UNIVERSITY OF MUMBAI

Syllabus for M.Sc. I.T. Part II
Semester III
Practical List
Programme: M.Sc.
Subject: Information Technology
CHOICE BASED(REVISED)
with effect from the academic year
2020 – 2021

M. Sc (Information Technology)		Semester – III	
Course Name: Applied Artificial Intelligence Practical		**Course Code: PSIT3P2a**	
Periods per week (1 Period is 60 minutes)		4	
Credits		2	
		Hours	**Marks**
Evaluation System	**Practical Examination**	2	50
	Internal	--	--

List of Practical:	
1.	Design an Expert system using AIML E.g: An expert system for responding the patient query for identifying the flu.
2.	Design a bot using AIML.
3.	Implement Bayes Theorem using Python
4.	Implement Conditional Probability and joint probability using Python
5.	Write a program for to implement Rule based system.
6.	Design a Fuzzy based application using Python / R.
7.	Write an application to simulate supervised and un-supervised learning model.
8.	Write an application to implement clustering algorithm.
9.	Write an application to implement support vector machine algorithm.
10.	Simulate artificial neural network model with both feedforward and backpropagation approach. [You can add some functionalities to enhance the model].
11.	Simulate genetic algorithm with suitable example using Python / R or any other platform.
12.	Design an Artificial Intelligence application to implement intelligent agents.
13.	Design an application to simulate language parser.
14.	Design an application to simulate semantic web.

M. Sc (Information Technology)		**Semester – III**	
Course Name: Computer Vision Practical		**Course Code: PSIT3P2b**	
Periods per week (1 Period is 60 minutes)		4	
Credits		2	
		Hours	**Marks**
Evaluation System	**Practical Examination**	2	50
	Internal	--	--

M. Sc (Information Technology)		Semester – III	
Course Name: Cloud Application Development Practical		**Course Code: PSIT3P2c**	
Periods per week (1 Period is 60 minutes)		4	
Credits		2	
		Hours	**Marks**
Evaluation System	**Practical Examination**	2	50
	Internal	--	--

List of Practical:	
1.	Develop an ASP.NET Core MVC based Stateless Web App.
2.	Develop a Spring Boot API.
3.	Create an ASP.NET Core Web API and configure monitoring.
4.	a. Create an Azure Kubernetes Service Cluster
	b. Enable Azure Dev Spaces on an AKS Cluster
	c. Configure Visual Studio to Work with an Azure Kubernetes Service Cluster
	d. Configure Visual Studio Code to Work with an Azure Kubernetes Service Cluster
	e. Deploy Application on AKS i. Core Web API ii. Node.js API
5.	Create an AKS cluster a. from the portal b. with Azure CLI
6.	Create an Application Gateway Using Ocelot and Securing APIs with Azure AD.
7.	Create a database design for Microservices an application using the database.
8.	a. Create an API management service b. Create an API gateway service
9.	Demonstrate a. Securing APIs with Azure Active Directory. b. Issuing a custom JWT token using a symmetric signing key c. Pre-Authentication in Azure API Management d. AWS API Gateway Authorizer
10.	Create a serverless API using Azure functions
11.	Create an AWS Lambda function
12.	Build AWS Lambda with AWS API gateway

M. Sc (Information Technology)		Semester – III	
Course Name: Security Breaches and Countermeasures Practical		**Course Code: PSIT3P3d**	
Periods per week (1 Period is 60 minutes)		4	
Credits		2	
		Hours	**Marks**
Evaluation System	**Practical Examination**	2	50
	Internal	--	-

List of Practical:	
1.	a. Use the following tools to perform footprinting and reconnaissance
	i. Recon-ng (Using Kali Linux)
	ii. FOCA Tool
	iii. Windows Command Line Utilities
	• Ping
	• Tracert using Ping
	• Tracert
	• NSLookup
	iv. Website Copier Tool – HTTrack
	v. Metasploit (for information gathering)
	vi. Whois Lookup Tools for Mobile – DNS Tools, Whois, Ultra Tools Mobile
	vii. Smart Whois
	viii. eMailTracker Pro
	ix. Tools for Mobile – Network Scanner, Fing – Network Tool, Network Discovery Tool, Port Droid Tool
	b. Scan the network using the following tools:
	i. Hping2 / Hping3
	ii. Advanced IP Scanner
	iii. Angry IP Scanner
	iv. Masscan
	v. NEET
	vi. CurrPorts
	vii. Colasoft Packet Builder
	viii. The Dude
	ix.
2.	c. Use Proxy Workbench to see the data passing through it and save the data to file.
	d. Perform Network Discovery using the following tools:
	i. Solar Wind Network Topology Mapper
	ii. OpManager
	iii. Network View
	iv. LANState Pro
	e. Use the following censorship circumvention tools:
	i. Alkasir
	ii. Tails OS

	f. Use Scanning Tools for Mobile – Network Scanner, Fing – Network Tool, Network Discovery Tool, Port Droid Tool
3.	a. Perform Enumeration using the following tools:
	i. Nmap
	ii. NetBIOS Enumeration Tool
	iii. SuperScan Software
	iv. Hyena
	v. SoftPerfect Network Scanner Tool
	vi. OpUtils
	vii. SolarWinds Engineer's Toolset
	viii. Wireshark
	b. Perform the vulnerability analysis using the following tools:
	i. Nessus
	ii. OpenVas
4.	a. Perform mobile network scanning using NESSUS.
	b. Perform the System Hacking using the following tools:
	i. Winrtgen
	ii. PWDump
	iii. Ophcrack
	iv. Flexispy
	v. NTFS Stream Manipulation
	vi. ADS Spy
	vii. Snow
	viii. Quickstego
	ix. Clearing Audit Policies
	x. Clearing Logs
5.	a. Use wireshark to sniff the network.
	b. Use SMAC for MAC Spoofing.
	c. Use Caspa Network Analyser.
	d. Use Omnipeek Network Analyzer.
6.	a. Use Social Engineering Toolkit on Kali Linux to perform Social Engineering using Kali Linux.
	b. Perform the DDOS attack using the following tools:
	i. HOIC
	ii. LOIC
	iii. HULK
	iv. Metasploit
	c. Using Burp Suite to inspect and modify traffic between the browser and target application.
7.	a. Perform Web App Scanning using OWASP Zed Proxy.
	b. Use droidsheep on mobile for session hijacking
	c. Demonstrate the use of the following firewalls:
	i. Zonealarm and analyse using Firewall Analyzer.
	ii. Comodo Firewall
	d. Use HoneyBOT to capture malicious network traffic.

	e. Use the following tools to protect attacks on the web servers:
	i. ID Server
	ii. Microsoft Baseline Security Analyzer
	iii. Syhunt Hybrid
8.	a. Protect the Web Application using dotDefender.
	b. Demonstrate the following tools to perform SQL Injection:
	i. Tyrant SQL
	ii. Havij
	iii. BBQSQL
9.	Use Aircrack-ng suite for wireless hacking and countermeasures.
10.	Use the following tools for cryptography
	i. HashCalc
	ii. Advanced Encryption Package
	iii. MD5 Calculator
	iv. TrueCrypt
	v. CrypTool

M. Sc (Information Technology)		Semester – III	
Course Name: Machine Learning Practical		**Course Code: PSIT3P3a**	
Periods per week (1 Period is 60 minutes)		4	
Credits		2	
		Hours	**Marks**
Evaluation System	**Practical Examination**	2	50
	Internal	--	-

List of Practical:	
1.	a. Design a simple machine learning model to train the training instances and test the same.
	b. Implement and demonstrate the FIND-S algorithm for finding the most specific hypothesis based on a given set of training data samples. Read the training data from a .CSV file
2.	a. Perform Data Loading, Feature selection (Principal Component analysis) and Feature Scoring and Ranking.
	b. For a given set of training data examples stored in a .CSV file, implement and demonstrate the Candidate-Elimination algorithm to output a description of the set of all hypotheses consistent with the training examples.
3.	a. Write a program to implement the naïve Bayesian classifier for a sample training data set stored as a .CSV file. Compute the accuracy of the classifier, considering few test data sets.
	b. Write a program to implement Decision Tree and Random forest with Prediction, Test Score and Confusion Matrix.
4.	a. For a given set of training data examples stored in a .CSV file implement Least Square Regression algorithm.
	b. For a given set of training data examples stored in a .CSV file implement Logistic Regression algorithm.
5.	a. Write a program to demonstrate the working of the decision tree based ID3 algorithm. Use an appropriate data set for building the decision tree and apply this knowledge to classify a new sample.
	b. Write a program to implement k-Nearest Neighbour algorithm to classify the iris data set.
6.	a. Implement the different Distance methods (Euclidean) with Prediction, Test Score and Confusion Matrix.
	b. Implement the classification model using clustering for the following techniques with K means clustering with Prediction, Test Score and Confusion Matrix.
7.	a. Implement the classification model using clustering for the following techniques with hierarchical clustering with Prediction, Test Score and Confusion Matrix
	b. Implement the Rule based method and test the same.
8.	a. Write a program to construct a Bayesian network considering medical data. Use this model to demonstrate the diagnosis of heart patients using standard Heart Disease Data Set.
	b. Implement the non-parametric Locally Weighted Regression algorithm in order to fit data points. Select appropriate data set for your experiment and draw graphs.

9.	a. Build an Artificial Neural Network by implementing the Backpropagation algorithm and test the same using appropriate data sets.
	b. Assuming a set of documents that need to be classified, use the naïve Bayesian Classifier model to perform this task.
10.	a. Write a program to demonstrate the working of the decision tree based ID3 algorithm. Use an appropriate data set for building the decision tree and apply this knowledge to classify a new sample.
	b. Build an Artificial Neural Network by implementing the Backpropagation algorithm and test the same using appropriate data sets.
11.	Perform Text pre-processing, Text clustering, classification with Prediction, Test Score and Confusion Matrix

M. Sc (Information Technology)		**Semester – III**	
Course Name: Biomedical Image Processing Practical		**Course Code: PSIT3P3b**	
Periods per week (1 Period is 60 minutes)		4	
Credits		2	
		Hours	**Marks**
Evaluation System	**Practical Examination**	2	50
	Internal	--	-

M. Sc (Information Technology)		Semester – III	
Course Name: Cloud Management Practical		**Course Code: PSIT3P3c**	
Periods per week (1 Period is 60 minutes)		4	
Credits		2	
		Hours	**Marks**
Evaluation System	**Practical Examination**	2	50
	Internal	--	-

List of Practical:	
1.	a. Create and Manage Cloud using SCVMM 2019
	b. Deploy a guarded host fabric using Microsoft SCVMM 2019
2.	a. Deploy and manage SDN Infra structure using SCVMM 2019
	b. Deploy and Manage Storage Space Direct (S2D) using SCVMM 2019
3.	a. Deploy Service Manager 2019 and install on 4 Computer Scenario
	b. Setup SQL Server reporting Service using Service Manager 2019
4.	a. User Connectors to import data: i. Import data from Active Directory Domain Services ii. Import data and alerts from Operations Manager iii. Import data from Configuration Manager iv. Import runbooks from Orchestrator v. Import data from VMM vi. Use a CSV file to import data
	b. Automate IT processes with workflows vii. Add or remove workflow activities viii. Configure the way activities manage and pass information ix. Deploy a workflow to Service Manager using the Authoring Tool x. Configure the Activities Toolbox in the Authoring Tool
5.	a. Managing devices with Configuration Manager
	b. Design a hierarchy of sites using Microsoft End Point Configuration manager.
6.	a. Data transfers between sites i. Types of data transfer ii. File-based replication iii. Database replication
	b. Configure sites and hierarchies i. Add site system roles ii. Install site system roles iii. Install cloud-based distribution points iv. Configuration options for site system roles v. Database replicas for management points
7.	a. Install Orchestrator.
	b. Create and test a monitor runbook
8.	a. Manage Orchestrator Servers - 1 i. Runbook permissions

	ii. Back up Orchestrator iii. Bench mark iv. Optimize performance of .Net activities v. Configure runbook throttling vi. Recover a database
	b. Manage Orchestrator Servers - 2 i. Recover web components ii. Add an integration pack iii. View Orchestrator data with PowerPivot iv. Change Orchestrator user groups v. Common activity properties vi. Computer groups
9.	Install and Deploy DPM i. Install DPM ii. Deploy the DPM protection agent iii. Deploy protection groups iv. Configure firewall settings
10.	Protect Workloads i. Back up Hyper-V virtual machines ii. Back up SQL Server with DPM iii. Back up file data with DPM iv. Backup system state and bare metal v. Backup and restore VMware servers vi. Backup and restore VMM servers

M. Sc (Information Technology)		Semester – III	
Course Name: Malware Analysis Practical		**Course Code: PSIT3P3d**	
Periods per week (1 Period is 60 minutes)		4	
Credits		2	
		Hours	**Marks**
Evaluation System	**Practical Examination**	2	50
	Internal	--	-

List of Practical:	
1.	a. Files: *Lab01-01.exe* and *Lab01-01.dll*.
	i. Upload the files to *http://www.VirusTotal.com/* and view the reports. Does either file match any existing antivirus signatures?
	ii. When were these files compiled?
	iii. Are there any indications that either of these files is packed or obfuscated? If so, what are these indicators?
	iv. Do any imports hint at what this malware does? If so, which imports are they?
	v. Are there any other files or host-based indicators that you could look for on infected systems?
	vi. What network-based indicators could be used to find this malware on infected machines?
	vii. What would you guess is the purpose of these files?
	b. Analyze the file *Lab01-02.exe*.
	i. Upload the *Lab01-02.exe* file to *http://www.VirusTotal.com/*. Does it match any existing antivirus definitions?
	ii. Are there any indications that this file is packed or obfuscated? If so, what are these indicators? If the file is packed, unpack it if possible.
	iii. Do any imports hint at this program's functionality? If so, which imports are they and what do they tell you?
	iv. What host- or network-based indicators could be used to identify this malware on infected machines?
	c. Analyze the file Lab01-03.exe.
	i. Upload the *Lab01-03.exe* file to *http://www.VirusTotal.com/*. Does it match any existing antivirus definitions?
	ii. Are there any indications that this file is packed or obfuscated? If so, what are these indicators? If the file is packed, unpack it if possible.
	iii. Do any imports hint at this program's functionality? If so, which imports are they and what do they tell you?
	iv. What host- or network-based indicators could be used to identify this malware on infected machines?
	d. Analyze the file Lab01-04.exe.
	i. Upload the *Lab01-04.exe* file to *http://www.VirusTotal.com/*. Does it match any existing antivirus definitions?
	ii. Are there any indications that this file is packed or obfuscated? If so, what are these indicators? If the file is packed, unpack it if possible.
	iii. When was this program compiled?

	iv. Do any imports hint at this program's functionality? If so, which imports are they and what do they tell you?
	v. What host- or network-based indicators could be used to identify this malware on infected machines?
	vi. This file has one resource in the resource section. Use Resource Hacker to examine that resource, and then use it to extract the resource. What can you learn from the resource?
	e. Analyze the malware found in the file Lab03-01.exe using basic dynamic analysis tools.
	i. What are this malware's imports and strings?
	ii. What are the malware's host-based indicators?
	iii. Are there any useful network-based signatures for this malware? If so, what are they?
	f. Analyze the malware found in the file Lab03-02.dll using basic dynamic analysis tools.
	i. How can you get this malware to install itself?
	ii. How would you get this malware to run after installation?
	iii. How can you find the process under which this malware is running?
	iv. Which filters could you set in order to use procmon to glean information?
	v. What are the malware's host-based indicators?
	vi. Are there any useful network-based signatures for this malware?
	g. Execute the malware found in the file Lab03-03.exe while monitoring it using basic dynamic analysis tools in a safe environment
	i. What do you notice when monitoring this malware with Process Explorer?
	ii. Can you identify any live memory modifications?
	iii. What are the malware's host-based indicators?
	iv. What is the purpose of this program?
	h. Analyze the malware found in the file Lab03-04.exe using basic dynamic analysis tools.
	i. What happens when you run this file?
	ii. What is causing the roadblock in dynamic analysis?
	iii. Are there other ways to run this program?
2.	a. Analyze the malware found in the file Lab05-01.dll using only IDA Pro. The goal of this lab is to give you hands-on experience with IDA Pro. If you've already worked with IDA Pro, you may choose to ignore these questions and focus on reverse-engineering the malware.
	i. What is the address of DllMain?
	ii. Use the Imports window to browse to gethostbyname. Where is the import located?
	iii. How many functions call gethostbyname?
	iv. Focusing on the call to gethostbynamelocated at 0x10001757, can you figure out which DNS request will be made?
	v. How many local variables has IDA Pro recognized for the subroutine at 0x10001656?

	vi.	How many parameters has IDA Pro recognized for the subroutine at 0x10001656?
	vii.	Use the Strings window to locate the string \cmd.exe /cin the disassembly. Where is it located?
	viii.	What is happening in the area of code that references \cmd.exe/c?
	ix.	In the same area, at 0x100101C8, it looks like dword_1008E5C4 is a global variable that helps decide which path to take. How does the malware set dword_1008E5C4? (Hint: Use dword_1008E5C4's cross-references.)
	x.	A few hundred lines into the subroutine at 0x1000FF58, a series of comparisons use memcmpto compare strings. What happens if the string comparison to robotworkis successful (when memcmpreturns 0)?
	xi.	What does the export PSLISTdo?
	xii.	Use the graph mode to graph the cross-references from sub_10004E79. Which API functions could be called by entering this function? Based on the API functions alone, what could you rename this function?
	xiii.	How many Windows API functions does DllMaincall directly? How many at a depth of 2?
	xiv.	At 0x10001358, there is a call to Sleep (an API function that takes one parameter containing the number of milliseconds to sleep). Looking backward through the code, how long will the program sleep if this code executes?
	xv.	At 0x10001701 is a call to socket. What are the three parameters?
	xvi.	Using the MSDN page for socketand the named symbolic constants functionality in IDA Pro, can you make the parameters more meaningful? What are the parameters after you apply changes?
	xvii.	Search for usage of the in instruction (opcode 0xED). This instruction is used with a magic string VMXh to perform VMware detection. Is that in use in this malware? Using the cross-references to the function that executes the in instruction, is there further evidence of VMware detection?
	xviii.	Jump your cursor to 0x1001D988. What do you find?
	xix.	If you have the IDA Python plug-in installed (included with the com- mercial version of IDA Pro), run *Lab05-01.py*, an IDA Pro Python script provided with the malware for this book. (Make sure the cursor is at 0x1001D988.) What happens after you run the script?
	xx.	With the cursor in the same location, how do you turn this data into a single ASCII string?
	xxi.	Open the script with a text editor. How does it work?
	b. analyze the malware found in the file Lab06-01.exe.	
	i.	What is the major code construct found in the only subroutine called by main?
	ii.	What is the subroutine located at 0x40105F?
	iii.	What is the purpose of this program?
	c. Analyze the malware found in the file Lab06-02.exe.	
	i.	What operation does the first subroutine called by mainperform?
	ii.	What is the subroutine located at 0x40117F?
	iii.	What does the second subroutine called by maindo?

	iv. What type of code construct is used in this subroutine?
	v. Are there any network-based indicators for this program?
	vi. What is the purpose of this malware?
	d. analyze the malware found in the file Lab06-03.exe.
	i. Compare the calls in main to Lab 6-2's main method. What is the new function called from main?
	ii. What parameters does this new function take?
	iii. What major code construct does this function contain?
	iv. What can this function do?
	v. Are there any host-based indicators for this malware?
	vi. What is the purpose of this malware?
	e. analyze the malware found in the file Lab06-04.exe.
	i. What is the difference between the calls made from the main method in Labs 6-3 and 6-4?
	ii. What new code construct has been added to main?
	iii. What is the difference between this lab's parse HTML function and those of the previous labs?
	iv. How long will this program run? (Assume that it is connected to the Internet.)
	v. Are there any new network-based indicators for this malware?
	vi. What is the purpose of this malware?
3.	a. Analyze the malware found in the file Lab07-01.exe.
	i. How does this program ensure that it continues running (achieves persistence) when the computer is restarted?
	ii. Why does this program use a mutex?
	iii. What is a good host-based signature to use for detecting this program?
	iv. What is a good network-based signature for detecting this malware?
	v. What is the purpose of this program?
	vi. When will this program finish executing?
	b. Analyze the malware found in the file Lab07-02.exe.
	i. How does this program achieve persistence?
	ii. What is the purpose of this program?
	iii. When will this program finish executing?
	c. For this lab, we obtained the malicious executable, Lab07-03.exe, and DLL, Lab07-03.dll, prior to executing. This is important to note because the mal- ware might change once it runs. Both files were found in the same directory on the victim machine. If you run the program, you should ensure that both files are in the same directory on the analysis machine. A visible IP string beginning with 127 (a loopback address) connects to the local machine. (In the real version of this malware, this address connects to a remote machine, but we've set it to connect to localhost to protect you.)
	i. How does this program achieve persistence to ensure that it continues running when the computer is restarted?
	ii. What are two good host-based signatures for this malware?
	iii. What is the purpose of this program?

	iv. How could you remove this malware once it is installed?
	d. Analyze the malware found in the file Lab09-01.exe using OllyDbg and IDA Pro to answer the following questions. This malware was initially analyzed in the Chapter 3 labs using basic static and dynamic analysis techniques.
	i. How can you get this malware to install itself?
	ii. What are the command-line options for this program? What is the password requirement?
	iii. How can you use OllyDbg to permanently patch this malware, so that it doesn't require the special command-line password?
	iv. What are the host-based indicators of this malware?
	v. What are the different actions this malware can be instructed to take via the network?
	vi. Are there any useful network-based signatures for this malware?
	e. Analyze the malware found in the file Lab09-02.exe using OllyDbg to answer the following questions.
	i. What strings do you see statically in the binary?
	ii. What happens when you run this binary?
	iii. How can you get this sample to run its malicious payload?
	iv. What is happening at 0x00401133?
	v. What arguments are being passed to subroutine 0x00401089?
	vi. What domain name does this malware use?
	vii. What encoding routine is being used to obfuscate the domain name?
	viii. What is the significance of the CreateProcessA call at 0x0040106E?
	f. Analyze the malware found in the file Lab09-03.exe using OllyDbg and IDA Pro. This malware loads three included DLLs (DLL1.dll, DLL2.dll, and DLL3.dll) that are all built to request the same memory load location. Therefore, when viewing these DLLs in OllyDbg versus IDA Pro, code may appear at different memory locations. The purpose of this lab is to make you comfortable with finding the correct location of code within IDA Pro when you are looking at code in OllyDbg
	i. What DLLs are imported by *Lab09-03.exe*?
	ii. What is the base address requested by *DLL1.dll*, *DLL2.dll*, and *DLL3.dll*?
	iii. When you use OllyDbg to debug *Lab09-03.exe*, what is the assigned based address for: *DLL1.dll*, *DLL2.dll*, and *DLL3.dll*?
	iv. When *Lab09-03.exe* calls an import function from *DLL1.dll*, what does this import function do?
	v. When *Lab09-03.exe* calls WriteFile, what is the filename it writes to?
	vi. When *Lab09-03.exe* creates a job using NetScheduleJobAdd, where does it get the data for the second parameter?
	vii. While running or debugging the program, you will see that it prints out three pieces of mystery data. What are the following: DLL 1 mystery data 1, DLL 2 mystery data 2, and DLL 3 mystery data 3?
	viii. How can you load *DLL2.dll* into IDA Pro so that it matches the load address used by OllyDbg?

4.	a. This lab includes both a driver and an executable. You can run the executable from anywhere, but in order for the program to work properly, the driver must be placed in the C:\Windows\ System32 directory where it was origi- nally found on the victim computer. The executable is Lab10-01.exe, and the driver is Lab10-01.sys.
	i. Does this program make any direct changes to the registry? (Use procmon to check.)
	ii. The user-space program calls the ControlService function. Can you set a breakpoint with WinDbg to see what is executed in the kernel as a result of the call to ControlService?
	iii. What does this program do?
	b. The file for this lab is Lab10-02.exe.
	i. Does this program create any files? If so, what are they?
	ii. Does this program have a kernel component?
	iii. What does this program do?
	c. This lab includes a driver and an executable. You can run the executable from anywhere, but in order for the program to work properly, the driver must be placed in the C:\Windows\System32 directory where it was originally found on the victim computer. The executable is Lab10-03.exe, and the driver is Lab10-03.sys.
	i. What does this program do?
	ii. Once this program is running, how do you stop it?
	iii. What does the kernel component do?
5.	a. Analyze the malware found in Lab11-01.exe
	i. What does the malware drop to disk?
	ii. How does the malware achieve persistence?
	iii. How does the malware steal user credentials?
	iv. What does the malware do with stolen credentials?
	v. How can you use this malware to get user credentials from your test environment?
	b. Analyze the malware found in *Lab11-02.dll*. Assume that a suspicious file named *Lab11-02.ini* was also found with this malware.
	i. What are the exports for this DLL malware?
	ii. What happens after you attempt to install this malware using
	iii. *rundll32.exe*?
	iv. Where must *Lab11-02.ini* reside in order for the malware to install properly?
	v. How is this malware installed for persistence?
	vi. What user-space rootkit technique does this malware employ?
	vii. What does the hooking code do?
	viii. Which process(es) does this malware attack and why?
	ix. What is the significance of the *.ini* file?
	c. Analyze the malware found in *Lab11-03.exe* and *Lab11-03.dll*. Make sure that both files are in the same directory during analysis
	i. What interesting analysis leads can you discover using basic static analysis?
	ii. What happens when you run this malware?
	iii. How does *Lab11-03.exe* persistently install *Lab11-03.dll*?

	iv. Which Windows system file does the malware infect?
	v. What does *Lab11-03.dll* do?
	vi. Where does the malware store the data it collects?
6.	a. Analyze the malware found in the file *Lab12-01.exe* and *Lab12-01.dll*. Make sure that these files are in the same directory when performing the analysis.
	i. What happens when you run the malware executable?
	ii. What process is being injected?
	iii. How can you make the malware stop the pop-ups?
	iv. How does this malware operate?
	b. Analyze the malware found in the file *Lab12-02.exe*.
	i. What is the purpose of this program?
	ii. How does the launcher program hide execution?
	iii. Where is the malicious payload stored?
	iv. How is the malicious payload protected?
	v. How are strings protected?
	c. Analyze the malware extracted during the analysis of Lab 12-2, or use the file *Lab12-03.exe*.
	i. What is the purpose of this malicious payload?
	ii. How does the malicious payload inject itself?
	iii. What filesystem residue does this program create?
	d. Analyze the malware found in the file *Lab12-04.exe*.
	i. What does the code at 0x401000 accomplish?
	ii. Which process has code injected?
	iii. What DLL is loaded using LoadLibraryA?
	iv. What is the fourth argument passed to the CreateRemoteThread call?
	v. What malware is dropped by the main executable?
7.	a. Analyze the malware found in the file *Lab13-01.exe*.
	i. Compare the strings in the malware (from the output of the strings command) with the information available via dynamic analysis. Based on this comparison, which elements might be encoded?
	ii. Use IDA Pro to look for potential encoding by searching for the string xor. What type of encoding do you find?
	iii. What is the key used for encoding and what content does it encode?
	iv. Use the static tools FindCrypt2, Krypto ANALyzer (KANAL), and the IDA Entropy Plugin to identify any other encoding mechanisms. What do you find?
	v. What type of encoding is used for a portion of the network traffic sent by the malware?
	vi. Where is the Base64 function in the disassembly?
	vii. What is the maximum length of the Base64-encoded data that is sent? What is encoded?
	viii. In this malware, would you ever see the padding characters (= or ==) in the Base64-encoded data?
	ix. What does this malware do?
	b. Analyze the malware found in the file *Lab13-02.exe*.

	i. Using dynamic analysis, determine what this malware creates.
	ii. Use static techniques such as an xor search, FindCrypt2, KANAL, and the IDA Entropy Plugin to look for potential encoding. What do you find?
	iii. Based on your answer to question 1, which imported function would be a good prospect for finding the encoding functions?
	iv. Where is the encoding function in the disassembly?
	v. Trace from the encoding function to the source of the encoded content. What is the content?
	vi. Can you find the algorithm used for encoding? If not, how can you decode the content?
	vii. Using instrumentation, can you recover the original source of one of the encoded files?
	c. Analyze the malware found in the file *Lab13-03.exe*.
	i. Compare the output of strings with the information available via dynamic analysis. Based on this comparison, which elements might be encoded?
	ii. Use static analysis to look for potential encoding by searching for the string xor. What type of encoding do you find?
	iii. Use static tools like FindCrypt2, KANAL, and the IDA Entropy Plugin to identify any other encoding mechanisms. How do these findings com- pare with the XOR findings?
	iv. Which two encoding techniques are used in this malware?
	v. For each encoding technique, what is the key?
	vi. For the cryptographic encryption algorithm, is the key sufficient? What else must be known?
	vii. What does this malware do?
	viii. Create code to decrypt some of the content produced during dynamic analysis. What is this content?
8.	a. Analyze the malware found in file *Lab14-01.exe*. This program is not harmful to your system.
	i. Which networking libraries does the malware use, and what are their advantages?
	ii. What source elements are used to construct the networking beacon, and what conditions would cause the beacon to change?
	iii. Why might the information embedded in the networking beacon be of interest to the attacker?
	iv. Does the malware use standard Base64 encoding? If not, how is the encoding unusual?
	v. What is the overall purpose of this malware?
	vi. What elements of the malware's communication may be effectively detected using a network signature?
	vii. What mistakes might analysts make in trying to develop a signature for this malware?
	viii. What set of signatures would detect this malware (and future variants)?
	b. Analyze the malware found in file *Lab14-02.exe*. This malware has been configured to beacon to a hard-coded loopback address in order to prevent it from harming your system, but imagine that it is a hard-coded external address.

	i. What are the advantages or disadvantages of coding malware to use direct IP addresses?
	ii. Which networking libraries does this malware use? What are the advantages or disadvantages of using these libraries?
	iii. What is the source of the URL that the malware uses for beaconing? What advantages does this source offer?
	iv. Which aspect of the HTTP protocol does the malware leverage to achieve its objectives?
	v. What kind of information is communicated in the malware's initial beacon?
	vi. What are some disadvantages in the design of this malware's communication channels?
	vii. Is the malware's encoding scheme standard?
	viii. How is communication terminated?
	ix. What is the purpose of this malware, and what role might it play in the attacker's arsenal?
	c. This lab builds on Practical 8 a. Imagine that this malware is an attempt by the attacker to improve his techniques. Analyze the malware found in file *Lab14-03.exe*.
	i. What hard-coded elements are used in the initial beacon? What elements, if any, would make a good signature?
	ii. What elements of the initial beacon may not be conducive to a longlasting signature?
	iii. How does the malware obtain commands? What example from the chapter used a similar methodology? What are the advantages of this technique?
	iv. When the malware receives input, what checks are performed on the input to determine whether it is a valid command? How does the attacker hide the list of commands the malware is searching for?
	v. What type of encoding is used for command arguments? How is it different from Base64, and what advantages or disadvantages does it offer?
	vi. What commands are available to this malware?
	vii. What is the purpose of this malware?
	viii. This chapter introduced the idea of targeting different areas of code with independent signatures (where possible) in order to add resiliency to network indicators. What are some distinct areas of code or configuration data that can be targeted by network signatures?
	ix. What set of signatures should be used for this malware?
	d. Analyze the sample found in the file Lab15-01.exe. This is a command-line program that takes an argument and prints "Good Job!" if the argument matches a secret code.
	i. What anti-disassembly technique is used in this binary?
	ii. What rogue opcode is the disassembly tricked into disassembling?
	iii. How many times is this technique used?
	iv. What command-line argument will cause the program to print "Good Job!"?
	e. Analyze the malware found in the file Lab15-02.exe. Correct all anti-disassembly countermeasures before analyzing the binary in order to answer the questions.
	i. What URL is initially requested by the program?
	ii. How is the User-Agent generated?

	iii. What does the program look for in the page it initially requests?
	iv. What does the program do with the information it extracts from the page?
	f. Analyze the malware found in the file *Lab15-03.exe*. At first glance, this binary appears to be a legitimate tool, but it actually contains more functionality than advertised.
	i. How is the malicious code initially called?
	ii. What does the malicious code do?
	iii. What URL does the malware use?
	iv. What filename does the malware use?
9.	a. Analyze the malware found in *Lab16-01.exe* using a debugger. This is the same malware as *Lab09-01.exe*, with added anti-debugging techniques.
	i. Which anti-debugging techniques does this malware employ?
	ii. What happens when each anti-debugging technique succeeds?
	iii. How can you get around these anti-debugging techniques?
	iv. How do you manually change the structures checked during runtime?
	v. Which OllyDbg plug-in will protect you from the anti-debugging techniques used by this malware?
	b. Analyze the malware found in *Lab16-02.exe* using a debugger. The goal of this lab is to figure out the correct password. The malware does not drop a malicious payload.
	i. What happens when you run *Lab16-02.exe* from the command line?
	ii. What happens when you run *Lab16-02.exe* and guess the command-line parameter?
	iii. What is the command-line password?
	iv. Load *Lab16-02.exe* into IDA Pro. Where in the mainfunction is strncmp
	v. found?
	vi. What happens when you load this malware into OllyDbg using the default settings?
	vii. What is unique about the PE structure of *Lab16-02.exe*?
	viii. Where is the callback located? (Hint: Use CTRL-E in IDA Pro.)
	ix. Which anti-debugging technique is the program using to terminate immediately in the debugger and how can you avoid this check?
	x. What is the command-line password you see in the debugger after you disable the anti-debugging technique?
	xi. Does the password found in the debugger work on the command line?
	c. Analyze the malware in *Lab16-03.exe* using a debugger. This malware is similar to *Lab09-02.exe*, with certain modifications, including the introduction of anti-debugging techniques.
	i. Which strings do you see when using static analysis on the binary?
	ii. What happens when you run this binary?
	iii. How must you rename the sample in order for it to run properly?
	iv. Which anti-debugging techniques does this malware employ?
	v. For each technique, what does the malware do if it determines it is running in a debugger?
	vi. Why are the anti-debugging techniques successful in this malware?

	vii. What domain name does this malware use?
	d. Analyze the malware found in *Lab17-01.exe* inside VMware. This is the same malware as *Lab07-01.exe*, with added anti-VMware techniques.
	i. What anti-VM techniques does this malware use?
	ii. If you have the commercial version of IDA Pro, run the IDA Python script from Listing 17-4 in Chapter 17 (provided here as *findAntiVM.py*). What does it find?
	iii. What happens when each anti-VM technique succeeds?
	iv. Which of these anti-VM techniques work against your virtual machine?
	v. Why does each anti-VM technique work or fail?
	vi. How could you disable these anti-VM techniques and get the malware to run?
	e. Analyze the malware found in the file *Lab17-02.dll* inside VMware. After answering the first question in this lab, try to run the installation exports using *rundll32.exe* and monitor them with a tool like procmon. The following is an example command line for executing the DLL: rundll32.exe Lab17-02.dll,InstallRT (or InstallSA/InstallSB)
	i. What are the exports for this DLL?
	ii. What happens after the attempted installation using *rundll32.exe*?
	iii. Which files are created and what do they contain?
	iv. What method of anti-VM is in use?
	v. How could you force the malware to install during runtime?
	vi. How could you permanently disable the anti-VM technique?
	vii. How does each installation export function work?
	f. Analyze the malware *Lab17-03.exe* inside VMware.
	i. What happens when you run this malware in a virtual machine?
	ii. How could you get this malware to run and drop its keylogger?
	iii. Which anti-VM techniques does this malware use?
	iv. What system changes could you make to permanently avoid the anti-VM techniques used by this malware?
	v. How could you patch the binary in OllyDbg to force the anti-VM techniques to permanently fail?
10.	a. Analyze the file *Lab19-01.bin* using *shellcode_launcher.exe*
	i. How is the shellcode encoded?
	ii. Which functions does the shellcode manually import?
	iii. What network host does the shellcode communicate with?
	iv. What filesystem residue does the shellcode leave?
	v. What does the shellcode do?
	b. The file *Lab19-02.exe* contains a piece of shellcode that will be injected into another process and run. Analyze this file.
	i. What process is injected with the shellcode?
	ii. Where is the shellcode located?
	iii. How is the shellcode encoded?

	iv. Which functions does the shellcode manually import?
	v. What network hosts does the shellcode communicate with?
	vi. What does the shellcode do?
	c. Analyze the file *Lab19-03.pdf*. If you get stuck and can't find the shellcode, just skip that part of the lab and analyze file *Lab19-03_sc.bin* using *shellcode_launcher.exe*.
	i. What exploit is used in this PDF?
	ii. How is the shellcode encoded?
	iii. Which functions does the shellcode manually import?
	iv. What filesystem residue does the shellcode leave?
	v. What does the shellcode do?
	d. The purpose of this first lab is to demonstrate the usage of the this pointer. Analyze the malware in *Lab20-01.exe*.
	i. Does the function at 0x401040 take any parameters?
	ii. Which URL is used in the call to URLDownloadToFile?
	iii. What does this program do?
	e. Analyze the malware In Lab20-02.exe.
	i. What can you learn from the interesting strings in this program?
	ii. What do the imports tell you about this program?
	iii. What is the purpose of the object created at 0x4011D9? Does it have any virtual functions?
	iv. Which functions could possibly be called by the call [edx] instruction at 0x401349?
	v. How could you easily set up the server that this malware expects in order to fully analyze the malware without connecting it to the Internet?
	vi. What is the purpose of this program?
	vii. What is the purpose of implementing a virtual function call in this program?
	f. Analyze the malware in Lab20-03.exe.
	i. What can you learn from the interesting strings in this program?
	ii. What do the imports tell you about this program?
	iii. At 0x4036F0, there is a function call that takes the string Config error, followed a few instructions later by a call to CxxThrowException. Does the function take any parameters other than the string? Does the function return anything? What can you tell about this function from the context in which it's used?
	iv. What do the six entries in the switch table at 0x4025C8 do?
	v. What is the purpose of this program?
	g. Analyze the code in *Lab21-01.exe*
	i. What happens when you run this program without any parameters?
	ii. Depending on your version of IDA Pro, main may not be recognized automatically. How can you identify the call to the main function?
	iii. What is being stored on the stack in the instructions from 0x0000000140001150 to 0x0000000140001161?

	iv. How can you get this program to run its payload without changing the filename of the executable?
	v. Which two strings are being compared by the call to strncmp at 0x0000000140001205?
	vi. Does the function at 0x00000001400013C8 take any parameters?
	vii. How many arguments are passed to the call to CreateProcess at 0x0000000140001093? How do you know?
	h. Analyze the malware found in *Lab21-02.exe* on both x86 and x64 virtual machines.
	i. What is interesting about the malware's resource sections?
	ii. Is this malware compiled for x64 or x86?
	iii. How does the malware determine the type of environment in which it is running?
	iv. What does this malware do differently in an x64 environment versus an x86 environment?
	v. Which files does the malware drop when running on an x86 machine? Where would you find the file or files?
	vi. Which files does the malware drop when running on an x64 machine? Where would you find the file or files?
	vii. What type of process does the malware launch when run on an x64 system?
	viii. What does the malware do?

M. Sc (Information Technology)		**Semester – III**	
Course Name: Robotic Process Automation Practical		**Course Code: PSIT3P4a**	
Periods per week (1 Period is 60 minutes)		4	
Credits		2	
		Hours	**Marks**
Evaluation System	**Practical Examination**	2	50
	Internal	--	-

List of Practical:	
1.	a. Create a simple sequence based project.
	b. Create a flowchart-based project.
	c. Create an UiPath Robot which can empty a folder in Gmail solely on basis of recording.
2.	a. Automate UiPath Number Calculation (Subtraction, Multiplication, Division of numbers).
	b. Create an automation UiPath project using different types of variables (number, datetime, Boolean, generic, array, data table)
3.	a. Create an automation UiPath Project using decision statements.
	b. Create an automation UiPath Project using looping statements.
4.	a. Automate any process using basic recording.
	b. Automate any process using desktop recording.
	c. Automate any process using web recording.

5.	a. Consider an array of names. We have to find out how many of them start with the letter "a". Create an automation where the number of names starting with "a" is counted and the result is displayed.
6.	a. Create an application automating the read, write and append operation on excel file.
	b. Automate the process to extract data from an excel file into a data table and vice versa
7.	a. Implement the attach window activity.
	b. Find different controls using UiPath.
	c. Demonstrate the following activities in UiPath: i. Mouse (click, double click and hover) ii. Type into iii. Type Secure text
8.	a. Demonstrate the following events in UiPath: i. Element triggering event ii. Image triggering event iii. System Triggering Event
	b. Automate the following screen scraping methods using UiPath i. Full Test ii. Native iii. OCR
	c. Install and automate any process using UiPath with the following plug-ins: i. Java Plugin ii. Mail Plugin iii. PDF Plugin iv. Web Integration v. Excel Plugin vi. Word Plugin vii. Credential Management
9.	a. Automate the process of send mail event (on any email).
	b. Automate the process of launching an assistant bot on a keyboard event.
	c. Demonstrate the Exception handing in UiPath.
	d. Demonstrate the use of config files in UiPath.
10.	a. Automate the process of logging and taking screenshots in UiPath.
	b. Automate any process using State Machine in UiPath.
	c. Demonstrate the use of publish utility.
	d. Create and provision Robot using Orchestrator.

M. Sc (Information Technology)		Semester – III	
Course Name: Virtual Reality and Augmented Reality Practical		**Course Code: PSIT3P4b**	
Periods per week (1 Period is 60 minutes)		4	
Credits		2	
		Hours	**Marks**
Evaluation System	**Practical Examination**	2	50
	Internal	--	-

M. Sc (Information Technology)		Semester – III	
Course Name: Data Centre Technologies Practical		**Course Code: PSIT3P4c**	
Periods per week (1 Period is 60 minutes)		4	
Credits		2	
		Hours	**Marks**
Evaluation System	**Practical Examination**	2	50
	Internal	--	-

List of Practical:	
1.	**Configuring ESXi Hosts**
	a. Install ESXi on a VM using your student desktop
	b. Install ESXi
2.	**Configuring ESXi Hosts**
	a. Examine the Options in the DCUI
	b. Configure the Management Network
	c. Enable SSH
3.	**Deploying and Configuring a Virtual Machine**
	a. Create a Virtual Machine
	b. Install a Guest Operating System and Disable Windows Updates
	c. Install VMware Tools/Install Files
4.	**Working with vCenter Server**
5.	**Navigating the vSphere Clients**
6.	**Creating Folders in vCenter Server Appliance**
7.	**Using Standard Switches**
8.	**Accessing iSCSI Storage**
a.	Managing VMFS Datastores
b.	Accessing NFS Storage

9.	**Using Templates and Clones**
10.	**Modifying Virtual Machines**
	a. Migrating Virtual Machines
	b. Managing Virtual Machines

M. Sc (Information Technology)		**Semester – III**	
Course Name: Offensive Security Practical		**Course Code: PSIT3P4d**	
Periods per week (1 Period is 60 minutes)		4	
Credits		2	
		Hours	**Marks**
Evaluation System	**Practical Examination**	2	50
	Internal	--	-

List of Practical: to be performed with Kali Linux and Meta-Sploit Framework)	
0.	Installation and preparing the lab ready Virtual or physical machine with Kali Linux. Exploring and getting acquainted with he other operating distributions used for offensive security testing mainly • Lion Sec • BackBox • Parrot • BlackArch
1.	Exploring the command line arguments
a.	Environment Variables , Tab Completion , Bash History Tricks
b.	Piping and Redirection, Text Searching and Manipulation
c.	Editing Files from the Command Line, Comparing Files, Managing Processes
2.	
a.	Using NETCAT Socat
b.	PowerShell and Powercat
c.	Wireshark and Tcpdump
3.	Passive Information Gathering
a.	Whois Enumeration/ Google Hacking
b.	Netcraft, Recon-ng, Shodan
c	SSL Server Test
4.	User Information Gathering
a.	Email Harvesting, Password Dumps
b.	Information Gathering Frameworks- OSINT Framework, Maltego
5.	Active Information Gathering

a.	DNS Enumeration
b.	Port Scanning
	SMB Enumeration
	NFS Enumeration
6.	Vulnerability Scanning
a.	Vulnerability Scanning with Nessus
b.	Vulnerability Scanning with Nmap
7.	Web Application Assessment Tools
a.	DIRB
b.	Burp Suite
c.	Nikto
d.	SQL Injection
8.	Client-Side Attacks
c.	HTA Attack
d.	Exploiting Microsoft Office
9.	Privilege Escalation
a.	Windows Privilege Escalation
b.	Linux Privilege Escalation
10.	Password Attacks
a.	Wordlists, Brute Force Wordlists
b.	Common Network Service Attack Methods
11.	Port Redirection and Tunneling
a.	Port Forwarding- RINETD
b.	SSH Tunneling
c.	PLINK., NETSH , HTTPTunnel-ing Through Deep Packet Inspection