSITY

OTHM provides equality and diversity training to staff and consultants. This makes clear that staff and consultants must comply with the requirements of the Equality Act 2010, and all other related equality and diversity legislation, in relation to our qualifications.

We develop and revise our qualifications to avoid, where possible, any feature that might disadvantage learners because of their age, disability, gender, pregnancy or maternity, race, religion or belief, and sexual orientation.

If a specific qualification requires a feature that might disadvantage a particular group (e.g. a legal requirement regarding health and safety in the workplace), we will clarify this explicitly in the qualification specification.

LEVEL 4 UNIT SPECIFICATIONS

INFORMATION TECHNOLOGY SECURITY

Unit Reference Number	A/651/4218
Unit Title	Information Technology Security
Unit Level	4
Number of Credits	20
Total Qualification Time (TQT)	200 Hours
Guided Learning Hours (GLH)	100 Hours
Mandatory / Optional	Mandatory
Sector Subject Area (SSA)	06.1 Digital Technology (practitioners)
Unit Grading Type	Pass / Fail

Unit Aims

This unit will introduce the basic principles of security in digital systems - cyber security - explaining the role of cyber security in protecting digital systems, networks, software, and data from unauthorised access, theft, damage, or disruption. This unit will help learners to understand the nature of digital information and the importance of keeping information secure, as well as methods for doing so. As part of the unit, learners will develop an appreciation for the broad space of threats that cyber systems face, and will understand how these threats can be prevented, mitigated, and responded to with appropriate frameworks; practices and principles; and techniques such as encryption and hashing.

Learning Outcomes, Assessment Criteria and Indicative Content

Learning Outcome – The learner will:	Assessment Criteria – The learner can:	Indicative Content
1. Understand the nature of information and what is meant by information privacy.	1.1 Define information and give examples of information. 1.2 Explain what is meant by information privacy. 1.3 Analyse the value of information from contrasting perspectives.	**Information and its value** ● Different views and definitions of information and its value: information theoretic definitions (choice and uncertainty), semantic (meaningful to humans), impact oriented (the impact of the information on individuals, organisations, and societies). ● The classes of information that are commonly stored, transmitted, and accessed: files and

	1.4 Analyse the importance of information privacy and the potential consequences when information is not kept private.	documents, messages and communications, computer systems and software. **Information privacy** ● Defining privacy in an intuitive sense, formal and quantitative definitions of privacy (e.g. differential privacy), the privacy-utility trade-off, introducing ethical considerations for privacy (e.g. encrypted communications). ● Consequences of privacy breaches at different scales: personal, corporate, societal and national; including impact on reputation, loss of intellectual property, financial risks, safety and security risks.
2. Understand key concepts in information security, including threats and risks.	2.1 Describe the common types of information security threats for modern computing systems. 2.2 Explain the Confidentiality, Integrity, and Availability (CIA Triad) framework and its meaning in the context of information security. 2.4 Analyse the role of risk management in identifying security threats and vulnerabilities. 2.3 Evaluate several key cyber security models and their focuses.	**Risks and threats** ● Types of threat: malware, ransomware, social engineering (including phishing etc.), exploitation of software/hardware vulnerabilities. ● Internal vs. external risks: human error, insider threats, hacking. **CIA Triad and Cyber Security Models** ● Definition of the different components of the CIA triad. ● Best practices and techniques for ensuring the confidentiality, integrity, and availability of information and possible consequences when aspects of the triad are breached. ● Authorisation and non-repudiation. ● Access Control Models: Bell LaPadula Model (BLP), Biba integrity model, Clark-Wilson Model, Role-Based Access Control (RBAC). ● Risk Management Models: NIST, ISO/ISEC 27001. ● Zero-Trust cyber security model.

		Risk Management for Threats and Vulnerabilities ● Identifying risks and vulnerabilities, the risk assessment process in the context of cyber systems. ● Threat modelling approaches: e.g. STRIDE, DREAD, attack trees. ● Security controls and countermeasures. *N.B. risk management approaches will be covered in depth in the Security Testing unit and need only be introduced in summary here.*
3. Understand the importance of secure design in Cyber Security systems.	3.1 Describe the classes of common vulnerabilities in software and hardware systems. 3.2 Explain the principles of secure design of cyber systems. 3.3 Analyse a range of strategies for implementing secure design to prevent and mitigate cyberattacks.	**Classes of Vulnerability** ● Human vulnerabilities: phishing (and related techniques such as smishing, vishing etc.), human error, vulnerable user credentials, inappropriate access control. ● Software vulnerabilities: misconfigurations, outdated and unpatched software, unsecured application programming interfaces (APIs), buffer overflows, SQL injection, cross-site scripting (XSS), cloud-based vulnerabilities, zero-day vulnerabilities. ● Hardware vulnerabilities: Side-channel attacks (e.g. cache attacks, timing attacks, power-monitoring, data remanence), firmware attacks. **Secure System Design Principles** ● Separation of duties, Principle of Least Astonishment (POLA), domain separation, least privilege, defence in depth, fail-safe defaults, secure baselines. ● Secure software development principles: security by design, information hiding, Secure

		Software Development Lifecycle (SDLC). **Vulnerability Mitigation through Secure Design** ● Vulnerability assessment, penetration testing, runtime testing, code review, secure coding practices e.g. OWASP developer guide. ● Practical implementation of best practices in secure design frameworks e.g. UK Government Secure by Design Principles. *N.B. Secure system design and vulnerability mitigation approaches are covered in depth in the Security Testing unit and need only be overviewed briefly here.*
4. Be able to apply cryptographic techniques to encrypt and decrypt information.	4.1 Describe key historic events leading up to the development of modern cryptographic techniques 4.2 Outline basic illustrative cipher techniques. 4.3 Compare encryption to hashing. 4.4 Explain the value of cryptography for a range of cyber security applications. 4.5 Explain the basic mathematical principles underlying encryption. 4.6 Analyse a variety of encryption techniques. 4.7 Use software to implement public and private key encryption of information and network connections.	**History of Cryptography** ● Origins of cryptography and simple cryptographic schemes e.g. Caesar ciphers. ● World War cryptographic development. ● Modern cryptography. **Encryption Techniques** ● Prime factorisation vs. multiplication and modular arithmetic. ● Symmetric (e.g. AES) and asymmetric (e.g. RSA) encryption. ● Encryption of information at rest vs. in transit (e.g. SSL/TLS, VPNs). *N.B. The mathematics needed to understand encryption need only be introduced at a surface level, as it will be covered in more detail in the Mathematics for Computer Science unit.* **Hashing Algorithms and Data Verification** ● Encryption vs. hashing definitions. ● Properties of hash functions as compared to

		encryption schemes: e.g. fixed size, efficiency, pre-image resistance, collision resistance, determinism, the avalanche effect. ● SHA-256, SHA-3, and MD5 hash algorithms. **Public and Private Keys** ● The use of public and private keys for secure communication and authentication. ● Digital certificates and signatures and their operational principles. ● Key management and distribution: Public Key Infrastructure (PKI), certificate authorities, Trusted Platform Modules (TPMs), Hardware Security Modules (HSMs), and Key Management Systems (KMSs). **Encryption Technique Strengths and Weaknesses** ● Comparison of cryptographic algorithm strengths and weaknesses in different contexts e.g. choice of bit length for RSA and AES, encryption speed for symmetric vs. asymmetric algorithms and consequent use cases. ● Methods by which encryption is and could be broken e.g. brute force attacks, quantum computing. ● Best practices for encryption and key management.
5. Understand legal and societal issues concerning information security.	5.1 Explain the needs for legal and regulatory frameworks to govern Cyber Security and data protection. 5.2 Analyse the effectiveness of legal frameworks in ensuring information security.	**Legal Frameworks** ● Data privacy and protection in law: e.g. General Data Protection Regulation (GDPR), Computer Misuse Act, Data Protection Act, California Consumer Privacy Act (CCPA). **Ethical Considerations**

	5.3 Evaluate the trade-off between privacy and security in computer systems.	● Principles of ethical hacking, white hat vs. black hat vs. grey hat hackers, responsible disclosure of vulnerabilities. ● Surveillance and data privacy. ● Data rights and right to be forgotten. ● Trade-off between ensuring safety of users and security of information and excessive or invasive monitoring and surveillance. ● Case studies: e.g. COVID-19 Contact Tracing, Wikileaks and Edward Snowden, Smart Home Devices (e.g. Alexa).

Assessment

To achieve a 'pass' for this unit, learners must provide evidence to demonstrate that they have fulfilled all the learning outcomes and meet the standards specified by all assessment criteria.

Learning Outcomes to be met	Assessment Criteria to be covered	Assessment type	Word count (approx. length)
LO1, LO2, LO3 and LO5	All ACs under LO1 to LO3	Report	3000 words
LO4	All ACs under LO4	Report	1500 words

Indicative Reading List

Pfleeger C. P. & Pfleeger S. L. (2018) *Security in Computing (5th Edition)* ISBN 978-9352866533

Singh, S. (2000) *The Code Book: The Science of Secrecy from Ancient Egypt to Quantum Cryptography* ISBN 978-0385495325

Kim G., Behr K., Spafford G. (2013) *The Phoenix Project: A Novel about IT, DevOps, and Helping Your Business Win* ISBN 978-0988262591

Kim G. (2019) *The Unicorn Project: A Novel about Digital Disruption, Redshirts, and Overthrowing the Ancient Powerful Order: A Novel about Developers, Digital Disruption, and Thriving in the Age of Data* ISBN 978-1942788768

Hubbard D.W. & Seiersen R. (2023) *How to Measure Anything in Cybersecurity Risk (2nd edition)* ISBN 978-1119892304

Katz J. & Lindell Y. (2020) *Introduction to Modern Cryptography (3rd edition)* ISBN 978-0815354369

Hoffstein J., Pipher J., Silverman J. H. (2014) *An Introduction to Mathematical Cryptography (2nd edition)* ISBN 978-1493917105

Paar C., Pelzl J., Preneel B. (2014) Understanding Cryptography: A Textbook for Students and Practitioners ISBN 978-3642446498

Additional Resources

OWASP Developer Guide: https://owasp.org/www-project-developer-guide/

UK Government Secure by Design Principles: https://www.security.gov.uk/policy-and-guidance/secure-by-design/principles/

PRINCIPLES OF COMPUTER SYSTEMS

Unit Reference Number	D/651/4219
Unit Title	Principles of Computer Systems
Unit Level	4
Number of Credits	200
Total Qualification Time (TQT)	200 Hours
Guided Learning Hours (GLH)	100 Hours
Mandatory / Optional	Mandatory
Sector Subject Area (SSA)	06.1 Digital Technology (practitioners)
Unit Grading Type	Pass / Fail

Unit Aims

This unit introduces the foundational concepts, methodologies, and techniques of computer systems and hardware. It focuses on the essential elements of how computers operate, covering topics such as logic design, state machines, assembly-level representation, parallelisation, and performance evaluation. Students will explore system design both theoretically and practically, enhancing their understanding by implementing their own designs and critically analysing existing systems. By the end of the unit, students will have developed a deeper understanding of the different components of computer systems, and how they interact to perform information processing.

Learning Outcomes, Assessment Criteria and Indicative Content

Learning Outcome – The learner will:	Assessment Criteria – The learner can:	Indicative Content
1. Understand the architecture and key components of computer systems.	1.1 Describe the key hardware components of a computer system and their functions. 1.2 Explain how data is processed and transferred within a computer. 1.3 Explain how different components interact to improve system performance.	**Computer Components** ● Central Processing Unit (CPU) and its components (Arithmetic Logic Units (ALUs), Floating Point Units (FPUs)). ● Memory. ● Buses. ● Input/Output Devices. ● Peripherals. ● Specialised processors: Graphics Processing

	1.4 Evaluate the benefits and impact of parallel processing on computing performance.	Unit (GPUs), Field Programmable Gate Arrays (FPGAs). Data Processing ● Clock cycles, edges (rising/falling), principles of state machines. The impact of clock speed. ● The fetch-decode-execute cycle. ● How data is transferred to and from memory via buses. ● Parallel processing and pipelining. The advantages and challenges of parallel processing. Software considerations for parallel processing.
2. Understand the types and functions of memory in computing systems.	2.1 Describe the Von Neumann computing paradigm. 2.2 Compare the different forms of memory that computers use and their roles (primary, cache, and secondary). 2.3 Analyse the impact of processor-memory separation on computing performance. 2.4 Evaluate the difference between volatile and non-volatile memory in different computing contexts.	**Memory Organisation and Considerations** ● The Von-Neumann computing paradigm (‘store’ and ‘mill’) and how modern computers fit this model. ● The impact of distance between processors and memory on access speed and performance. Trade-off between access time and cost per bit in memory. ● Volatile vs. non-volatile memory. **Types and Roles of Computer Memory** ● CPU Registers. ● Cache Memory, Cache Levels (L1, L2, L3). Effects of cache hierarchy on performance. ● Primary Memory: Random Access Memory (RAM), Static Random Access Memory (SRAM), Dynamic Random Access Memory (DRAM), Read Only Memory (ROM). ● Secondary Storage: Hard Drives (HDD) and Solid-State Drives (SSD).

3. Be able to design and implement simple state machines.	3.1 Describe the principles of state machines and their role in computer systems. 3.2 Design state diagrams for a range of processing scenarios. 3.3 Implement basic state machines using programming languages and/or logic design tools. 3.4 Critically evaluate the performance of state machines and propose improvements.	**Principles of State Machines** ● Finite State Machines (FSMs), Mealy models, Moore models. ● State diagrams: transitions, inputs/outputs. **Implementation of State Machines** ● Design of basic state machines for well-known scenarios e.g. dishwasher, washing machine, vending machine. ● Use of digital tools for the implementation and testing of state machines, including Python and Verilog. *N.B. Here, an opportunity is presented to teach students some essential programming skills in e.g. Python, which will be built on later.* **State Machine Analysis and Testing** ● Testing and verification of state machine performance, debugging and edge case analysis.
4. Understand the fundamentals of logic circuits and gates.	4.1 Describe the representation of information in binary in computer systems. 4.2 Explain the common types of logic operation and corresponding logic gates. 4.3 Explain how logical operations are implemented using transistors in electronic circuits. 4.4 Explain how complex arithmetic operations can be built up from basic logic.	**Information Representation** ● Binary, decimal, and hexadecimal number systems. ● The advantages of different number systems. ● Correspondence between binary values and ON/OFF (high/low etc.) in computer memory and hardware. **Logic** ● Propositional logic, truth and falsehood. ● Binary logic: AND, OR, and NOT. ● NAND, NOR, XOR and XNOR. ● Truth tables.

	4.5 Implement basic arithmetic operations using logic gates.	*N.B. A brief introduction to binary numbers and propositional logic can be given as part of the teaching of this Learning Objective, but a more formal treatment is expected in the Mathematics for Computer Science unit.* **Logic Gates** ● Transistors as switches and historical development of transistors. Transistor types and Metal Oxide Semiconductor Field-Effect Transistors (MOSFETs). ● The implementation of logical operations using transistors. **Arithmetic Operations** ● The importance of fundamental arithmetic operations (half-adder, adder etc.) for computer processing. ● Implementation of logic gates and arithmetic circuits in Hardware Description Languages (HDLs) such as Verilog.
5. Understand the fundamentals of assembly language and how it relates to higher level programming languages.	5.1 Describe the role of assembly language in performing low-level computer operations. 5.2 Explain the different assembly languages that exist and evaluate their differences. 5.3 Analyse how higher-level programming languages are abstracted from lower-level languages and the benefits of doing so. 5.4 Implement basic routines in ARM assembly language.	**Fundamentals** ● Low-level machine code vs. assembly language vs. high-level programming languages. Principle of abstraction and levels of correspondence to machine code. Advantages and disadvantages of higher-level languages in terms of factors such as ease-of-programming, control over implementation, and efficiency. ● Importance of assembly language in computer systems. ● Different types of Assembly language instruction sets and why different assembly languages are needed: ARM, MIPS, x86, x86-64, PowerPC, RISC-V.

		Fundamentals of Assembly Programming ● Instruction sets and common instructions. ● Control flow: loops, jumps, and branches. ● Registers. ● Memory addressing. ● The stack. **Programming in Assembly** ● The ARM assembly language: essential instructions, registers etc. ● Implementation of basic and more complex arithmetic routines in assembly.

Assessment

To achieve a 'pass' for this unit, learners must provide evidence to demonstrate that they have fulfilled all the learning outcomes and meet the standards specified by all assessment criteria.

Learning Outcomes to be met	Assessment Criteria to be covered	Assessment type	Word count (approx. length)
LO1-LO3	All ACs under LO1 to LO3	Report and Diagram	2500 words
LO4-LO5	All ACs under LO4 to LO5	Report, Annotated Code and Circuit Diagram	2000 words

Indicative Reading List

Hennessy J. L. & Patterson D. A. (2011) *Computer Architecture: A Quantitative Approach (6th edition)* ISBN 978-0-12-811905-1

Patterson, D. A. & Hennessy, J. L**.** (2020) *Computer Organization and Design: The Hardware/Software Interface (6th edition)* ISBN 978-0128201091

Mano M. M. & Ciletti M. (2017) *Digital Design: With an Introduction to the Verilog HDL, VHDL, and SystemVerilog (6th edition)* ISBN 978-0134549897

Schocken S. & Nisan N. (2017) *From Nand to Tetris* https://www.nand2tetris.org/

Nisan N. & Schocken S. (2021) *The Elements of Computing Systems: Building a Modern Computer from First Principles (2nd edition)* ISBN 978-0262539807

Harris D. & Harris S. (2012) *Digital Design and Computer Architecture (2nd edition)* ISBN 978-0123944245

Petzold C. (2022) *Code: The Hidden Language of Computer Hardware and Software (2nd edition)* ISBN 978-0137909100

Warford J. S. (2016) *Computer Systems (5th edition)* ISBN: 978-1284079630

Von Neumann J. (1945, published 1993) *First Draft of Report on the EDVAC* IEEE Annals in the History of Computing Volume 15 Issue 4

Additional Resources

Writing ARM Assembly Language: https://developer.arm.com/documentation/dui0473/c/writing-arm-assembly-language

EDA Playground Verilog simulator: https://www.edaplayground.com/

VisUAL ARM assembly emulator: https://salmanarif.bitbucket.io/visual/

ALGORITHMS AND DATA STRUCTURES

Unit Reference Number	J/651/4220
Unit Title	Algorithms and Data Structures
Unit Level	4
Number of Credits	200
Total Qualification Time (TQT)	200 Hours
Guided Learning Hours (GLH)	100 Hours
Mandatory / Optional	Mandatory
Sector Subject Area (SSA)	06.1 Digital Technology (practitioners)
Unit Grading Type	Pass / Fail

Unit Aims

This module introduces data structures, algorithms, and the principles of computational complexity. Learners will become familiar with a variety of data structures and understand their similarities and differences. Both linear data structures such as arrays and lists, and non-linear data structures including graphs and trees, will be introduced. Time and space complexity, including notation for and analysis of complexity, are introduced as core considerations for algorithmic design. A range of popular sort, search, and pathfinding algorithms are introduced. Learners will gain practical experience in writing pseudocode for algorithms and using this to write, test, and debug programmed implementations of algorithms.

Learning Outcomes, Assessment Criteria and Indicative Content

Learning Outcome – The learner will:	Assessment Criteria – The learner can:	Indicative Content
1. Understand a range of essential data structures.	1.1 Describe a data structure. 1.2 Explain a range of linear and non-linear data structures. 1.3 Analyse the impact of the choice of data structure on algorithmic performance.	**Data structures** ● Linear data structures: arrays, lists and linked lists, stacks, queues, sets, dictionaries. ● Non-linear data structures: graphs and trees. **Considerations for choosing data structures** ● Impact on the time an algorithm takes to run, the space it takes up on the disk (can be used as an introduction and motivation for the next

		learning objective).
2. Understand algorithmic complexity and appreciate its importance.	2.1 Demonstrate an ability to write Big O and little o notation to describe algorithmic complexity. 2.2 Explain the impact of algorithmic design choice on computational complexity for a variety of examples. 2.3 Explain the difference between P and NP. 2.4 Evaluate open problems in problem classification and the impacts of future technologies on the difficulty in solving different classes algorithms.	**Measuring algorithmic complexity and performance** ● Big O and little o notation. ● Practical algorithm design and implementation considerations for time and space complexity: e.g. for loops and nested for loops, recursion and its impact, memory usage. **Problem classes** ● Computability and Turing machines. ● P space and NP classes (solving vs. verification in polynomial time). ● The P versus NP problem. ● NP-completeness, NP-hardness and consequences for algorithmic design, as well as future considerations e.g. quantum computing.
3. Understand a range of algorithmic techniques.	3.1 Use recursion and memoisation as tools to design solutions to problems. 3.2 Describe a range of sort, search, and pathfinding algorithms. 3.3 Explain the trade-off between memory and computational complexity in algorithmic design. 3.4 Critically analyse algorithmic paradigms such as the greedy approach and divide and conquer approach. 3.5 Evaluate sort, search, and pathfinding algorithms in terms of their space and time complexity.	**Recursion and memoisation** ● Recursion: use of recursion for solving problems e.g. Tower of Hanoi. ● Memoisation: trade-off between memory and complexity. **Algorithmic paradigms** ● Divide and conquer. ● Backtracking. ● Greedy algorithms. ● Dynamic Programming. ● Space and time complexity of different approaches. **Sort, search and pathfinding algorithms** ● Sorting algorithms: e.g. merge sort, bubble sort. ● Search and pathfinding algorithms: Brute-force,

		depth-first, breadth-first approaches; A* search, Dijkstra's algorithm. ● Comparison of aforementioned algorithms according to space and time complexity.
4. Be able to write algorithms in popular programming languages.	4.1 Explain how to write pseudo-code for algorithmic design. 4.2 Implement pseudocode in a programming language. 4.3 Evaluate the correctness and speed of an algorithm. 4.4 Use debugging to correct errors in algorithmic implementation.	Implementing Algorithms ● Writing pseudo-code and its purpose in providing a universal template for algorithmic implementation. ● Converting an algorithm written in pseudocode to a routine or function in a popular programming language e.g. Python. ● A refresher of the essentials of the given programming language would be useful: if statements, for loops etc. For Jupyter/Python: an introduction to Jupyter notebooks and magic functions. Evaluation ● Proving the correctness of algorithms formally (pre-conditions, post-conditions and invariants). ● Debugging for implementation error detection and correction e.g. using ipdb. ● Timing algorithms. This could be done using Jupyter magic functions, for example (%timeit) or manually programmed.

Assessment

To achieve a 'pass' for this unit, learners must provide evidence to demonstrate that they have fulfilled all the learning outcomes and meet the standards specified by all assessment criteria.

Learning Outcomes to be met	Assessment Criteria to be covered	Assessment type	Word count (approx. length)
LO1-LO2	All AC under LO1-LO2	Report	3500 words
LO3-LO4	All AC under LO3-LO4	Annotated Pseudocode and Report	1000 words

Indicative Reading List

Cormen T. H. & Leiserson C. E. (2022) *Introduction to Algorithms (4th edition)* ISBN 978-0262046305

Sedgewick R. & Wayne K. (2011) *Algorithms (4th edition)* ISBN 978-0321573513

Skiena S. S. (2021) *The Algorithm Design Manual (3rd edition)* ISBN 978-3030542580

Sipser M. (2012) Introduction to the Theory of Computation (3rd edition) ISBN 978-1133187790

Matthes E. (2023) *Python Crash Course: A Hands-On, Project-Based Introduction to Programming (3rd edition)* ISBN 978-1718502703

Cormen T. H. (2013) *Algorithms Unlocked* ISBN 978-0262518802

Homer S. & Selman A. L. (2011) *Computability and Complexity Theory (2nd edition)* ISBN 978-1461406815

Additional Resources

Python 3 documentation: https://docs.python.org/3/

Leetcode (programming challenges): https://leetcode.com/

COMPUTER NETWORKS

Unit Reference Number	K/651/4221
Unit Title	Computer Networks
Unit Level	4
Number of Credits	200
Total Qualification Time (TQT)	200 Hours
Guided Learning Hours (GLH)	100 Hours
Mandatory / Optional	Mandatory
Sector Subject Area (SSA)	06.1 Digital Technology (practitioners)
Unit Grading Type	Pass / Fail

Unit Aims

This unit introduces the principles of computer networking. Learners are introduced to network models and the different layers of computer networks. They will learn about the TCP/IP model and protocol suite, understanding the need for the separation between the different layers. A range of protocols are introduced, with a particular focus given to the key protocols of ethernet, IP and TCP/UDP. Learners will learn about the networking hardware that is essential for connecting devices in the physical world, including routers, switches. The unit concludes with the learners putting theory into practice by designing a practical computer network for a given setting.

Learning Outcomes, Assessment Criteria and Indicative Content

Learning Outcome – The learner will:	Assessment Criteria – The learner can:	Indicative Content
1. Understand the core principles of computer networks.	1.1 Describe the OSI and TCP/IP networking models and their layers and compare them. 1.2 Explain how different TCP/IP network protocols enable communication across devices and networks. 1.3 Analyse protocols and protocol suites.	**Network Models** ● Application Layer (Application Layer, Presentation Layer, Session Layer), Transport Layer (Transport Layer), Networking/Internet Layer (Network Layer), Link/Physical Layer (Data Link Layer, Physical Layer). *Note: Bracketed items are OSI and unbracketed are TCP/IP model layers.* ● The TCP/IP model as a practical

		implementation of the idealised OSI networking model. 4 layers vs. 7 layers. **Protocols** ● Definition: a set of rules that specify the procedure for communication between systems. ● Historical protocols: IPX and SPX, leading to the modern protocol standard: TCP/IP. ● The advantages of layered protocol stacks from the perspective of the modularity of the layers. ● Protocol suite definition and the TCP/IP (internet) protocol suite – named after the TCP and IP protocols it contains. ● Protocols at different levels and different application layer protocols (e.g. SMTP, POP3, and Imap4 for email; HTTP for transfer of information on the internet). ● TCP/UDP – connection-oriented vs. connectionless transmission protocols at the transport layer. ● IP networking layer protocol. ● Ethernet, ARP and MAC addresses at the datalink/physical layer.
2. Understand how devices connect to form networks.	2.1 Describe the IP protocol. 2.2 Explain the role of network hardware components such as routers and switches. 2.3 Analyse the security measures used to secure communication across computer networks. 2.4 Evaluate mobile and wireless connection protocols and technologies.	**Internet Protocol (IP)** ● IP and ICMP protocols. ● IP address formats and subnets. ● IPv4 and IPv6 – larger address space advantages of IPv6; comparison of the versions. **Networking Hardware** ● Routers (networking layer devices). ● Network switches (datalink layer devices). ● Gateways for protocol conversion to connect different networks.

		● Network infrastructure: fibre optic and copper cables; transmission speeds. **Security Measures** ● End-to-end connections. ● Firewalls. ● Encryption. **Wireless and Mobile Networks** ● Mobile connectivity technologies: GSM, UMTS, LTE, NR enabling 2G, 3G, 4G, 5G. ● 6G and emerging mobile connectivity infrastructure. ● Wireless Local Area Network (WLAN) and Wi-Fi (IEEE 802.11 standards).
3. Be able to design a computer network suited to a given setting.	3.1 Decide on a network topology that is best suited to a given application and setting. 3.2 Analyse different network topologies and their role in disparate communication settings. 3.3 Design appropriate addressing schemes for a particular network. 3.4 Evaluate the best choice of network hardware for a given network.	**Network Topologies** ● Bus topology: connection to a central cable. ● Star topology: indirect connection through switches. ● Other topologies: e.g. ring, mesh. ● Use cases: small networks vs. large networks; redundancy to node failure; performance; expense in setting up. **Designing a Computer Network** ● Choosing IP addresses for devices in the network. Separation of the network structure using subnetting. Static/dynamic, public/private addressing schemes. ● Choosing appropriate hardware for the network (e.g. routers, switches, access points, cable connections) taking into account design requirements (e.g. cost, network scale).

Assessment

To achieve a 'pass' for this unit, learners must provide evidence to demonstrate that they have fulfilled all the learning outcomes and meet the standards specified by all assessment criteria.

Learning Outcomes to be met	Assessment Criteria to be covered	Assessment type	Word count (approx. length)
LO1-LO3	All AC under LO1 to LO3	Report	4500 words

Indicative Reading List

Kurose J., Ross K. (2021) *Computer Networking: A Top-Down Approach (8th edition)* ISBN 978-1292405469

Fourouzan B. A. (2021) *Data Communications and Networking with TCP/IP Protocol Suite (6th edition)* ISBN 978-0078022098

Tannenbaum A., Wetherall D. (2010) *Computer Networks (5th edition)* ISBN 978-0132126953

Leon-Garcia A., Widjaja I. (2003) *Communication Networks* ISBN 978-0072463521

Fall K., Stevens W. (2011) *TCP/IP Illustrated: The Protocols, Volume 1 (2nd edition)* ISBN 978-0321336316

Wright G. R. & Stevens W. R. (1995) *TCP/IP Illustrated, Volume 2: The Implementation* ISBN 978-0201633542

Stevens W. R. (1996) *TCP/IP Illustrated: v. 3: TCP for Transactions, HTTP, NNTP and the Unix Domain Protocols* ISBN 978-0201634952

Additional Resources

IEEE 802.11 Wireless Local Area Networks standards: https://ieee802.org/11/

MATHEMATICS FOR COMPUTER SCIENCE

Unit Reference Number	L/651/4222
Unit Title	Mathematics for Computer Science
Unit Level	4
Number of Credits	200
Total Qualification Time (TQT)	200 Hours
Guided Learning Hours (GLH)	100 Hours
Mandatory / Optional	Mandatory
Sector Subject Area (SSA)	06.1 Digital Technology (practitioners)
Unit Grading Type	Pass / Fail

Unit Aims

In this unit, learners will become familiar with key subjects in mathematics that are of use in the field of computer science. The unit begins with an introduction to binary numbers, which form the basis of the representation of information in computers. Learners will then be taught key principles in the fields of Boolean propositional logic and set theory, including Boolean operators, and will be given an introduction to proof techniques in mathematics. Number theory, linear algebra, and sequences and series are also covered, to equip learners with a mathematical toolset that will enable them to reason about problems in computing.

Learning Outcomes, Assessment Criteria and Indicative Content

Learning Outcome – The learner will:	Assessment Criteria – The learner can:	Indicative Content
1. Understand binary and the representation of numbers in different bases.	1.1 Describe what is meant by binary representation. 1.2 Explain how to change between bases. 1.3 Explain how numbers are represented in binary. 1.4 Evaluate the use of binary representation in	**Bases and representation** ● Base 16, base 10, base 2. ● Historical reasons for bases and their representation. **Binary in computer science** ● Resilience of binary values to noise and error in storage media. ● Understand the representation of numbers in

	the field of computer science.	binary. Floating point representations including the mantissa and exponent. Two's complement. ● Connections to Boolean logic and Boolean set operations.
2. Understand the principles of propositional logic, set theory, and proof as applicable to computer science.]	2.1 Describe a range of mathematical proof techniques. 2.2 Describe the possible algebraic operations in set theory. 2.3 Explain the key principles of (Boolean) propositional logic. 2.4 Use truth tables to detail the possible outcomes of binary logical operators. 2.5 Analyse the similarities between Boolean logical operators and Boolean algebra in set theory. 2.6 Evaluate mathematical proof techniques and the circumstances under which they may be appropriate.	**Proof Techniques** ● Mathematical induction, strong induction. ● Proof by contradiction, argument by cases, and counterexamples, proving existential and universal statements. ● Other proof techniques e.g. pigeonhole principle. **Propositional Logic** ● Propositions and Boolean logical operators (disjunction, conjunction, implication, negation). ● Truth tables. ● Logical equivalence and laws: useful logical equivalences and laws e.g. commutative laws, identity laws. ● Normal forms: literals and clauses, conjunctive normal forms, conversion to conjunctive normal form. **Set Theory** ● Introduction to sets and visualisation of set theory concepts using Venn diagrams (e.g. set membership, partitions, union). ● Set theory notation and Boolean algebra i.e. intersection, union, complement. Boolean identities. ● Common sets: empty set, real numbers, natural numbers, integers etc.

3. Understand key topics in number theory and their application in computer science.	3.1 Describe the key principles of modular arithmetic and finite fields. 3.2 Explain a range of key number theoretic theorems and proofs. 3.3 Analyse the differences between ordinary and modular arithmetic. 3.4 Evaluate the use of modular arithmetic and prime numbers in security applications such as cryptography and hashing.	**Finite Fields and Modular Arithmetic** ● Algebras: sets and accompanying properties. ● Introduction to finite fields. ● Multiplicative and additive identities (0 and 1) and inverses (division and subtraction). ● Greatest Common Divisor (GCD) and Lowest Common Multiple (LCM), Euclid's algorithm. ● Fermat's little theorem, Euler's theorem, and other key theorems and proofs and consequences. **Prime Numbers** ● Modular primes. ● Prime factorisation: asymmetric nature of factorisation and verification. ● Applications in computer science: cryptography and hashing using prime numbers over finite fields.
4. Understand series and sequences and their importance in computer science.	4.1 Describe what is meant by a sequence and a series. 4.2 Explain the difference between arithmetic and geometric series and sequences. 4.3 Use appropriate notation to represent sequences	**Sequences and Series** ● Sequences and types of sequence: arithmetic, geometric, and harmonic. Progressions. ● Series: geometric, power, harmonic, alternating etc. ● Summation and multiplicative notation for series.
5. Understand key topics in linear algebra.	5.1 Describe what is meant by vectors and matrices. 5.2 Use row-reduction to convert a matrix to echelon form. 5.3 Solve linear systems of equations using matrices.	**Vectors and Matrices** ● Basic principles of vectors and matrices. ● Linear dependence and independence. ● Matrix multiplication and matrix inverses. **Solving Linear Problems** ● Row reduction and echelon form. Solution of linear systems of equations.

	5.4 Analyse the significance of the determinant of a matrix. 5.5 Use a range of techniques to decompose matrices and evaluate their usefulness.	● Linear least squares. **Matrix Types and Properties** ● Symmetric matrices; upper/lower triangular matrices; positive/positive definite matrices, and their properties. ● Trace of a matrix. ● Determinant and its significance. **Matrix Decomposition and Factorisation** ● Lower upper (LU) decomposition. ● Singular Value Decomposition (SVD), diagonalisation, and eigenvalue decomposition (eigenvectors and eigenvalues). ● Principal component analysis (PCA) for datasets represented as matrices.

Assessment

To achieve a 'pass' for this unit, learners must provide evidence to demonstrate that they have fulfilled all the learning outcomes and meet the standards specified by all assessment criteria.

Learning Outcomes to be met	Assessment Criteria to be covered	Assessment type	Word count (approx. length)
LO1-LO3	All ACs under LO1 to LO3	Report	3000 words
LO4-LO5	All ACs under LO4 to LO5	Report	1500 words

Indicative Reading List

Graham R. L., Knuth D. E., Patashnik O. (1994) Concrete Mathematics: A Foundation for Computer Science (2nd edition) ISBN 978-0201558029

Strang G. (2016) *Introduction to Linear Algebra (5th edition)* ISBN 978-0980232776

Levin O. (2018) *Discrete Mathematics: An Open Introduction* ISBN 978-1792901690

Rosen K. H. (2018) *Discrete Mathematics and Its Applications (8th edition)* ISBN 978-1260091991

Epp S. S. (2019) *Discrete Mathematics with Applications (5th edition)* ISBN 978-1337694193

Das A. (2013) *Computational Number Theory* ISBN 978-1439866153

Cunningham D. W. (2016) *Set Theory: A First Course* ISBN 978-1107120327

Stoll R. R. (2003) *Set Theory and Logic* ISBN 978-0486638294

Cummings J. (2021) *Proofs: A Long-Form Mathematics Textbook* ISBN 979-8595265973

Polya G. (1990) *How to Solve It: A New Aspect of Mathematical Method (2nd edition)* ISBN 978-0140124996

Additional Resources

Project Euler (computer programming/mathematics practice problems): https://projecteuler.net/

OPERATING SYSTEMS

Unit Reference Number	M/651/4223
Unit Title	Operating Systems
Unit Level	4
Number of Credits	200
Total Qualification Time (TQT)	200 Hours
Guided Learning Hours (GLH)	100 Hours
Mandatory / Optional	Mandatory
Sector Subject Area (SSA)	06.1 Digital Technology (practitioners)
Unit Grading Type	Pass / Fail

Unit Aims

In this unit, learners will explore the role and principles of operating systems. Operating systems are introduced as software to manage resources in a computer and the need for operating systems is explained. The principal functions of operating systems are introduced, including memory management and process management, and learners will gain an appreciation for the trade-offs that exist in allocating hardware resources to processes. Learners will conclude by leveraging the understanding acquired through the unit to write their own operating system code at the kernel level.

Learning Outcomes, Assessment Criteria and Indicative Content

Learning Outcome – The learner will:	Assessment Criteria – The learner can:	Indicative Content
1. Understand the essential function and architecture of modern operating systems.	1.1 Outline a range of common operating systems for computers and mobile devices. 1.2 Describe the historical development and need for operating systems. 1.3 Explain the key components of an operating system.	**History of Operating System (OS) Development** ● Transition from queues of individuals using mainframes to greater management and tracking of hardware resources, finally kernels as the origin of modern operating systems. ● Leading up to the definition of an operating system as a software system that manages the execution of software (programs), monitoring, and the management and allocation of

	1.4 Explain how operating systems manage hardware and software resources. 1.5 Analyse the different kinds of operating system that exist. 1.6 Evaluate common trade-offs in resource management for operating systems.	hardware resources. ● Shifting standards and expectations beyond core functions of operating systems: graphical user interfaces, basic packaged software (text editors) etc. **Operating System Functions** ● Resource management of CPU, I/O devices, file systems, memory. ● Hardware abstraction (management). Interaction of user-level software and hardware via the kernel through system calls. ● User and process (software) management (monitoring and allocation of time share and hardware resources). **OS Architectures** ● Monolithic: user programs initiate system calls through interfaces which are executed in kernel space (memory management, communication, process scheduling, file system operations, I/O management, network management). Linux as an example. ● Layered: hierarchical abstraction involving separation of user programs, system services, file system management, I/O buffer, memory management, process management, CPU scheduling. Windows NT as an example. ● Microkernel: similar to monolithic, but only interprocess communication, memory management and synchronisation are executed inside the kernel space. *N.B. These operations are covered in detail later in the Unit.* ● Hybrid architectures. ● Advantages/disadvantages of each architecture:

		complexity, execution speed, modularity/resilience and fault isolation etc. ● Trade-offs between efficiency and security/resilience.
2. Understand memory and I/O management techniques used by modern operating systems.	2.1 Compare different file system types. 2.2 Describe different approaches to memory management. 2.3 Describe the role of device drivers in facilitating communication with hardware devices. 2.4 Explain the role of virtual (swap) memory. 2.5 Explain the role of paging in allowing for non-contiguous memory allocation. 2.6 Analyse the difference between fixed and dynamic loading and linking strategies.	**Memory Management** ● Logical (virtual) vs. physical address space and dynamic relocation. ● Memory allocation strategies: principle of contiguous blocks, fragmentation (internal/external), paging/page tables/memory management unit, fitting strategies. ● Static and dynamic loading and linking. ● Swapping and virtual memory. **Drivers** ● Software that enables communication between the operating system and hardware. ● Examination of the device drivers present on personal computers. ● Types of drivers: user-mode drivers, kernel drivers, virtual drivers. **File Systems** ● Common types: FAT32, NTFS, ext4 etc. ● Key features: Journalling, file permissions and locks, data structures e.g. inodes. ● File operations: reading, writing, and managing. Caching strategies.
3. Understand process management techniques used by modern operating systems.	3.1 Describe what is meant by a process. 3.2 Explain the contents of the process control block and its importance.	**Processes and Threads** ● Process: A job to be executed – may be a user program or a system process. ● Threads: lightweight processes that share the same memory. User-level threads and kernel-level threads.

	3.3 Explain the way that processes are represented for management by operating systems. 3.3 Analyse the range of responsibilities an operating system has for managing the execution of processes.	**Process Representation and Execution** ● Process states (ready, wait, running, terminated). ● Process attributes that uniquely determine a process: process ID, state, CPU registers, I/O status, page tables, accounting information etc. ● Process control block: data structure that contains process attributes, used after system calls or interrupts to continue the execution of a suspended process. **Fundamental Tenets of Process Management** ● OS process operations (i.e. creation, scheduling, execution, killing). ● Context switching and mode switching. ● Process scheduling. Scheduling algorithms: first-come-first-serve, round robin, shortest job first, priority-based scheduling, multilevel queue scheduling etc. ● Concurrency and Inter Process Communication (IPC). ● Hardware and software interrupt handling.
4. Understand the principles of and potential problems with concurrency in operating systems.	4.1 Outline the difference between independent and cooperating processes. 4.2 Describe what is meant by concurrency. 4.3 Explain the memory sharing and message passing approaches to inter process communication. 4.4 Analyse deadlock and race conditions as potential problems in concurrent processing settings and propose strategies to mitigate them.	**Concurrency** ● Concurrency refers to the handling of processes which happen at the same time by the OS. ● Benefits and drawbacks of currency. **Inter Process Communication (IPC)** ● Low-level and high-level IPC. ● Shared memory approach. ● Message passing approach – direct/indirect, synchronous/asynchronous. ● Advantages and disadvantages of IPC in terms of efficiency, complexity, and security risks.

	4.5 Evaluate the importance of low-level and high-level inter process communication.	**Potential Pitfalls of Concurrent Processing** ● Deadlock: where two processes are both waiting for one another to release resources. ● Software race conditions. ● Detection of deadlock and race conditions. ● OS strategies to avoid potential problems e.g. semaphores. Deadlock avoidance using Banker's algorithm.
5. Be able to write programs at the kernel level.	5.1 Develop kernel modules that interact with hardware devices and system resources. 5.2 Implement system calls to extend kernel functionality and interact with user-level applications. 5.3 Implement kernel-level programs to perform different tasks. 5.4 Use debugging to test kernel-level code to ensure stability and performance.	Kernel Modules ● Introduction to LKMs and how they extend the functionality of the running base kernel. ● Hardware interaction: how kernel modules interface with hardware (e.g. block and character devices). System Calls ● Understanding the importance of system calls as an interface between the kernel-level and the user-level. ● Writing and using system calls for tasks such as file manipulation, process control, and inter process communication. Programming Kernel Modules ● Writing, loading, and unloading kernels in Linux. ● Device driver programming and interaction with hardware components. ● Simple example of a kernel module to print "Hello World". ● Extended example of kernel module programming for e.g. system calls, device drivers, or filesystem driver functions. ● Testing and debugging kernel modules.

Assessment

To achieve a 'pass' for this unit, learners must provide evidence to demonstrate that they have fulfilled all the learning outcomes and meet the standards specified by all assessment criteria.

Learning Outcomes to be met	Assessment Criteria to be covered	Assessment type	Word count (approx. length)
LO1-LO4	All ACs under LO1 to LO4	Report	2500 words
LO5	All ACs under LO5	Report	2000 words

Indicative Reading List

Silberschatz A., Gagne G., Galvin P. B. (2018) *Operating Systems Concepts (10th edition)* ISBN 978-1119439257

Tanenbaum A., Bos H. (2023) *Modern Operating Systems (5th edition)* ISBN 978-1292459660

Love R. (2010) *Linux Kernel Development* ISBN 978-8184958522

Patterson, D. A. & Hennessy, J. L. (2020) *Computer Organization and Design: The Hardware/Software Interface (6th edition)* ISBN 978-0128201091

McKusick M., Neville-Neil G., Watson R. (2014) *The Design and Implementation of the FreeBSD Operating System (2nd edition)* ISBN 978-0321968975

Additional Resources

Blundell N. *Writing a Simple Operating System from Scratch*: https://www.cs.bham.ac.uk/~exr/lectures/opsys/10_11/lectures/os-dev.pdf

LEVEL 5 UNIT SPECIFICATIONS

SECURITY TESTING

Unit Reference Number	R/651/4224
Unit Title	Security Testing
Unit Level	5
Number of Credits	200
Total Qualification Time (TQT)	200 Hours
Guided Learning Hours (GLH)	100 Hours
Mandatory / Optional	Mandatory
Sector Subject Area (SSA)	06.1 Digital Technology (practitioners)
Unit Grading Type	Pass / Fail

Unit Aims

This unit provides learners with an understanding of security testing methodologies, and of the tools used to identify and exploit vulnerabilities in web applications and mobile applications. Learners will acquire practical skills in vulnerability assessment of exemplar web and mobile applications using industry-standard security testing tools and will gain an introduction to reporting on identified vulnerabilities. Learners are also introduced to ethical and professional considerations for security testing.

Learning Outcomes, Assessment Criteria and Indicative Content

Learning Outcome – The learner will:	Assessment Criteria – The learner can:	Indicative Content
1. Understand a range of security testing methods.	1.1 Describe the role of security testing in protecting cyber systems. 1.2 Explain the different approaches to security testing. 1.3 Analyse the distinct roles that different security testing methodologies have in improving the security of cyber systems, and the advantages and disadvantages of different	**Need for Security Testing** ● Definition and benefits of security testing for organisations and Cyber Security systems. ● Understanding the role of ethical hacking in identifying vulnerabilities; ethical concerns; and responsible disclosure. **Security Testing Approaches** ● Static Application Security Testing (SAST), Code Review.

	approaches. 1.4 Analyse the role of ethical hacking within the broader field of cyber security testing.	● Dynamic Application Security Testing (DAST) (a.k.a. vulnerability scanning/assessment). ● Interactive Application Security Testing (IAST). ● Penetration testing (ethical hacking), types of penetration tests (black-box, white-box, grey-box). Includes vulnerability scanning and assessment. ● Adversarial simulation (red teaming and blue teaming). ● Human and organisational vulnerability testing. ● Threat and Risk-Based Modelling. Threat modelling methods (e.g. STRIDE, DREAD). ● Security Audits and Compliance. Reviewing system logs, configurations and procedures. Compliance audits for e.g. GDPR, HIPAA, PCI-DSS. *N.B. Penetration testing, as a specific form of security testing, is covered in detail in the Ethical Hacking unit.*
2. Be able to perform essential web application security testing.	2.1 Describe the key features of web applications, from a security perspective. 2.2 Explain common web application vulnerabilities and attack techniques. 2.3 Analyse web browser vulnerabilities and their impact on security. 2.4 Use web security testing tools to identify vulnerabilities in web applications. 2.5 Evaluate the effectiveness of security tools in identifying, preventing, and mitigating web-based threats.	**Web Application Security** ● “What, why, when, where, and how” of testing web applications. ● Key features of web applications from a security perspective. ● Understanding of common threats e.g. Cross-Site Scripting (XSS), Clickjacking, cookie theft, and phishing. **Web Application Security Risks** ● Exploration of common vulnerabilities as outlined by OWASP Top Ten: Injection Attacks (SQL, XML, etc.), Cross-Site Scripting (XSS), Insecure Design, Broken Access Control, Security Misconfiguration etc. (Updated every

		four years). ● Overview of browser-based threats, including vulnerabilities related to plug-ins, cookies, session management, and insecure communication. ● Understanding of other relevant attacks, such as Cross-Site Request Forgery (CSRF), Server-Side Request Forgery (SSRF), and vulnerabilities related to authentication/authorisation. **Web Security Tools** ● Practical use of testing tools, such as Burp Suite, OWASP Zed Attack Proxy (ZAP), and other vulnerability scanners. ● Techniques for packet interception, man-in-the-middle testing, and session management analysis. ● Analysis of the strengths and limitations of tools in detecting vulnerabilities and enhancing security. ● Comparison of automated and manual security testing methods.
3. Be able to perform essential mobile application security testing.	3.1 Describe the defining features of mobile applications, from a security perspective. 3.2 Explain common security vulnerabilities and attack vectors for mobile applications. 3.3 Use mobile application security testing tools to uncover and address vulnerabilities. 3.4 Analyse the security of mobile applications based on vulnerability assessments and propose solutions.	**Mobile Ecosystem Overview** ● The unique architecture of mobile applications, including client-server interactions and local device storage. ● Mobile Platforms: Differences between iOS and Android security models (e.g., sandboxing, permissions management). ● App Stores: code signing, app vetting, and distribution mechanisms. ● Permissions Model: How apps request access to sensitive data (location, camera, contacts, etc.), and the risks of granting unnecessary

		permissions. ● Authentication: Methods such as biometric authentication (fingerprint, facial recognition), two-factor authentication (2FA), and OAuth. ● Data Storage and Encryption: How sensitive data is stored (e.g., internal storage, external SD cards) and protected (e.g., device encryption, key management). **Mobile Vulnerabilities** ● OWASP Mobile Top 10: Improper Credential Usage, Insecure Communication, Insecure Data Storage, Insufficient Cryptography etc. ● Android vs. iOS security: Jailbreaking/rooting risks, sandboxing, update mechanisms. **Mobile Application Security Testing** ● Tools and frameworks: e.g. OWASP mobile security tools, Drozer for Android testing, jailbreak detection, network traffic monitoring with tools such as Wireshark. ● Static Application Analysis: OWASP Mobile Security Testing Guide, Apktool, iOS Binary Analysis tools. ● Hands-on testing with live mobile applications to scan for key vulnerabilities, such as permissions misuse, weak encryption, or insecure external storage. **Security Analysis and Solutions** ● Use of scoring systems (e.g. Common Vulnerability Scoring System (CVSS)) to evaluate threats. ● Remediation strategies: applying security patches, enforcing secure coding practices,

		performing end-to-end encryption for sensitive data. ● Best practices for mobile application security, e.g. least privilege access for app permissions, improvements to session management (session timeouts, token-based authentication).
4. Understand the ethical and professional standards in security testing	4.1 Describe a range of existing professional codes of ethics. 4.2 Explain the importance of professional ethics and standards in penetration testing. 4.3 Analyse the legal considerations in conducting security tests.	**Ethics of Security Testing** ● Definition of professional ethics: understanding of the responsibilities of security testers. ● Confidentiality and integrity: the importance of safeguarding sensitive information, and potential impacts on clients and stakeholders. **Professional Standards** ● Codes of ethics of professional bodies e.g. EC-Council, ISACA ● Principles of responsible disclosure. **Legal Considerations and Compliance** ● Relevant data protection laws: e.g. GDPR, HIPAA, PCI-DSS. ● Permissions, extent and consent: rules of engagement and scope agreements, non-disclosure agreements (NDAs). *N.B. Further detailed examination of ethical issues and penetration testing specific ethics is conducted in the Ethical Hacking unit.*

Assessment

To achieve a ‘pass’ for this unit, learners must provide evidence to demonstrate that they have fulfilled all the learning outcomes and meet the standards specified by all assessment criteria.

Learning Outcomes to be met	Assessment Criteria to be covered	Assessment type	Word count (approx. length)
LO2	All ACs under LO2	Report	1500 words
LO3	All ACs under LO3	Report	1500 words
LO1 and LO4	All ACs under LO1 and LO4	Report	1500 words

Indicative Reading List

Seirsen R. (2022) *The Metrics Manifesto: Confronting Security with Data* ISBN 978-1119515364

Stoll C. (2005) *Cuckoo's Egg: Tracking a Spy Through the Maze of Computer Espionage* ISBN 978-1416507789

Stuttard D. & Pinto M. (2011) *The Web Application Hacker's Handbook: Finding and Exploiting Security Flaws (2nd edition)* ISBN 978-1118026472

Ball C. J. (2022) *Hacking APIs: Breaking Web Application Programming Interfaces* ISBN 978-1718502444

Finney G. (2022) *Project Zero Trust: A Story about a Strategy for Aligning Security and the Business* ISBN 978-1119884842

OWASP Top Ten Web Application Security Risks: https://owasp.org/www-project-top-ten/

OWASP Top Ten Mobile Risks: https://owasp.org/www-project-mobile-top-10/

OWASP Web Security Testing Guide: https://owasp.org/www-project-web-security-testing-guide/

OWASP Mobile Application Security: https://owasp.org/www-project-mobile-app-security/

Additional Resources

EC-Council - Understand the Five Phases of the Penetration Testing Process:
https://www.eccouncil.org/cybersecurity-exchange/penetration-testing/penetration-testing-phases/

CVSS SIG : https://www.first.org/cvss/

EC-Council Code of Ethics: https://www.eccouncil.org/code-of-ethics/

ISACA Code of Ethics: https://www.isaca.org/code-of-professional-ethics

ARTIFICIAL INTELLIGENCE

Unit Reference Number	T/651/4225
Unit Title	Artificial Intelligence
Unit Level	5
Number of Credits	200
Total Qualification Time (TQT)	200 Hours
Guided Learning Hours (GLH)	100 Hours
Mandatory / Optional	Mandatory
Sector Subject Area (SSA)	06.1 Digital Technology (practitioners)
Unit Grading Type	Pass / Fail

Unit Aims

In this module, learners will become acquainted with artificial intelligence, with a focus on the modern topics of machine learning and deep learning. The historical developments motivating the development of artificial intelligence will be discussed, and neural networks will be understood as being one subfield of artificial intelligence, albeit a very relevant one. A range of deep learning architectures and training strategies will be introduced, and learners will gain first-hand experience designing, building and training their own neural networks using Python and popular deep learning libraries. Finally, the role of artificial intelligence in the realm of cyber security is discussed.

Learning Outcomes, Assessment Criteria and Indicative Content

Learning Outcome – The learner will:	Assessment Criteria – The learner can:	Indicative Content
1. Understand the principles of artificial intelligence and data-driven learning.	1.1 Define artificial intelligence and describe the different approaches to learning. 1.2 Describe a range of simple and complex machine learning strategies. 1.3 Explain the difference between supervised, unsupervised, and reinforcement learning.	**Artificial Intelligence** ● Historic developments of artificial intelligence: Babbage's difference engine, computers, the Turing test. ● Symbolic AI: based on formal systems e.g. propositional logic. ● Machine learning: learning automatically and without programming, directly from data using statistical principles.

	1.4 Analyse the significance of the bias-variance trade-off in machine learning.	**Machine Learning** ● Most popular AI technique currently – deep learning is a subset. ● Supervised learning: learning to associate inputs to outputs (e.g. supervised classification). ● Unsupervised learning: learning to transform data to a form which is beneficial for some later purpose (e.g. clustering). ● Reinforcement learning: learning based on episodes and positive rewards/negative punishment. ● Bias-variance trade-off: the lower the bias on the learning algorithm, the higher the variance of the training result. ● Machine learning approaches: linear regression, logistic regression, support vector machines, kernel methods, random forest, ensemble methods, neural networks etc. The goal should be to emphasise that neural networks (and subsequently deep neural networks) are just one method.
2. Understand the principles underlying neural network training.	2.1 Describe how backpropagation is used to update neural network parameters to achieve a desired goal. 2.2 Explain the bias-variance trade-off in machine 2.3 Explain the importance of loss functions in providing an objective for the training of neural networks.	**Neural Network Preliminaries** ● Biological inspiration and idealised modelling. ● Universal function approximation. ● Layer structure. ● Activation functions and their purposes: ReLU (simple, yet effective), PReLU, Sigmoid (for output of e.g. probabilities), Tanh etc. ● Equivalence to/reliance upon matrix-vector multiplication; efficient evaluation using Graphics Processing Units (GPUs). **Training Neural Networks**

	2.4 Analyse and compare common optimisation strategies for neural network training. 2.5 Evaluate the advantages and disadvantages of deep and shallow neural network architectures.	● Stochastic gradient descent (SGD), learning rate and its impact, and more advanced optimisation strategies (Adam, RMSprop etc.). Advantages over full gradient descent: computational efficiency, stochasticity aids with avoiding local minima. ● Backpropagation: learners might derive updates to dense neural network parameters by hand to gain an understanding, for example. ● Techniques for improving training performance: dropout, learning rate schedulers etc. ● Loss functions: (binary) cross-entropy, mean squared error etc. **Deep Neural Networks** ● The benefits of larger numbers of layers. ● Intuitive understanding of purpose of layers (abstraction of features e.g. visualising the feature extraction process for convolutional neural networks). ● Trade-off: increased complexity (time to learn) vs. increased learning power.
3. Be able to build and train neural networks.	3.1 Describe a range of neural network layer architectures and their use cases. 3.2 Use popular neural network libraries to implement neural network architectures. 3.3 Implement training strategies to fit model parameters. 3.4 Evaluate the performance of trained neural network models with reference to the training-test gap.	**Neural Network Architectures** ● Fully connected/dense layers: general purpose. ● Transformers: general purpose, natural language processing, translation. ● Convolutional layers: spatial/temporal data where there exist local correlations in space/time. ● Graph neural networks: generalisation of convolutional layers. ● Other network architectures: ResNet, U-Net. ● Comparison of architectures in terms of computational power, computational complexity

		and generalisation capabilities. **Building Neural Networks** ● Introduction to a popular Python neural network library i.e. Pytorch or Tensorflow (including Keras). ● Programming and debugging a simple (e.g. fully connected) neural network architecture. **Training Neural Networks** ● Commonly used datasets e.g. MNIST, ImageNet, and CIFAR-10 for images. ● Training the neural network on a training dataset, testing the trained network on a test set; comparison of train and test performance. ● Overfitting and underfitting. ● Proposal and testing of strategies for improving the architecture and/or training of the network e.g. adding layers, changing the learning rate, changing the activation functions.
4. Understand a range of generative modelling architectures and techniques in deep learning.	4.1 Describe a range of generative model architectures. 4.2 Compare deep generative modelling approaches along several dimensions. 4.3 Explain the method of contrastive language-image pretraining and its importance in learning mappings to meaningful representations for images and text. 4.4 Analyse the principal training methods of large language models.	**Deep Generative Modelling Approaches** ● Variational Autoencoders. ● Normalising Flows. ● Generative Adversarial Networks (GANs): generator and discriminator architecture; counterfeit money analogy. ● Diffusion models. ● Advantages/disadvantages and comparison of different approaches in terms of computational complexity, performance, deterministic generation/stochasticity. **Image and Video Generation** ● Contrastive learning and CLIP (Contrastive

	4.5 Evaluate the importance of the transformer architecture in the success of large language models.	Language-Image Pretraining) for learning semantic representations of visual information, using corresponding human text descriptions and labels. ● Importance and success of diffusion models in this domain. ● Popular examples of image and video generation models e.g. DALLE-2. **Large Language Models (LLMs)** ● Principal training methods: e.g. tokenisation, partial masking and prediction of missing tokens, reinforcement learning from human feedback (RLHF). ● Transformer architecture and its importance in the success of large language models. ● Popular proprietary and open architectures e.g. ChatGPT, Llama 3.
5. Understand how AI can be used for both offence and defence in the realm of Cyber Security.	5.1 Identify scenarios in which artificial intelligence can be applied as a preventative security measure. 5.2 Explain the ways in which AI can be used to enhance existing cyber-attacks and provide new opportunities for attackers. 5.3 Evaluate the benefits provided and risks posed by AI in the field of security.	**Applications of AI in Cyber Security defence** ● Scanning network traffic. ● Email filtering. **Cyber Threats Posed by AI** ● Data poisoning (deliberate introduction of information to training data for malicious purposes). ● Deepfakes (use of the likeness (e.g. voice, appearance) of an individual in generated settings). ● AI generated social engineering campaigns (using large language models) ● Adversarial attacks (perturbations which are introduced to deliberately modify algorithm functionality e.g. mislead classification algorithms).

		• Dark AI (use of AI for exploiting security vulnerabilities).

Assessment

To achieve a 'pass' for this unit, learners must provide evidence to demonstrate that they have fulfilled all the learning outcomes and meet the standards specified by all assessment criteria.

Learning Outcomes to be met	Assessment Criteria to be covered	Assessment type	Word count (approx. length)
LO1-LO3	All ACs under LO1-LO3	Report	2000 words
LO4-LO5	All ACs under LO4-LO5	Report	2500 words

Indicative Reading List

Goodfellow I., Bengio Y., Courville A. (2016) *Deep Learning* ISBN 978-0262035613

Bishop C. M. (2006) *Pattern Recognition and Machine Learning* ISBN 978-0387310732

Russel S. & Norvig P. (2021) *Artificial Intelligence: A Modern Approach (4th edition)* ISBN 978-1292401133

Murphy K. P. (2012) *Machine Learning – A Probabilistic Perspective* ISBN 978-0262018029

Chollet F. (2022) *Deep Learning with Python* ISBN 978-1617296864

Kingma D. P. & Welling M. (2013) *Auto-Encoding Variational Bayes* arXiv:1312.6114

Goodfellow I. et al. (2014) *Generative Adversarial Nets* Advances in Neural Information Processing Systems 27

Turing A. (1950) *Computing Machinery and Intelligence* Mind 49: 433-460

Vaswani A. et al. (2017) *Attention is all you need* Proceedings of the 31st International Conference on Neural Information Processing Systems

Additional Resources

AI powered cyberattacks:
https://www.crowdstrike.com/cybersecurity-101/cyberattacks/most-common-types-of-cyberattacks/#12.%20AI-Powered%20Attacks

Pytorch documentation: https://pytorch.org/docs/stable/index.html

Tensorflow documentation : https://www.tensorflow.org/api_docs

DATABASES

Unit Reference Number	Y/651/4226
Unit Title	Databases
Unit Level	5
Number of Credits	200
Total Qualification Time (TQT)	200 Hours
Guided Learning Hours (GLH)	100 Hours
Mandatory / Optional	Mandatory
Sector Subject Area (SSA)	06.1 Digital Technology (practitioners)
Unit Grading Type	Pass / Fail

Unit Aims

This unit introduces the key principles of databases. Learners will gain an understanding of the core concepts underlying database design, optimisation, and implementation. The different types of databases are introduced, with a focus on relational databases. Learners will be taught to use key commands in SQL to enable them to extract, modify, and create information in popular database management systems. Learners will put theory into practice through designing and building a relational database and integrating it with web applications.

Learning Outcomes, Assessment Criteria and Indicative Content

Learning Outcome – The learner will:	Assessment Criteria – The learner can:	Indicative Content
1. Understand the fundamental concepts of databases.	1.1 Compare several major database management systems. 1.2 Describe the key concepts of database systems 1.3 Explain the principle of queries and views in databases. 1.4 Explain the use of the SQL query language in	**Database Concepts** ● Database structures: tables (entities), attributes and relationships. ● Keys: primary keys, foreign keys, indexes. ● Database queries and views. **Relational Databases** ● Different types of database models: hierarchical, network, relational etc. ● Comparison of models based on different

	databases. 1.5 Critically evaluate the benefits of the relational database model.	attributes (reliability, maintainability, scalability etc.). ● Relational database model: separate tables with explicit relationships. **Database Management Systems** ● Definition and need for database management systems. ● Popular database management systems: MySQL, PostgreSQL, MongoDB and their attributes (e.g. relational vs. non-relational, query language). ● Comparison of database management systems based on strengths, weaknesses, use cases. **SQL** ● SQL introduction: role in querying databases. ● Basic SQL commands: SELECT, INSERT, UPDATE, DELETE, JOIN. ● Practice in using SQL to query and modify relational databases.
2. Be able to design a relational database based on user specifications.	2.1 Identify user requirements and needs for data for particular applications. 2.2 Create an entity-relationship diagram based on user requirements. 2.3 Explain the process of normalisation and its importance in database design. 2.4 Implement a relational database schema that satisfies user requirements.	**Entity-Relationship Diagrams** ● Identification of user requirements for the database. ● Creation of entity-relationship diagrams using a diagramming tool e.g. LucidChart. **Normalisation** ● Explanation of the process of normalisation using normal forms. ● Application of normalisation to database tables. ● Benefits: reductions to data redundancy, and improvements to data integrity. **Conversion to SQL**

		● Converting the entity-relationship diagram to SQL. ● Deployment in e.g. PostgreSQL for evaluation and testing.
3. Be able to use SQL to interact with databases.	3.1 Use SQL to formulate queries to extract data from databases. 3.2 Use SQL to formulate queries to manipulate and update data in databases. 3.3 Use PHP to access and update an SQL database for practical applications. 3.4 Explain how PHP can be used to interface with databases in server-side applications.	**SQL Queries** ● SQL queries for extracting information from a database. ● SQL queries for updating and manipulating records and tables. ● Efficient query writing. **PHP and SQL** ● Introduction to basic server-side application programming using PHP. ● Basic client-server communication using HTTP GET and POST requests. ● Creating connections to SQL databases in PHP. ● Running SQL code based on client-side website interactions. ● Illustrative programming of a basic login system using PHP and SQL. *Learners might first create an example database for LO2, and then subsequently build a web application which interfaced with it for LO3.*

Assessment

To achieve a ‘pass’ for this unit, learners must provide evidence to demonstrate that they have fulfilled all the learning outcomes and meet the standards specified by all assessment criteria.

Learning Outcomes to be met	Assessment Criteria to be covered	Assessment type	Word count (approx. length)
LO1-LO3	All ACs under LO1-LO2 and 3.1-3.2	Report and Diagram	3000 words

LO3	AC 3.3-3.4	Report	1500 words

Indicative Reading List

Elmasri R. (2015) *Fundamentals of Database Systems (7th edition)* ISBN 978-0133970777

Silberschatz A., Sudarshan S., Korth H. F. (2010) *Database System Concepts (6th edition)* ISBN 978-0073523323

Connolly T., Begg C. (2014) *Database Systems: A Practical Approach to Design, Implementation, and Management (6th edition)* ISBN 978-0132943260

Molinaro A., De Graaf R. (2020) *SQL Cookbook, 2E: Query Solutions and Techniques for All SQL Users (2nd edition)* ISBN 978-1492077442

Valade J. (2009) *PHP and MySQL For Dummies (4th edition)* ISBN 978-0470527580

Kimball R., Ross M. (2013) *The Data Warehouse Toolkit: The Definitive Guide to Dimensional Modeling (3rd edition)* ISBN 978-1118530801

Winand M. (2012) *SQL Performance Explained Everything Developers Need to Know about SQL Performance* ISBN 978-3950307825

Pascal F. (2000) *Practical Issues in Database Management* ISBN 978-0201485554

Additional Resources

LucidChart ER Diagram Tool: https://www.lucidchart.com/pages/examples/er-diagram-tool

W3Schools SQL Tutorial: https://www.w3schools.com/sql/

SQLBolt (Learn SQL with simple interactive exercises) https://sqlbolt.com/

DIGITAL FORENSICS

Unit Reference Number	A/651/4227
Unit Title	Digital Forensics
Unit Level	5
Number of Credits	200
Total Qualification Time (TQT)	200 Hours
Guided Learning Hours (GLH)	100 Hours
Mandatory / Optional	Mandatory
Sector Subject Area (SSA)	06.1 Digital Technology (practitioners)
Unit Grading Type	Pass / Fail

Unit Aims

This unit introduces the key principles of digital forensics. Different fields of data forensics are introduced, the types of digital evidence that can be acquired are detailed, and learners are introduced to the key stages of digital forensic investigations. Learners will gain practical experience in performing forensic analyses of computer systems, as well identifying the source and integrity of multimedia, using digital forensics software and tools. Legal and ethical considerations relating to chain of custody, privacy-security trade-offs, and the impact of artificial intelligence are discussed.

Learning Outcomes, Assessment Criteria and Indicative Content

Learning Outcome – The learner will:	Assessment Criteria – The learner can:	Indicative Content
1. Understand the principles of digital forensics	1.1 Describe the core concepts of digital forensics. 1.3 Explain the main stages of digital forensic investigations. 1.3 Explain the role of digital forensics in criminal and corporate investigations.	**Digital Forensics:** ● Crime scene to digital crime scene: a comparative evaluation of the tools and techniques used for forensics in each. ● Locard's exchange principle. ● Challenges: counter forensics, data hiding, steganography, steganalysis, diversity of data sources. ● Types of digital forensics: computer, mobile device, network, database, cloud, memory,

	1.4 Critically analyse the importance of maintaining a chain of custody.	forensic data. **Stages of Forensic Investigations** ● Identification ● Preservation ● Interpretation and Analysis ● Documentation ● Presentation **Chain of Custody** ● Emphasis on the importance of maintaining a chain of custody, to preserve the integrity of digital evidence, as part of a forensic investigation.
2. Understand the key techniques used in digital forensic investigations	2.1 Identify a range of best practices in digital forensics. 2.2 Explain the range of types of digital evidence. 2.3 Analyse the role of device fingerprints in matters of identity. 2.4 Evaluate the techniques most suitable to different forensic investigations.	**Best Practices in Digital Forensics** ● Contemporaneous note taking. ● ACPO Good Practice Guide for Digital Evidence. **Types and Sources of Digital Evidence** ● Sources of evidence: computers, mobile devices, hard drives, and more recent sources e.g. cars, drones, cloud services. ● Types of evidence: data (active, metadata, residual, volatile, replicant, archived), logs (OS, database, email etc.), multimedia. **Techniques used in Digital Forensics** ● Cross-drive analysis. ● Live analysis. ● Data carving and file recovery. ● Network traffic analysis. ● Keyword searching. ● Reverse steganography. ● Stochastic forensics.

		Device Fingerprints ● Types of digital fingerprints (e.g. browser, hardware, network). ● Capture, representation, and enhancement of device fingerprints. ● Identifying source devices based on fingerprints.
3. Be able to conduct forensic analysis of computer systems and storage media.	3.1 Describe best practices for collecting digital evidence. 3.2 Explain how the volatility of digital evidence impacts on evidence collection procedures. 3.3 Use forensic tools to forensically analyse digital data and recover hidden data.	**Digital Evidence Acquisition** ● Order of volatility: CPU cache and registers; Routing table, ARP cache, process table, kernel statistics; Memory (RAM) etc. ● Best practices for collecting evidence, taking into account order of volatility i.e. photographs of screen, live data capture e.g. RAM imaging, hard disk imaging, etc. **Forensic Tools and Techniques** ● Techniques for data extraction e.g. file carving. ● Tools such as Autopsy, FTK, EnCase. ● Hashing and encryption tools for verifying evidence integrity. ● Windows computer forensic techniques: registry analysis, metadata analysis, logs. ● Network forensics: traffic analysis.
4. Be able to use digital forensic techniques to verify the source and authenticity of image and video data.	4.1 Use source analysis to identify the source of image and video data. 4.2 Use content analysis to verify the authenticity and integrity of media. 4.2 Explain how digital watermarking can be used to protect intellectual property and for security.	**Multimedia Forensics** ● Source detection and device identification. ● Content analysis and integrity verification: statistical approaches, forgery detection, recovery of processing history. **Digital Watermarking**

	4.3 Critically analyse the challenges posed by generative artificial intelligence in verifying data authenticity. 4.4 Evaluate the challenges and risks associated with multimedia digital forensic investigations.	● Purposes of digital watermarking: copyright protection, content authentication, fingerprinting etc. ● Techniques for digital watermarking and requirements for watermarking frameworks (imperceptibility, tamper resistance etc.). ● Emerging media: AI watermarking. **Challenges and Risks** ● Risks of vicarious trauma. ● Challenges posed by generative AI.
5. Understand the legal and ethical aspects of digital forensics	5.1 Explain the legal frameworks governing digital evidence and forensic investigations. 5.2 Explain the ethical considerations in conducting forensic investigations. 5.3 Evaluate the impact of anti-forensic techniques on evidence collection and analysis.	**Legal and Ethical Concerns** ● Digital evidence admissibility from a legal perspective: requirements and handling practices to maintain admissibility. ● Privacy concerns and data protection laws; trading off personal privacy rights and security concerns. ● AI and copyright. **Anti-forensic Techniques** ● Encryption and steganography. ● Impact of anti-forensic techniques on investigations.

Assessment

To achieve a ‘pass’ for this unit, learners must provide evidence to demonstrate that they have fulfilled all the learning outcomes and meet the standards specified by all assessment criteria.

Learning Outcomes to be met	Assessment Criteria to be covered	Assessment type	Word count (approx. length)
LO1-LO3	All ACs under LO1-LO3	Report	3000 words
LO4-LO5	All ACs under LO4-LO5	Report	1500 words

Indicative Reading List

Cowen D. (2013) *Computer Forensics InfoSec Pro Guide* ISBN 978-0071742450

Carrier B. (2005) *File System Forensic Analysis* ISBN 978-0321268174

Casey E. (2011) *Digital Evidence and Computer Crime: Forensic Science, Computers, and the Internet (3rd edition)* ISBN 978-0123742681

Carvey H. (2016) *Windows Registry Forensics (2nd edition)* ISBN 978-0128032916

Altheide C. & Carvey H. (2011) *Digital Forensics with Open Source Tools* ISBN 978-1597495868

Ho A. T. S. & Li S. (2015) *Handbook of Digital Forensics of Multimedia Data and Devices* ISBN 978-1118640500

Ligh M. H., Case A., Levy J., Walters A. (2014) *The Art of Memory Forensics: Detecting Malware and Threats in Windows, Linux, and Mac Memory* ISBN 978-1118825099

Sheward M. (2017) *Digital Forensic Diaries* ISBN 978-1521514467

Additional Resources

ACPO Good Practice Guide for Digital Evidence (2012) https://library.college.police.uk/docs/acpo/digital-evidence-2012.pdf

ETHICAL HACKING

Unit Reference Number	D/651/4228
Unit Title	Ethical Hacking
Unit Level	5
Number of Credits	200
Total Qualification Time (TQT)	200 Hours
Guided Learning Hours (GLH)	100 Hours
Mandatory / Optional	Mandatory
Sector Subject Area (SSA)	06.1 Digital Technology (practitioners)
Unit Grading Type	Pass / Fail

Unit Aims

This unit aims to provide learners with the knowledge and skills required to identify and exploit vulnerabilities in cyber systems through ethical hacking techniques, otherwise known as penetration testing. Students will explore various offensive security methods used in penetration testing, including passive and active attacks, while gaining an understanding of the legal and ethical considerations involved. Learners will develop an appreciation of how ethical hacking helps to strengthen security by identifying weaknesses in systems before they can be exploited.

Learning Outcomes, Assessment Criteria and Indicative Content

Learning Outcome – The learner will:	Assessment Criteria – The learner can:	Indicative Content
1. Understand the role of penetration testing in enhancing Cyber Security.	1.1 Describe what is meant by penetration testing/ethical hacking. 1.2 Explain how penetration testing is connected to the broader notion of security testing. 1.3 Explain the differences between white box, grey box, and black box penetration testing.	**Penetration Testing (Ethical Hacking)** ● Definition of penetration testing and its role as a form of offensive security. ● Understanding penetration testing as deep and specific form of security testing, with a focus on exploiting vulnerabilities. ● Black box vs. white box vs. grey box penetration testing. **Phases of Penetration Testing**

	1.4 Analyse the key stages of penetration testing, including reconnaissance, scanning, vulnerability assessment, exploitation, and reporting. 1.5 Critically evaluate the benefits of penetration testing from a variety of perspectives.	● Reconnaissance (active and passive). ● Scanning to identify open ports and monitor network traffic. ● Vulnerability Assessment using data collected in reconnaissance and scanning phases. Use of resources (e.g. Common Vulnerabilities and Exposures (CVE) database) to identify vulnerabilities according to severity. ● Exploitation of vulnerabilities, ensuring the system is not compromised or damaged. ● Reporting and documenting findings. **Benefits of Penetration Testing** ● Compliance with laws and regulations. ● Prevention of cyberattacks, yielding benefits from financial, safety, and information security perspectives. ● Improved understanding of the latest attacks for Cyber Security researchers and professionals.
2. Be able to identify and assess security vulnerabilities through reconnaissance, scanning and vulnerability assessment.	2.1 Outline the key information that is collected as part of the reconnaissance stage. 2.2 Explain the purposes of reconnaissance, enumeration, and scanning for penetration testing. 2.3 Use industry-standard software tools to perform active and passive enumeration. 2.4 Use industry-standard software tools and databases to perform scanning.	**Reconnaissance** ● Reconnaissance is the initial collection of top-level information about a system or organisation, that can be used at later stages. ● Examples of general information (e.g. IP address ranges) that can be collected. **Enumeration** ● Active vs. passive enumeration and packet sniffing. ● System information through NetBIOS enumeration. ● Network structure and connected devices through DNS enumeration, SNMP enumeration, NTP enumeration.

		● Email usernames and data through SMTP enumeration. ● Directory contents through Lightweight Directory Access Protocol (LDAP) directory services. **Scanning** ● Port scanning (techniques including TCP Connect, TCP SYN, UDP, TCP FIN, TCP Null, TCP Xmas etc.) of open ports to identify running services and potential attacks. ● Network scanning (e.g. through ping sweeps). ● Vulnerability scanning (using vulnerability tools and databases e.g. Nessus, OpenVAS). ● Scanning tools: Nmap, Wireshark, Angry IP Scanner, Unicorn Scanner etc. *N.B. The tools to perform reconnaissance and enumeration/scanning, along with the next learning objective (exploitation), can be introduced in a practical way through an illustrative hands-on penetration test of a given web application, for example.*
3. Be able to exploit security vulnerabilities and report findings to help mitigate and prevent security attacks.	3.1 Use tools to exploit identified security vulnerabilities to agreed extents. 3.2 Use existing exploitations to capitalise and extract information or extend the degree of exploitation to test system vulnerabilities. 3.3 Analyse results to document actions, evidence, and vulnerabilities, and to provide security recommendations through an exploit report.	**Exploiting Security Vulnerabilities** ● Awareness of common vulnerabilities for exploitation: e.g. injection attacks, buffer overflows, Cross Site Scripting (XSS), privilege escalation, insecure authentication. ● Tools for exploitation: Metasploit, Burp Suite etc. ● Post exploitation: Privilege escalation, maintaining access, backdoors, data exfiltration, pivoting, lateral movement. ● Documenting and reporting: logs of actions, data capture, reproducibility of exploits, clear documentation of exploits for the exploit report.

	3.4 Evaluate results to enact steps to mitigate vulnerabilities.	**Exploit Report and Mitigation** ● Reporting findings and evaluating risk of the identified exploits, communication to stakeholders. ● Mitigating vulnerabilities and addressing exploits through recommendations e.g. patching, secure coding practices.
4. Understand the legal and ethical considerations for penetration testing.	4.1 Explain the legal frameworks governing ethical hacking and penetration testing. 4.2 Analyse the responsibilities of an ethical hacker, including responsible disclosure of vulnerabilities. 4.3 Evaluate the potential ethical dilemmas faced by penetration testers.	**Legal Frameworks and Penetration Testing Standards** ● This should include a strong emphasis on ethics and legality in hacking, including laws (in different areas of the world) such as the Computer Misuse Act in the United Kingdom, Computer Fraud and Abuse Act (CFAA) in the United States. GDPR can also be discussed. ● The Penetration Testing Execution Standard (PTES). **Ethical Issues** ● Responsible disclosure, client permissions, and hacking boundaries (i.e., what should and should not be done as part of ethical hacking). ● Grey areas: balancing client requirements with legal obligations and ethical boundaries.

Assessment

To achieve a ‘pass’ for this unit, learners must provide evidence to demonstrate that they have fulfilled all the learning outcomes and meet the standards specified by all assessment criteria.

Learning Outcomes to be met	Assessment Criteria to be covered	Assessment type	Word count (approx. length)
LO1-LO4	All ACs under LO1-LO4	Report	4500 words

Indicative Reading List

Velu V.K. (2017) *Mastering Kali Linux for Advanced Penetration Testing (2nd edition)* ISBN 978-1787120235

Bock L. (2022) *Learn Wireshark (2nd edition)* ISBN 978-1803231679

Seitz J. & Arnold T. (2021) *Black Hat Python: Python Programming for Hackers and Pentesters (2nd edition)* ISBN 978-1718501126

OccupyTheWeb (2018) *Linux Basics for Hackers: Getting Started with Networking, Scripting, and Security in Kali* ISBN 978-1593278557

Hickey M. (2020) *Hands on Hacking: Become an Expert at Next Gen Penetration Testing and Purple Teaming* ISBN 978-1119561453

Erickson J. (2008) *Hacking: The Art of Exploitation (2nd edition)* ISBN 978-1593271442

Kennedy D., Aharoni M., Kearns D., O'gorman J., Graham D. G. (2025) *Metasploit: The Penetration Tester's Guide (2nd edition)* ISBN 978-1718502987

Stuttard D. & Pinto M. (2011) *The Web Application Hacker's Handbook: Finding and Exploiting Security Flaws (2nd edition)* ISBN 978-1118026472

Additional Resources

OWASP Offensive Web Testing Framework: https://owasp.org/www-project-owtf/

Penetration Testing Execution Standard (PTES): https://www.pentest-standard.org/

MALWARE ANALYSIS

Unit Reference Number	F/651/4229
Unit Title	Malware Analysis
Unit Level	5
Number of Credits	200
Total Qualification Time (TQT)	200 Hours
Guided Learning Hours (GLH)	100 Hours
Mandatory / Optional	Mandatory
Sector Subject Area (SSA)	06.1 Digital Technology (practitioners)
Unit Grading Type	Pass / Fail

Unit Aims

This unit equips learners with the skills and knowledge to analyse and respond to malware threats. Students will learn about the different types of malware that exist and how they spread, including viruses, worms, ransomware, and trojans. They will learn how to detect, analyse, and mitigate malware, gaining practical experience in identifying malware and analysing malicious executables through static analysis, dynamic analysis, and reverse engineering. Challenges concerning identification and analysis of malware are also introduced.

Learning Outcomes, Assessment Criteria and Indicative Content

Learning Outcome – The learner will:	Assessment Criteria – The learner can:	Indicative Content
1. Understand the different types of malware that exist and how they spread.	1.1 Identify the different kinds of malware. 1.2 Describe the different kinds of malware 1.3 Analyse how malware can infect systems and spread across networks. 1.4 Evaluate the motives for the creation of malware.	**Types of Malware** ● Malware is ‘malicious software’ and encompasses any software designed to cause harm to a computer system. ● Viruses, worms, ransomware, Trojan Horse (Trojans), spyware, adware, rootkits, keyloggers, botnet, backdoor, etc. **Infection Methods** ● Social engineering (e.g. phishing), malicious

		email attachments, drive-by downloads, software vulnerabilities, removable media (USB drives), etc. **Malware Motives and Methods** ● Financial gain. ● Espionage. ● Sabotage. ● Hacktivism. ● Discussion of how the motive shapes the design of the malware and the associations between the common types of malware and these motives.
2. Understand how to detect malware and describe the challenges of doing so.	2.1 Explain the techniques used to identify malware. 2.2 Explain the characteristics of packed executables. 2.3 Critically analyse the challenges of detecting and analysing malware. 2.4 Evaluate the most effective methods of detecting malware	**Malware Identification** ● Signature-based detection. ● Heuristic analysis. ● Behaviour-based detection. ● Packed malware. **Packed executables** ● PE Headers. ● Packing using free e.g. UPX or custom e.g. Warzone Crypter packers to evade detection. ● Identifying packed malware using e.g. entropy. ● Unpacking packed malware using e.g. PE tools. **Challenges in detecting malware** ● Anti-reverse-engineering (obfuscation) and packing. ● Polymorphic and metamorphic malware. ● Rootkits.

3. Be able to perform malware analysis to identify and understand the function of malware.	3.1 Explain the principles of malware analysis. 3.2 Explain the importance of sandboxing to analyse malware behaviour. 3.3 Implement malware analysis tools to perform static and dynamic analysis of suspicious executables. 3.4 Use tools to perform reverse engineering of malware.	**Principles of Malware Analysis** ● x86 architecture; x86 assembly; Windows internals. ● Executable formats (e.g. PE headers for Windows, ELF files for Unix-like systems). ● Virtual machines and sandboxing. **Static and Dynamic Malware Analysis** ● Static analysis (reviewing code e.g. PE header). ● Dynamic analysis in sandboxed environments (e.g. virtual machines). Debugging using e.g. OllyDbg (x86 debugger) for dynamic malware analysis **Reverse Engineering** ● Reverse engineering toolkits e.g. IDA Free/Pro, Ghidra (National Security Agency) for disassembly. *Note that students may already be familiar with certain aspects of the x86 architecture, assembly etc. from the Principles of Computer Systems and Operating Systems units at Level 4.*
4. Understand strategies to protect from malware and recover from attacks.	4.1 Describe the most common techniques used to protect systems from malware infections. 4.2 Use recovery tools and techniques to restore systems compromised by malware. 4.3 Critically analyse the effectiveness of response strategies in real-world case studies.	**Malware Detection, Prevention and Recovery** ● Firewalls, antiviruses, intrusion detection/prevention systems (IDS/IPS), endpoint protection. ● Recovery from attacks: cleaning infected systems, backups, reimaging. ● Preventing the spread of malware: segmentation, access control policies, patch management, user awareness training.

	4.4 Evaluate different methods of containing the spread of malware across networks of computers.	**Case Studies** ● Analysis of high-profile malware attacks, (e.g. WannaCry, CryptoLocker, Emotnet, Stuxnet) and response strategies. ● Case studies can cover different types of malware, e.g. ransomware, trojans, worms.

Assessment

To achieve a 'pass' for this unit, learners must provide evidence to demonstrate that they have fulfilled all the learning outcomes and meet the standards specified by all assessment criteria.

Learning Outcomes to be met	Assessment Criteria to be covered	Assessment type	Word count (approx. length)
LO1 and LO4	All ACs under LO1 and AC 4.1-4.2	Report	1500 words
LO2-LO4	All ACs under LO2-LO3 and AC 4.3-4.4	Report	3000 words

Indicative Reading List

Sikorski M. (2012) *Practical Malware Analysis: The Hands-On Guide to Dissecting Malicious* Software ISBN 978-1593272906

NATO CCDCOE (2020) *Malware Reverse Engineering Handbook* https://ccdcoe.org/library/publications/malware-reverse-engineering-handbook/

Szor P. (2005) *The Art of Computer Virus Research and Defense* ISBN 978-0321304544

Cucci K. (2024) *Evasive Malware: Understanding Deceptive and Self-Defending Threats: A Field Guide to Detecting, Analyzing, and Defeating Advanced Threats* ISBN 978-1718503267

Kleymenov A. & Thabet A. (2022) *Mastering Malware Analysis - Second Edition: A malware analyst's practical guide to combating malicious software, APT, cybercrime, and IoT attacks (2nd edition)* ISBN 978-1803240244

Eilam E. (2005) *Reversing: Secrets of Reverse Engineering* ISBN 978-0764574818

Additional Resources

Ghidra: https://ghidra-sre.org/

IDA Pro: https://hex-rays.com/ida-pro

IMPORTANT NOTE

Whilst we make every effort to keep the information contained in programme specification up to date, some changes to procedures, regulations, fees matter, timetables, etc may occur during the course of your studies. You should, therefore, recognise that this booklet serves only as a useful guide to your learning experience.

For updated information please visit our website www.othm.org.uk.