**THEORETICAL ASSIGNMENT**

**SE-2**

**Section No: ………….………**

**Group No: ………….………**

**Members name:**

**………..…………….……**

**………...…………………**

**……………...…………….**

**…………………………….**

**…………………………….**

**…………………………….**

**Instructions:**

* Total marks for this assignment are 10.
* Last Date for Submission of Assignment is **04 April 2020**.
* Any Assignment submitted **LATE**, will be marked as **ZERO**.
* Any assignment found **copied,** will be marked as **ZERO**.
* Assignment should be submitted **ONLINE only**.
* **Proper reference** should be given at the end of Answer.

1. Explain why distributed software systems are more complex than centralized software systems, where all the system functionality is implemented on a single computer.

Distributed software systems are more complex than centralized software systems that run on a single computer. This complexity of distributed software systems arises because,

• It is practically impossible to have a top-down model of control of these systems.

• The nodes in the system that deliver functionality are often independent systems with no single authority in charge of them.

• The network connecting these nodes is a separately managed system.

• It is a complex system in its own right and cannot be controlled by the owners of systems using the network.

• There is therefore an inherent unpredictability in the operation of distributed systems that has to be taken into account by the system designer.

2. You have been asked to design a secure system that requires strong authentication and authorization. The system must be designed so that communications between parts of the system cannot be intercepted and read by an attacker. Suggest the most appropriate client–server architecture for this system and, giving reasons for your answer, propose how functionality should be distributed between the client and the server systems.

When a system is proposed which requires strong authentication and authorization, two-tier client server architecture with fat clients is effective. In this model some of the application processing is carried out on the client and the data management and database functions are implemented on the servers. Data management is carried out straightforward as there is no need to manage the interaction between the client and application processing system.  
  
Consider an example of bank ATM system, the ATM is the client computer and the server is a mainframe which runs the customer account database. The communication between ATM and server (database) is not direct rather a middleware like teleprocessing monitor is used which organizes communication with remote clients and serializes client transactions for processing by the database. This ensures that transactions are independent and do not interfere with one other and hence attacker cannot intercept the transactions. The system can recover from faults without corrupting the system data because of serial transactions.

3. Explain why deploying software as a service can reduce the IT support costs for a company. What additional costs might arise if this deployment model is used?

Deploying software as a service has the potential for reducing the IT support costs as there is no need to install and support separate software on each client. Rather, all software is hosted on a server and when e.g. upgrades are required, only the server (or servers) need be upgraded. There are no support problems with different computers in an organization running different software versions. General help support is provided by the service provider rather than the local IT staff.

**The additional costs that can arise from this model are:**

1. Network costs, as obviously there is a considerable increase in network traffic. Service providers (such as Amazon) may charge for data uploads and downloads. This is only applicable of the service is provided by a 3rd party rather than in-house.

2. Server costs, as the servers are responsible for all computation and so must either be more powerful or more numerous. This is most significant if the service is provided in-house.

3. There may in fact be additional support costs from this model in the short term if it requires users to change the software that they normally use. This is likely to lead to additional demands for help.

4. Your company wishes to move from using desktop applications to accessing the same functionality remotely as services. Identify three risks that might arise and suggest how these risks may be reduced.

**1. Redundancy:**

If server or system of that runs the remote service goes down the ability for companies to continue to function is a must but when have a backup system is with minimal functionality or a desktop can be used until system make back up.

**2.Connectivity**

If software will be remotely used the use of software may be no longer in companies' control. Which completely in hands of their internet provider as which is speed that they can access data.

**3. Accessibility:**

If the application has any large calculations which it performs, then these features must be implemented with great caution when put into a networked service. After all, if a client (or number of clients) connects to the service and ask it to run its most computationally intensive functionality, the service must be able to juggle its responsibilities such that no single client monopolizes the service; and that new users will be able to join and still have their user experiences unharmed.

To solve this, the application must be converted to utilize a task manager with hierarchies of tasks. This way, core communications can be given priority across all sessions and all users, while the expensive operations can be relegated to be given proportional distribution of the available remaining calculation space. Another option might be to make for 'fat' clients; where these heavy calculations are given to the client rather than the server; but this solution might not be available depending on the specifications of the project, and comes with its own issues.

**4. Security:**

The most significant feature which a local application does not have to account for is security; after all, if you're logged into the computer and able to open the app, you're probably supposed to be there. However, networked services have no such protection; anyone in the network (which could likely be the global web) may interface with the service. This means that security becomes a significant issue; by having every application user's information stored and computed within this networked service, it creates a significant potential for a malicious entity to intercept, view, or alter this data; with potentially ruinous effects on the individuals under attack. Therefore, remote services must have significant and extensive security in place, to protect the confidentiality and integrity of its processes.

The only way to solve security is to take it seriously. Most likely, all aspects of validation, verification, and communication between client and server will have to be redesigned in order to take advantage of modern network security frameworks; luckily, many of these frameworks are becoming increasingly well-designed and easy to implement, making security approachable as a requirement, despite never being truly solvable.