UE20CS501 : Computer Systems for Programmers (4-0-0-4-4)

This course is an in-depth analysis of the programming interface to the hardware subsystems/mapping of contemporary processor architectures-based computing system. This course provides an end-to-end picture in sufficient advanced detail of Instruction Set Architecture, pipelining and program construct mapping to memory hierarchy.

Course Objectives:
- Introduce concepts of basic processor architecture and its design.
- Understand the concepts of pipeline architecture and hazards.
- Describe the memory hierarchy, cache memory and its optimization.
- Understand the concepts of linking, from traditional static linking, to dynamic linking of shared libraries at load time, to dynamic linking of shared libraries at run time.
- Describe the various benefits of a virtual memory system.

Course outcomes:
At the end of the course the student will be able to
- Trace the execution of a program with respect to modern processor architecture fundamentals, Caching.
- Design and implement Instruction Set Simulators for novel processor architectures.
- Utilize modern Processor architectures with complete understanding of contemporary programming language features.
- Write and debug complex programs.
- Gain extensive knowledge on Virtual Memory implementations.

Course Content:
Unit 1 : Introduction to computer systems
Compilation system, Processor functioning, Caches, Storage devices, Networks, information storage, Processes, Threads and Concurrency, Parallelism, Number representations. Machine level representation of programs.  
10 Hours

Unit 2 : Processor architecture
Instruction set architecture logic design, Clocking, Pipelining, Data hazards, Exception handling, Simulators.  
12 Hours

Unit 3 : Memory Hierarchy
10 Hours

Unit 4 : Linking
Compiler drivers, Static linking, Object files formats, relocatable object files, Symbol tables, Symbol resolution, Relocation, Dynamic linking, Shared libraries, Loading executable object files, Position independent code.  
12 Hours

Unit 5 : Virtual Memory
Page tables, Locality, Address translation, memory mapping, Dynamic memory allocation, Garbage collection, and Common memory related bugs.  
12 Hours

Tools/Languages: C/C++/Python/Java based IDEs and Instruction Set Simulators for Pipelined Multicore processors.

Text Book:

**Reference Book(s):**

Data Structures play a central role in modern computer science. Data Structures are the essential building blocks in building efficient algorithms. The course on Advanced Data Structures includes review of basic data structures like stacks, queues and lists. The course also includes study of complex data structures, analysis of complex data structures and their applications.

**Course Objectives:**
- Appreciate the impact of Data Structures on Algorithms, Program Design and Program Performance.
- Understand and apply Amortized Analysis on Data Structures, including Binary search trees, Mergeable Heaps and Dynamic Tables.
- Analyze the applications of static and dynamic Search Trees and Heaps.
- Understand advanced ADTs with Interface and Implementation separation.
- Understand State Space search and Spatial Data Structures with R-Trees.

**Course Outcomes:**
At the end of the course the student will be able to
- Demonstrate the notion of Abstract Data Types (ADT) & Recursive accesses on them.
- Illustrate the relation between Data Structure operations and Amortized Complexity analysis.
- How to Analyze Iterated Lists and variations thereof and demonstrate tree data structures and how to balance them, for specific access needs.
- Implement different Heaps and B-Tree variations.
- Analyze and implement Spatial data implementations and their search.

**Course Content:**

**Unit 1 : Complexity, Amortized Analysis, Abstract Data Types(ADT)**
Asymptotic Complexity Notations, Amortized Complexity Analysis of Stacks, Binary Counters, and Dynamic Tables, Concept of interface and implementation, Array as an ADT: Different types of Array Implementations. List Interface & List implementations, Concept of Iterator: Operations on Lists and Arrays – traverse, search, replace, reverse, copy. 12 hours

**Unit 2 : List and its variations, Graphs**
Doubly Linked List, Skip List: interface and implementation, Multilist: Sparse Matrices, Graphs-Representation, Graph Algorithms: elementary algorithms, Bellman Ford, Johnson’s algorithm for sparse graphs, Graph Isomorphism. 10 hours

**Unit 3 : Trees**
Tree: Basics, Self-Balancing Binary Search Trees, Trie: Prefix and Suffix trees, x-fast and y-fast trie, Treaps, Van Emde Boas Trees. 12 hours

**Unit 4 : Queue and Priority Queue**
Double Ended Queue, Priority Queues: Binomial Heaps, Leftist Heaps, Skewed Heaps, Fibonacci Heaps. 12 hours

**Unit 5 : Combination of Data Structures**
State space search techniques, Greedy method, Branch and Bound techniques, Introduction to Spatio-Temporal Data structures and R-Trees. 10 hours

**Tools/Language:** C, C++, Java or Python.

**Text Book :**
Reference Books:
UE20CS503 : Fundamentals of Scalable Computing (4-0-0-4)

With an increase in the amount of data and need for remote computation due to cloud computing, this course intends to introduce the students to the business needs that have driven the change and then the underlying technologies and principles of computing at scale.

Course Objectives:
The objective(s) of the course is to
- Introduce the basic principles of computing at scale and differentiate between scale up and scale out.
- Introduce the business need and applications for scaling computing.
- Introduce case studies of scalability from Cloud computing and Big Data and how the two relate to each other.
- Understand the theoretical considerations that impact the design of scalable systems.
- Introduce different programming models for computing at scale.

Course Outcomes:
At the end of the course the student will be able to
- Motivate and explain trade-offs in computing at scale
- Demonstrate development of Cloud/Hadoop applications.
- Evaluate Service-oriented technologies and their potential for transformation of business.
- Analyze various cloud programming models and choose the appropriate model for application development.
- Demonstrate use of tools for developing applications at scale.

Course Content:
Unit 1: Systems Modeling, Clustering and SOA
Operating systems concepts review, Scalable computing over the internet, Technologies for network based systems, system models for distributed and cloud computing, Software environments for distributed systems, performance metrics, Services and SOA, REST, RPC.
12 Hours

Unit 2: Cloud Platform Architectures
12 Hours

Unit 3: Programming Models
12 Hours

Unit 4: Big Data Programming models
Introduction, Distributed File systems Case Study: HDFS/GFS, MapReduce Programming Model, Case Study: Hbase/BigTable, Matrix operations.
10 Hours

Unit 5: Distributed systems and Trends
Master slave, p2p and overlay networks, Orchestration, Continuous integration, DevOps, Case Study: Kubernetes
10 Hours

Tools/Language: AWS, PostMan, Flask, Docker, Hadoop.

Reference Books:
5: Cloud Native DevOps with Kubernetes, Justin Domingus and John Arundel, O'Reilly, 2019
Machine Learning (ML) surrounds us today: in phones that respond to voice commands, programs that beat humans at Chess and Go, robots that assist surgeries, vehicles that drive in urban traffic, and systems that recommend products to customers on e-commerce platforms. This course aims to familiarise students with the breadth of modern ML, to impart an understanding of the dramatic surge of ML in the last decade, and to foster an appreciation for the distinctive role that ML can play in shaping the future of our society.

Course Objectives:
- Understand the basic concepts of learning and decision trees.
- Understand various techniques such as neural networks and genetic algorithms.
- Understand Stochastic Learning and in Decision Trees, Bayesian Models, and Hidden Markov Models.
- Understand Unsupervised Learning models like Clustering.
- Understand Computational learning complexity and Dimensionality reduction methods.

Course Outcomes:
At the end of the course the student will be able to,
- Display thorough knowledge about the key algorithms and theory that form the foundation of machine learning and computational intelligence.
- Identify and apply the appropriate machine learning technique to classification, pattern recognition, optimization and decision problems.
- Compare and contrast different machine learning algorithms.
- Design and implement classifiers for several domains.

Course Content:

**Unit 1 : Concept Learning and Decision Trees**
Defining a Machine Learning problem, Version Spaces and Candidate Elimination Algorithm and its Inductive Bias.  
10 Hours

**Unit 2 : Decision Tree learning**
Representation, Algorithm, Hypothesis Space Search, Inductive Bias and Issues.  
10 Hours

**Unit 3 : Neural Nets and SVM**
Perceptrons, Gradient Descent, Back propagation, Neural Nets, Support Vector Machines (SVM).  
12 Hours

**Unit 4 : Stochastic Models for Learning**
Bayesian Learning, Bayes Classifiers, Belief Networks, Bayesian Estimation, Expectation Maximisation (EM), Hidden Markov Models, Genetic Algorithms.  
12 Hours

**Unit 5 : UnSupervised Learning**
Clustering, Instance based Learning, Frequent Item Set Analysis, FP-Growth Algorithm. Learning Complexity and Dimensionality Reduction :PAC models, Sample Complexity, Principal Component Analysis and Singular Value Decomposition.  
12 Hours

**Tools/Languages** : Python 3.5 or above, Tensorflow 2.0 using Keras API.

**Text Book:**

**Reference Books:**
This course will cover the essentials of Cybersecurity, and students will learn about the characteristics of security principles, technologies, and procedures to secure networks, software and other assets of an organization. They will also gain an insight into Risk Management, Incident Management, Cryptography, Operations Security, Digital Forensics etc.

Course Objectives:
- Understand various cyber security issues with respect to operating system, wired and wireless networks.
- Analyse the risks and incidents’ response.
- Learn the art of encryption and decryption.
- To Learn the policies and loss of cyber world.
- Understand the violations in cyber world.

Course Outcomes:
At the end of the course, the student will be able to:
- Design a Threat model.
- Perform various attacks and their mitigation strategies.
- Perform cryptanalysis.
- Analyse the security issues and risks.
- Aware of the policies and law in cyber security.

Course Content:
Unit 1 : Introduction

Unit 2 : Network Security: Wired Security Issues

Unit 3 : Software and Web Security: Operating system security
Attack Surfaces of Set-UID Programs, Principle of Least Privilege; Environment variables attack surface, Control Hijacking– Buffer overflow and Countermeasures, Web security: Cross-Site Request Forgery, Cross-Site Scripting, SQL Injection, Threat Modelling- design, Types of Security testing : Fuzz testing, Vulnerability scanning, Penetration Testing; Static and Dynamic analysis.

Unit 4 : Cryptography

Unit 5 : Cyber security
The legal perspectives: Cyber crime and legal landscape around the world, Why do we need cyber laws: The Indian context, Indian IT Act, Challenges to Indian Law, Weakness in IT Act, Digital Signatures and Indian IT Act, Amendments to Indian IT Act, Cyber crime and punishment, Policy approaches.

Tools/ Languages : Nmap, Wireshark, Claynet

Text Book:
Reference Book(s):
2: “Introduction to Modern Cryptography”, Jonathan Katz, Yehuda Lindell, CRC Press, 2018
Algorithm Design and Analysis is fundamental and important part of Computer Science. The course on Advanced Algorithms introduces the learner to advanced techniques for design and analysis of Algorithms and explores a variety of applications.

Course Objectives:
- Understand in depth, Complexity notions and Computability of functions.
- Understand a few string matching/prediction algorithms and their applications.
- Understand Graph Centric, Max-Flow and Shortest Path algorithms.
- Understand efficient Polynomial Multiplication and hence DFT/FFT algorithm.
- Understand Number Theoretic algorithms and hence RSA cryptography.

Course Outcomes:
At the end of the course the student will be able to
- Perform Complexity analysis of algorithms.
- Design and implement advanced string matching algorithms.
- Design Maxflow, advanced Graph Centric algorithms and FFT Algorithms.
- Design and implement Number Theoretic algorithms and RSA encryption.
- Design Randomized and Approximate algorithms and estimate complexity.

Pre-requisite : UE20CS502-Advanced Data Structures.

Course Content:
Unit 1 : Review of Analysis Techniques
Growth of Functions, Asymptotic notations, Standard notations and common functions, Recurrences and Solution of Recurrence equations - The Substitution method, the Recurrence tree method, the Master method, Amortized Analysis; Aggregate, Accounting and Potential Methods.NP-Completeness.

12 Hours

Unit 2 : Shortest Path algorithms, Polynomials and FFT

12 Hours

Unit 3 : Number-Theoretic Algorithms
Elementary notions; GCD, Modular Arithmetic, Solving modular linear equations, The Chinese remainder theorem, Powers of an element; RSA cryptosystem; Primality testing; Integer factorization.

10 Hours

Unit 4 : String-Matching Algorithms

10 Hours

Unit 5 : Probabilistic, Randomized and Approximation Algorithms
Probabilistic analysis, Indicator Random variables, uses of indicator random variables, the Hiring problem, Classes of Randomized algorithms.Approximation Algorithms, Vertex Cover Problem, Travelling Sales man Problem, Randomisation and Linear Programming.

12 Hours

Tools/ Languages : C /C++.

Text Book:

Reference Book(s):
UE20CS552 : Distributed Computing (4-0-0-4-4)

Distributed computing deals with all forms of computing, information access, and information exchange across multiple processing platforms connected by computer networks. Design of distributed computing systems is a complex task. It requires a solid understanding of the design issues and an in depth understanding of the theoretical and practical aspects of their solutions. This course covers the fundamental principles and models underlying the theory, algorithms, and systems aspects of distributed computing.

Course Objectives:
- Expose students to both the abstractions and details of the file system.
- Introduce concepts related to distributed computing systems.
- To focus on performance and flexibility issues related to systems design decisions.
- To prepare students for an industrial programming environment.
- To prepare students to work on Cluster Computing.

Course Outcomes:
At the end of the course student will be able to,
- Apply knowledge of distributed systems techniques and methodologies.
- Explain the design and development of distributed computing and distributed computing applications.
- Use the application of fundamental Computer Science methods and algorithms in the development of distributed computing and distributed computing applications.
- Discuss the design and testing of a large software system, and to be able to communicate that design to others.
- Design and discuss cluster computing.

Course Content:

Unit 1 : Introduction to Distributed Computing

Unit 2 : Global snapshot
Snapshot algorithms for FIFO/ Non FIFO channels.Terminology and Basic algorithms: Classifications and basic concepts, Elementary graph algorithms, Synchronizers, Maximal Independent set, connected dominating set.Message ordering and group communication: Message ordering Paradigms, Group communication, Application level multicast.

Unit 3 : Distributed Mutual Exclusion
Assertion based and Token based Mutual exclusion.Consensus and Agreement Protocols: Agreement in failure free and systems with failures, wait-free shared memory consensus in asynchronous systems.

Unit 4 : Self-Stabilization
Designing self-stabilizing systems, self stabilizing distributed spanning tree, probabilistic self stabilizing leader election algorithm, self-stabilization as a solution to fault tolerance.Peer-to-Peer computing and Overlay graphs: Data indexing and overlays, unstructured and structured overlays: Bit Torrent, Tor, Bit coin, CHORD overlay, Internet graphs, Small world networks, Scale free networks, and Evolving networks.

Unit 5 : Cluster Computing
Tools / Languages: Wireshark / C compiler with Network Libraries.

Text Book:

Reference Book(s):
UE20CS561 : Million Way Parallelism (4-0-0-4-4)

With increasing amount of computation being done due to compute intensive applications like machine learning and audio/video processing, it is imperative to understand various parallel programming models. The student must also become familiar with various different hardware choices and the design choices. The course will also introduce various tools to work.

Course Objectives:
- Introduce the various parallel programming models required to scale.
- Introduce applications that benefit from scalability.
- Introduce different types of hardware such as multi-core CPUs, GPUs and FPGAs.
- Understand and use the tools to analyze these applications.
- Understand the impact of role of various other system components on extracting performance in parallel programs.

Course Outcomes:
At the end of the course the student will be able to,
- Choose the right programming model for a problem.
- Demonstrate development of applications on different types of hardware.
- Evaluate choice of the right type of hardware to solve the problem.
- Demonstrate use of tools for developing parallel applications and debugging them.
- Analyze systems for issues in performance.

Course Content:
Unit 1 : Introduction and Parallel Program Design
Multicore, Taxonomy, Performance Metrics, Amdahl’s Law, Gustafson’s law, Decomposition patterns, Program structure patterns. Matching program structure with decomposition. 10 Hours

Unit 2 : Shared Memory Programming
Threads, design concerns, semaphores, applying semaphores, debugging multithreaded applications, OpenMP- loop level parallelism, Task level parallelism, Synchronization, Correctness and Optimization. Case Study. 12 Hours

Unit 3 : GPU Programming
GPU architecture, CUDA - programming model, execution model, Memory hierarchy. Optimization techniques, Debugging, Case Study. 12 Hours

Unit 4 : FPGAs
Motivation, FPGA capabilities, Elements – Look up table, switches, logic blocks, interconnect, I/O blocks, multipliers, memory blocks, CAD software – HDL Simulator, technology mapping, placement, bitstream generation. 12 Hours

Unit 5 : Miscellaneous and Trends
FPGA application – Machine learning, power considerations, memory systems (3D), Interconnection Networks. 10 Hours

Tools / Languages : OpenMP, CUDA.

Text Book:

Reference Book:
Course Objectives:
- Introduce the student to advances in multimedia technologies relevant to Big Data.
- Understand the business relevance of speech and natural language technologies.
- Introduce the basic models used for processing speech and natural language.
- Work with tools to perform speech/natural language processing.
- Gain a practical insight into solving these problems.

Course Outcomes:
At the end of the course the student will be able to,
- Demonstrate the use of speech/natural language processing in solving real-life problems.
- Demonstrate the use of tools for performing speech/NLP.
- Demonstrate capability to perform analysis and compare various models.
- Work in a team to solve related SNLP problems.
- Communicate the solution to the instructor using a report.

Course Content:
Unit 1: Introduction

12 Hours

Unit 2: Vector Semantics and Embeddings

12 Hours

Unit 3: Lexical Semantics

10 Hours

Unit 4: Syntactic Parsing
Parsing as Search, Ambiguity, Search in the Face of Ambiguity, Dynamic Programming Parsing Methods, Partial Parsing, Statistical Parsing – Probabilistic CFGs, Probabilistic CKY Parsing, Ways to Learn PCFG, Rule Probabilities, Problems with PCFGs, Improving PCFGs, Probabilistic Lexicalized CFGs, Evaluating Parsers. Dependency Parsing.

12 Hours

Unit 5: Phonetics
Likelihood Computation, Lexicon and Language Model, Search and Decoding, Embedded Training, Word Error Rate.

10 Hours

**Tools/Languages:** CoreNLP, Natural Language Toolkit (NLTK), TextBlob, Gensim, SpaCy, an industrial-strength NLP, PyTorch-NLP, OpenNLP.

**Text Book:**

**Reference Book(s):**
UE20CS563 : Topics in Computer and Network Security (4-0-0-4-4)

Computer Network security prevent and monitor unauthorized access, misuse, modification, or denial of a computer network and network-accessible resources by entailing the policies and practices adaptation. It is important to know about different kinds of vulnerabilities like buffer overflow, SQL Injection, Firewall injection attacks, etc.

Course Objectives:
- Have First depth view of Perimeter Security (Firewall, IDS, IPSEC, VPN).
- Learn Authentication and Access management.
- Have a beginner’s overview of Cryptography, Malware, Secure Programming, Applications Security.
- Explore case studies, hands on experience through assignments/ project, extra readings for alternate view or real time application.

Course Outcomes:
At the end of course, student will be able to
- Have a good understanding of security as a problem and current solutions like Firewall, IDS, Authentication, Access Control.
- Think what causes software vulnerabilities at a conceptual level.
- Do penetration testing, use cryptography appropriately.
- Identify many vulnerabilities like buffer overflow, SQL injection and resolve them.
- Design a defending protocol for many network vulnerabilities.

Course Contents:
Unit 1: Introduction to Computer Network Security

10 hours

Unit 2: Network Security Analysis

12 hours

Unit 3: Network Security Systems

12 hours

Unit 4: Risk, DNS, Heartbleed

12 hours

Unit 5: Cloud Security, Wireless Network Security
Security. Giving hands on experience for relevant topics in the form of Lab or Assignment, Relevant cyber security Case for undergraduate students are discussed.

**10 hours**

**Tools / Languages**: Wireshark, Python, Seed Ubuntu 16 version, Netwox, C Programming, Scapy.

**Text Book**:

**References**
3: “Intrusion and intrusion detection,” John McHugh
5: Programming Satan's Computer Ross Anderson and Roger Needham
6: Why crypto systems fail - Ross Anderson
With increasing usage of the cloud to develop applications, it is imperative to understand the various threats and security challenges due to this shift. The course will introduce the concepts and the tools required for dealing with some of the security challenges.

**Course Objectives:**
- Introduce students to various aspects of cloud security
- Equip students for better appreciation of technology, human and societal aspects of security with respect to the cloud
- Introduce challenges of security with various models of cloud such as IaaS, PaaS and SaaS
- Introduce the student to threat analysis and modelling in the cloud.

**Course Outcomes:**
At the end of the course the student will be able to
- Understand various security issues in the cloud.
- Design appropriate security for various cloud models.
- Analyze security from both technology and legal perspective
- Design mechanisms for security for access control and data

**Course Content:**

**Unit 1: Introduction and IaaS Security**
10 Hours

**Unit 2 : PaaS and SaaS Security**
12 Hours

**Unit 3 : Data Security and IAM**
Public cloud security limitations, aspects of Data security, mitigation, Provider Data and it’s security. IAM – Challenges, Architecture, IAM in Cloud - Google, Azure, SaaS.  
12 Hours

**Unit 4 : Privacy, Audit and Compliance**
12 Hours

**Unit 5 : Threat models**
What is a threat model, finding threats – STRIDES and attack trees. Privacy Tools, Managing and Addressing Threats.  
10 Hours

**Tools/Languages :** Google Oauth, Gvisor/Firecracker-Secure Container, Nginix Dos Protection.

**Text Books:**
Physical entity is simulated into virtual or imaginary environment which are designed as software or as a program that defy beliefs of a user compelling him/her to accept it as actual reality. Virtual Reality actually exploits and plays with the sensations & perceptions of our brain by simulating an artificial environment which actually doesn’t really exists but our brains thinks that it does it’s just like make belief.

**Course Objectives:**
- To create an adaptive 3D the virtual environment that meets the needs of trainee interpreters and those who need to learn about how to work with interpreters.
- To develop a range of interpreting scenarios (e.g. a business meeting room, a court room, a tourist office, a community centre) that can be run in different modes (‘interpreting practice’, ‘exploration’ and ‘live’).
- To develop multilingual content for use in the interpreting scenarios of the virtual environment (as source texts for interpreting practice), by using and adapting existing multimedia corpora from the LLP project BACKBONE [1] and the ELISA corpus [2], and creating three new corpora in Greek, Russian and Hebrew.
- To create pedagogical activities for interpreting students and users of interpreting services (e.g. interpreting skills, awareness-raising activities).
- To test and evaluate the virtual environment and the pedagogic content (the multilingual material and the pedagogical activities) from both functional and pedagogical perspectives.

**Course Outcomes**
- Differentiate between Virtual, Mixed and Augmented Reality platforms.
- Identify appropriate design methodologies for immersive technology development, especially from a physiological perspective.
- Demonstrate foundational literacy in game engine use.
- Effectively categories the benefits/shortcomings of available immersive technology platforms.
- Developing Pedagogical activities for interpreting students and users of interpreting services.

**Course Content:**

**Unit 1 : Introduction and Input devices**

*Input Devices: Trackers, Navigation, Gestures, Interface*

10 Hours

**Unit 2 : Output Devices: Graphics, Three dimensional Sound, AND Haptic Displays**


12 Hours

**Unit 3 : Modelling**

**VR PROGRAMMING:** Toolkits and Scene Graphs, WorldToolKit, Model Geometry and Appearance, The WTK Scene Graph, Sensors and Action Functions, WTK Networking, Java 3D, Model Geometry and Appearance, Java 3D Scene Graph, Sensors and Behaviors, Java 3D Networking, WTK and Java 3D Performance Comparison, General Haptics Open Software Toolkit, GHOST Integration with the Graphics Pipeline, The GHOST Haptics Scene Graph, Collision Detection and Response, Graphics and PHANToM Calibration, PeopleShop, DI-Guy Geometry and Path, Sensors and Behaviors, PeopleShop Networking.

**Unit 4: Human factors in VR**

**Unit 5: Emerging Application of VR**
VR Applications in Manufacturing, Virtual Prototyping, Other VR Applications in Manufacturing, Applications of VR in Robotics, Robot Programming Robot Teleoperation, Information Visualization, Oil Exploration and Well Management, Volumetric Data Visualization.

**Tools/Languages:** Unity3d, React VR.

**Text Book:**

**Reference Book(s):**
The course presents the challenges and mechanisms to develop a safe and secure software. The main focus of the course is to provide an insight of the software vulnerabilities, its consequences during exploitation and the procedure to harden the system against those attacks.

Course Objectives:
- Apply and manage secure software development process.
- Gain a good comprehension of the landscape of Operating system vulnerabilities.
- Gain the ability to analyse the secure coding practises and advocate about the significance of vulnerabilities.
- Learn representative tools for web application security analysis.
- Realize the capabilities and limitations by threat modelling and understand the best testing practices.

Course Outcomes:
At the end of the course, the student will be able to
- Apply a strategy to design software with security.
- Understand the vulnerabilities of commonly used Operating Systems and browsers.
- Comprehend the security limitations of C programming language.
- Analyse various threats and privacy issues involved in web application development and apply mitigation approaches to avoid them.
- Inspect various security testing strategies and apply penetration testing to understand the intrusion resiliency.

Pre-requisite Course: UE20CS505-Cyber Security Essentials.

Course Content
Unit 1 : Introduction

12 hours

Unit 2 : Privilege Escalation attacks
Set UID program and environment variable, Shell shock attack program. Buffer Overflow: Vulnerable code, Stack and Function Invocation, Challenges in exploitation, Shellcode, Countermeasures, Function Prologue and Epilogue; Format String Vulnerability: Functions with Variable number of arguments, Exploiting format string and Mitigation approaches.

12 hours

Unit 3 : Malware and Threat Modelling
Review of Threat Modelling techniques, Applying STRIDE threat modelling technique, Privacy threats, Taxonomy Of Privacy, privacy tools, processing threats, defensive tactics and technologies. EOP card game, Malware- AbraWorm, Stuxnet worm and Morris Worm.

10 hours

Unit 4 : Web Application Security

12 Hours

Unit 5 : Security Testing
Static analysis, Penetration testing - Benefits and Drawbacks, Pen testing tools review, Network probing using nmap and Metasploit framework; Ethical Hacking, FUZZING. A case study and practical study on exploiting the vulnerability and penetrating the system.

10 hours
Tools usage: Claynet and Wireshark.

Text Book:

Reference Book(s):
UE20CS581 : Advanced Big Data Analytics (4-0-4-4)

The course explores the big data analytics lifecycle: question formulation, data collection and cleaning, exploratory analysis, statistical inference, prediction and decision-making. Focuses on building analytical models using key principles and techniques.

Course Objectives:
- Introduce alternative techniques to perform big data processing.
- Introduce applications of Big Data Processing.
- Use tools and techniques to analyze a large data corpus.
- Technologies for performing processing at large scale.
- Perform a group-based activity to apply tools and techniques learnt to a real world problem.

Course Outcomes:
At the end of the course the student will be able to
- Motivate and explain trade-offs in big data processing technique design and analysis in written and oral form.
- Demonstrate the usage of tools to design Big Data applications.
- Demonstrate development of analytics applications using alternative technologies to Hadoop.
- Demonstrate ability to work in a small team to solve a real world problem with applications to the society
- Communicate the design through a presentation and build a prototype to showcase the design.

Course Content:

Unit 1 : Introduction and MapReduce
Introduction, HDFS overview, formats, MapReduce architecture,YARN, limitations of MapReduce, Algorithms: PageRank .
10 Hours

Unit 2 : In Memory processing
Alternatives to Map Reduce – Iterative, Workflow processing, Graph Processing, In-memory computation., Workflow model case study – Apache Spark – RDDs, Scala Performance advantages.
12 Hours

Unit 3 : Other models and algorithms
12 Hours

Unit 4 : Timeseries analysis
Business applications of timeseries data, challenges/tools to process timeseries data. Searching/Matching algorithms.
10 Hours

Unit 5 : Multimedia –Speech Processing
12 Hours


Text Book:

Reference Books:
Deep Learning has received a lot of attention over the past few years and has been employed successfully by companies like Google, Microsoft, IBM, Facebook, Twitter etc. to solve a wide range of problems in Computer Vision and Natural Language Processing. In this course we will learn about the building blocks used in these Deep Learning based solutions. At the end of this course students would have knowledge of deep architectures used for solving various Vision and NLP tasks.

Course Objectives:
- To impart hands-on knowledge on Advanced Machine Learning Topics.
- Introduce students to programming with TensorFlow and Keras tools.
- Provide in-depth coverage of Support Vector Machines.
- Introduce students to Deep Learning techniques – CNN and RNN.
- Introduce students to Reinforcement Learning and Generative Adversarial Networks.

Course Outcomes:
At the end of this course, the student will be able to:
- Implement Machine Learning techniques with TensorFlow and Keras and develop simple game engines using Reinforcement Learning.
- Solve time-series related problems with RNN.
- Classify real-world data using Support Vector Machines.
- Classify images using CNN.
- Generate data in the form of images using GAN.

Course Contents
Unit 1: Introduction Historical Trends in Deep Learning
Deep Feed forward Networks Example: Learning XOR, Gradient-Based Learning, Hidden Units, Architecture Design, Back-Propagation and Other Differentiation Algorithms, Historical Notes.
Regularization for Deep Learning Parameter Norm Penalties, Norm Penalties as Constrained Optimization, Regularization and Under-Constrained Problems, Dataset Augmentation, Noise Robustness, Semi-Supervised Learning, Multi-Task Learning, Early Stopping, Parameter Tying and Parameter Sharing, Sparse Representations, Bagging and Other Ensemble Methods, Dropout, inverted drop out, adversarial training, Tangent Distance, Tangent Prop, and Manifold Tangent Classifier.
12 Hours

Unit 2: Convolutional Networks
10 Hours

Unit 3: Sequence Modeling: Recurrent and Recursive Nets
Unfolding Computational Graphs, Recurrent Neural Networks, Bidirectional RNNs, Encoder-Decoder Sequence-to-Sequence Architectures, Deep Recurrent Networks, Recursive Neural Networks, The Challenge of Long-Term Dependencies, Echo State Networks, Leaky Units and Other Strategies for Multiple Time Scales, The Long Short-Term Memory and Other Gated RNNs, Optimization for Long-Term Dependencies, Explicit memory, attention mechanism.
12 Hours

Unit 4: Autoencoders
12 Hours

Unit 5: Practical Methodology
Performance Metrics, Default Baseline Models, Determining Whether to Gather More Data, Selecting Hyperparameters, Debugging Strategies, Example: Multi-Digit Number Recognition Applications Large Scale Deep Learning, Computer Vision, Speech Recognition, Natural Language Processing, Other Applications. Overview of GAN.

10 Hours

Tools / Languages: Tensorflow 1.15, Keras 2.3.1, Python 3.7.

Text Book:
   http://www.deeplearningbook.org/(E-Book)

Reference Books:
Cryptography is the science of securing data by using mathematical concepts. Cryptography involves the authentication and verification of data in all domains by applying Cryptographic protocols.

Course Objectives:
- Enable to learn the fundamental concepts of cryptography and utilize these techniques in computing systems.
- Discuss about various encryption techniques.
- Understand the concept of public key Cryptography.
- Introduce message authentication and hash function.
- Provide Lab sessions for each unit to help gain deeper insight into Cryptography.

Course Outcomes:
At the end of the course the student will be able to,
- Appreciate the impact of cyber-attacks on the society and the necessity of cryptography.
- Analyse Cryptographic techniques using the mathematical foundations of cryptography.
- Design applications/protocols using cryptographic techniques.
- Apply cryptanalysis to solve real time problems.
- Evaluate the authentication and Hash Algorithms.

Course Content:
**Unit 1 : Introduction to Cryptography**

**Unit 2 : Modern Cryptography**
Principles of Modern cryptography, Perfectly-secret encryption – Vernam’s One-time-pad encryption – Limitations, Shannon’s theorem, Stream Ciphers, Block cipher design principles, Block Vs Stream cipher.

**Unit 3 : Private Key Cryptography**

**Unit 4 : Public Key Cryptography**

**Unit 5 : MAC and Hash**

Tools / Languages : Seed virtual machine environment.

Text Book:

Reference Books: