Narasaraopeta Engineering College (Autonomous)

Kotappakonda Road, Yellamanda (P.0), Narasaraopet- 522601, Guntur District, AP.

Subject Code: R1GCC41OE9

IV B.Tech I Semester Regular & Supple Examinations, January-2022

CLOUD COMPUTING (OPEN ELECTIVE—II)(CSE)

Time: 3 hours Max Marks: 60

Question Paper Consists of Part-A and Part-B.

Answering the question in Part-A is Compulsory &Four Questions should be answered from Part-B

All questions carry equal marks of 12.

PART-A

1. (a)Define network virtualization ?. [CO1,K1, 2M]

Network Virtualization is the process of combining hardware and software network resources and network functionality into a single, software-based administrative entity, a virtual network. Network virtualization involves platform virtualization, often combined with resource virtualization.

(b)what is cloud computing ? [CO2,K1, 2M]

The cloud is a large group of interconnected computers. These computers can be personal computers or network servers; they can be public or private.

For example, Google hosts a cloud that consists of both smallish PCs and larger servers. Google's cloud is a private one (that is, Google owns it) that is publicly accessible (by Google's users).

(c) What is a data center ?. [CO3,K1, 2M]

(d) what is disaster Recovery. [CO5,K1, 2M]

Disaster recovery is the practice of making a system capable of surviving unexpected or extraordinary failures. A disaster recovery plan, for example, will help your IT systems survive a fire in your data center that destroys all of the servers in that data center and the systems they support. Every organization should have a documented disaster recovery process and should test that process at least twice each year

(e) what is IaaS. [CO4,K1, 2M]

Infrastructure-as-a-service is really data center-as-a-service or the ability to access computing resources remotely. In essence, you lease a physical server that is yours to do with what you will and that for all practical purposes is your data center, or at least part of a data center. The difference with this approach versus more mainstream cloud computing is that instead of using an interface and a metered service, you get access to the entire machine and the software on that machine. In short, it is less packaged

(f) List types of cloud. [CO1,K1, 2M]

The three main types of Cloud Computing are:
A) Public Cloud
B) Private Cloud
C) Hybrid Cloud
D) Community Cloud

(2+2+2+2+2+2]

PART-B 4 X 12 =48

2. (a)Explain how service oriented Architecture (SOA) is supported in cloud computing [CO1,K1, 2M]

Overview of SOA
Services are natural building blocks allowing to organize capabilities naturally, similar to objects and components SOA consists of a service provider and service consumer that requested a service. Loose coupling is closely associated with SOA
Its benefits are: flexibility, scalability, replacability and fault tolerance
SOA Design and Development

- Identify different units of business logic and work units
- Explain functionality of various units in terms of services
- Identify core infrastructure services
- Identify major links of common functionality between different services
- Capture service functionalities in terms of services
- Define events of interest in the service
- Create workflows to enable service choreography
- Publish services in a registry or multiple registries

v From the point of view of the business:
SOA is a set of services that are configured to form composite applications with dynamic and flexible process flows. Those processes and services can be exposed to customers and partners, or to other parts of the organization
v From the point of view of an enterprise architect:
SOA is an architectural style that promotes the concepts of business processes and the orchestration of enterprise-level business services. It is also a set of architectural principles, patterns and criteria which address characteristics such as modularity, encapsulation, loose coupling, separation-of-concerns, reuse and composability.
v From the point of view of a project manager
SOA is a development approach supporting highly productive parallel development
v From the point of view of a tester and/or quality assurance engineer

SOA represents a way to simplify overall system testing.

v From the point of view of a software developer

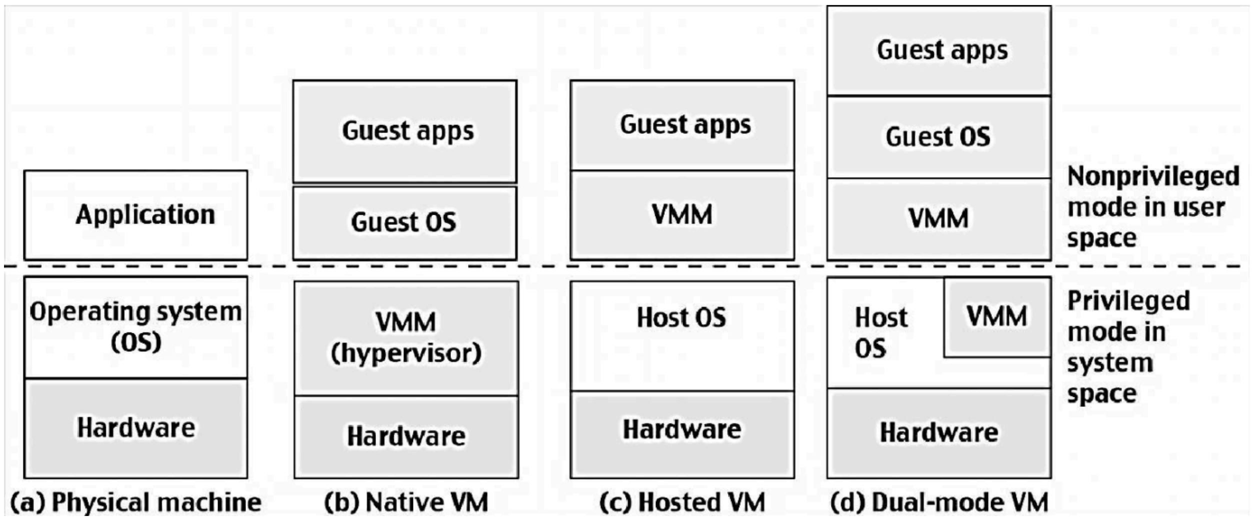
SOA is a programming model supported by standards, tools and technologies including, but not limited to Web Services.

(b) what virtual machine ? Discuss about virtulazion Middleware [[CO1,K2, 6M]

A conventional computer has a single OS (Operating System) image. Single OS image offers a rigid architecture that tightly couples application software to a specific hardware platform.

Some software running well on one machine may not be executable on another platform with a different instruction set under a fixed OS.

The below diagram shows the comparison between physical machine and three VM architectures



In **FIG-A: -** X-86 architecture desktop running its installed windows OS.
FIG-B: - shows a native VM installed with the use of VMM called Hypervisor in privileged mode.

Virtual machines (VM's) offer solutions to

1. Underutilized resources,
2. Application inflexibility,
3. Software manageability
4. Security concerns in existing physical machines.

Hypervisor approach is also called Bare-Metal VM because the hypervisor handle bare hardware (CPU, Memory, and I/O) devices directly. The VM can be provisioned for any hardware system, the VM is built with virtual resources managed by guest OS to run a specific application.

The VM is built with virtual resources managed by guest OS to run a specific application. Between VM's and host platform one need s to deploy a middleware layer called VMM (Virtual Machine Monitor).

In FIG-C, VMM runs in non-privileged mode. The host OS need not be modified.

In FIG-D, the VM is implemented in dual mode. Part of VMM runs at user level and another part runs at supervisor level.

In this case the host OS may have to be modified to some extent. The VM approach offers hardware independence of OS and applications. The user application running on its dedicated OS could be bundled together as a virtual appliance that can be ported to any hardware platform

VM Primitive Operations

The VMM provides VM abstraction to the guest OS. With full virtualization, the VMM exports a VM abstraction identical to physical machine so that a standard OS (windows 2000 or Linux) can run just as it would on the physical hardware.
The following are the low level VMM operations

1. Multiplexing
2. Suspension
3. Provision
4. Life migration.

First VM's can be multiplexed between hardware machines
Second a VM can be suspended and stored in stable storage. Third a suspended VM can be resumed or provisioned to a new hardware platform.

Finally a VM can be migrated from one hardware platform to another. These VM operations enable a VM to be provisioned to any available hardware platforms.

The VM approach will significantly enhance the utilization of server resources. Multiple server functions can be consolidated on the same hardware platform to achieve higher system efficiency.

3. (a) Briefly discuss about cloud infrastructure models [CO2,K2, 6M]

Before we move into building systems in the cloud, we should take a moment to understand a variety of cloud infrastructure models. The models are Infrastructure as a **Service (IaaS) to Platform as a Service (PaaS).**

Platform As a Service:

PaaS environments provide you with an infrastructure as well as complete operational and development environments for the deployment of your applications.

The most commonly used example of pure PaaS is Google App Engine. To leverage Google App Engine, you write your applications in Python against Google's development frameworks with tools for using the Google file system and data repositories. This approach works well for applications that must be deployed rapidly and don't have significant integration requirements.

The downside to the PaaS approach is vendor lock-in. With Google, for example, you must write your applications in the Python programming language to Google-specific APIs. Python is a wonderful programming language—but it isn't a core competency of most development teams

Infrastructure As a Service:

An Overview of Amazon Web Services (AWS):

AWS is Amazon's umbrella description of all of their web-based technology services. It encompasses a wide variety of services, all of which fall into the concept of cloud computing. Different types of Amazon Web Services:

• Amazon Elastic Cloud Compute (Amazon EC2)

• Amazon Simple Storage Service (Amazon S3)

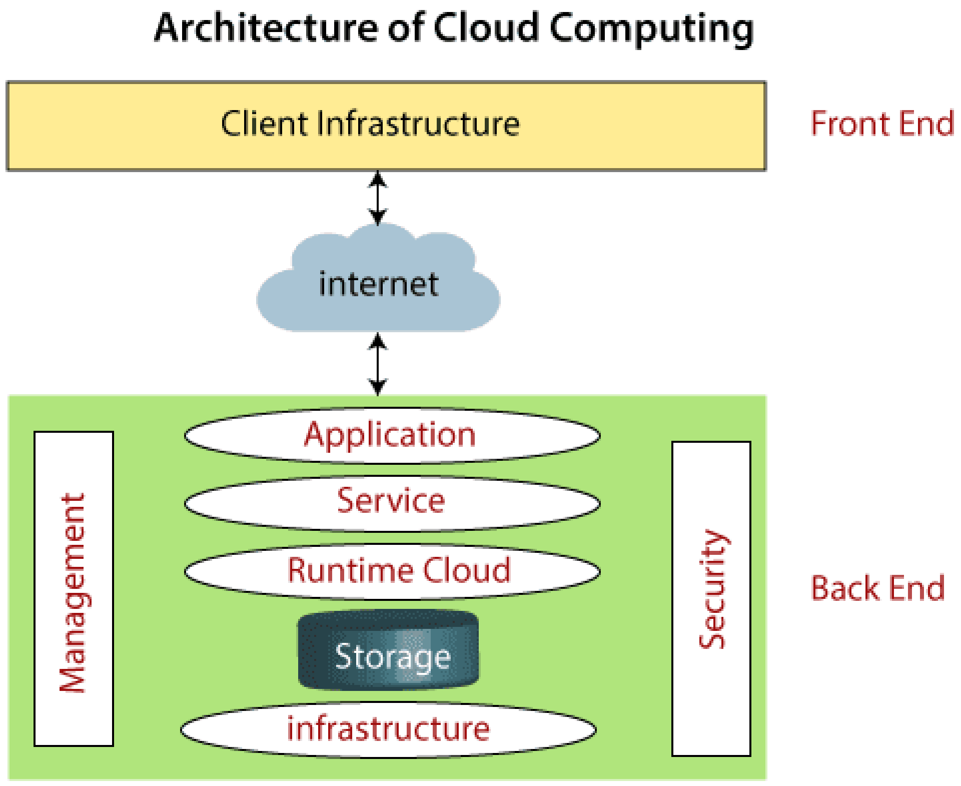
• Amazon Simple Queue Service (Amazon SQS)

• Amazon CloudFront

• Amazon SimpleDB

(b) Explain about cloud application architecture . [CO2,K2, 6M]

Cloud computing architecture is divided into the following two parts -
Front End
Back End
The below diagram shows the architecture of cloud computing –



Front End

The front end is used by the client. It contains client-side interfaces and applications that are required to access the cloud computing platforms. The front end includes web servers (including Chrome, Firefox, internet explorer, etc.), thin & fat clients, tablets, and mobile devices.

Back End

The back end is used by the service provider. It manages all the resources that are required to provide cloud computing services. It includes a huge amount of data storage, security mechanism, virtual machines, deploying models, servers, traffic control mechanisms, etc.

Components of Cloud Computing Architecture

There are the following components of cloud computing architecture -

1. Client Infrastructure
2. **Application**
3. **Service**
4. **Runtime Cloud**
5. Storage
6. **Infrastructure**
7. **Management**
8. **Security**
9. **Internet**

4.a)Explain about Shift to a Cloud Cost Model. [CO3,K2, 6M]

You pay for resources in the cloud as you use them. For Amazon, that model is by the CPU-hour. For other clouds, such as GoGrid, it's by the RAM hour. Let's look at how you can anticipate costs using the example resource demands described earlier (two application servers from midnight until 9 a.m., eight from 9 a.m. until 5 p.m., and four from 5 p.m. until midnight).

Suppose your core infrastructure is:

$0.10/CPU-hour: one load balancer

$0.40/CPU-hour: two application servers

$0.80/CPU-hour: two database servers

Each day you would pay:

$2.40 + $44.00 + $38.40 = $84.80

Your annual hosting costs would come to $30,952.00—not including software licensing fees, cloud infrastructure management tools, or labor.

How to Approach Cost Comparisons:

The best way to compare costs in the cloud to other models is to determine the total cost of ownership over the course of your hardware depreciation period. Depending on the organization, a hardware depreciation period is generally two or three years. To get an accurate picture of your total cost of ownership for a cloud environment, you must consider the following cost elements:

• Estimated costs of virtual server usage over three years.
• Estimated licensing fees to support virtual server usage over three years.
• Estimated costs for cloud infrastructure management tools, if any, over three years.
• Estimated labor costs for creating machine images, managing the infrastructure, and responding to issues over three years.
• Any third-party setup costs for the environment.

In comparison, you must examine the following elements of your alternatives:

• What are your up-front costs (setup fees, physical space investment, hardware purchases, and license fee purchases)?
• What labor is required to set up the infrastructure?
• What are the costs associated with running the infrastructure (hosting space, electricity, insurance)?
• What is the labor costs associated with supporting the hardware and network infrastructure?
• What are the ongoing license subscription/upgrade fees? Maintenance fees?

A Sample Cloud ROI Analysis:

Let's perform an ROI analysis of a specific infrastructure that compares building it internally to launching it in the cloud.

This particular example assumes that two application servers easily support standard demand. It also assumes, however, that the business has a peak period of traffic on the 15th day of each month that lasts for 24 hours. Serving this peak capacity at the same performance levels as standard capacity requires an extra four servers.

If you're starting from scratch, you will minimally need the following equipment for your IT shop:

• Half rack at a reliable ISP with sufficient bandwidth to support your needs
• Two good firewalls
• One hardware load balancer
• Two good GB Ethernet switches
• Six solid, commodity business servers

For the cloud option, you will need just a few virtual instances:

• One medium 32-bit instance
• Four large 64-bit instances during standard usage, scaled to meet peak demand at 8

In addition, you need software and services. Assuming an entirely open source environment, your software and services costs will consist of time to set up the environments, monitoring services, support contracts, and labor for managing the environment. Table 3-1 lays out all of the expected up-front and ongoing costs.

To complete the analysis, you need to understand your organization's depreciation schedule and cost of capital. For hardware, the depreciation schedule is typically two or three years. For the purposes of this example, I will use the more conservative three-year schedule. This schedule essentially defines the expected lifetime of the hardware and frames how you combine monthly costs with one-time costs.

<u>Costs initiated with different infrastructures:</u>

The cost of capital for most organizations lies in the 10% to 20% range. In theory, the cost of capital represents what your money is worth to you if you were to invest it somewhere else. For example, if you have $10,000, a 10% cost of capital essentially says that you know you can grow that $10,000 to $11,046.69 (with the cost of capital being compounded monthly) after a year through one or more standard investment mechanisms. Another way to look at it is that gaining access to $10,000 in capital will cost you $11,046.69 after a year. Either way, the cost of $10,000 to your business is 10%, calculated monthly.

The financial expression is that you want to know the present value of all of your cash outflows over the course of the depreciation period.
For each option, calculate the present value of your monthly payments and add in any upfront costs:
Internal: = (–PV(10%/12,36,1900,0)) + 53200 = $112,083.34
Cloud: = (–PV(10%/12,36,3900,0)) + 1200 = $94,452.63
Not only is the cloud cheaper, but the payment structure of the cloud versus up-front investment saves you several thousand dollars.
The final point to note is that you can use the $112,083.34 and $94,452.63 numbers to help you understand how much money these systems need to generate over three years in order to be profitable.

b) Explain briefly about Service Levels for Cloud Applications. [C03,k2,6M]

<u>Service Levels for Cloud Applications</u>

A service level agreement (SLA) that identifies key metrics (service levels) that the customer can reasonably expect from the service. The ability to understand and to fully trust the availability, reliability, and performance of the cloud is the key conceptual block for many technologists interested in moving into the cloud.
<u>Availability:</u>
Availability describes how often a service can be used over a defined period of time. For example, if a website is accessible to the general public for 710 hours out of a 720-hour month, we say it has a 98.6% availability rating for that month.

If, for example, Google's spider is down for 24 hours but you can still search and get results, would you consider Google to be down?
Most people consider a system to have high availability if it has a documented expectation of 99.99% to 99.999% availability. At 99.999% availability, the system can be inaccessible for at most five minutes and 15 seconds over the course of an entire year.

How to estimate the availability of your system:
What constitutes availability?
Cloud service availability:
Amazon Web Services service levels:

<u>Reliability:</u>

reliability refers to how well you can trust a system to protect data integrity and execute its transactions.

<u>Performance</u>

When performing a high-performance transactional application for deployment in a physical data center applies to deployment in the cloud. Standard best practices apply:
• Design your application so logic can be spread across multiple servers.
• If you are not clustering your database, segment database access so database reads can run against slaves while writes execute against the master.
• Leverage the threading and/or process forking capabilities of your underlying platform to take advantage of as much of each individual CPU core as possible.

Clustering versus independent nodes:
EC2 performance constraints:

5. (a) Discuss about Machine image design . [CO3,K2, 6M]

Machine Image:

A machine image should include all of the software necessary for the runtime operation of a virtual instance based on that image and nothing more. The starting point is obviously the operating system, but the choice of components is absolutely critical. The full process of establishing a machine image consists of the following steps:
1. Create a component model that identifies what components and versions are required to run the service that the new machine image will support.
2. Separate out stateful data in the component model. You will need to keep it out of your machine image.
3. Identify the operating system on which you will deploy.
4. Search for an existing, trusted baseline public machine image for that operating system.

5. Harden your system using a tool such as Bastille.
6. Install all of the components in your component model.
7. Verify the functioning of a virtual instance using the machine image.
8. Build and save the machine image.

b) How data security is implemented in cloud computing. [CO3,K4, 6M]

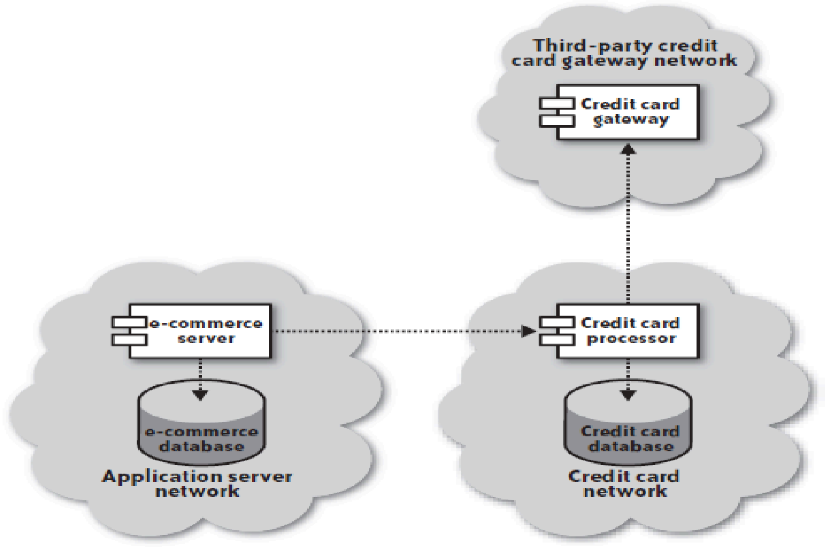
The key to privacy in the cloud—or any other environment—is the strict separation of sensitive data from non-sensitive data followed by the encryption of sensitive elements. The simplest example is storing credit cards. You may have a complex e-commerce application storing many data relationships, but you need to separate out the credit card data from the rest of it to start building a secure e-commerce infrastructure.

It's a pretty simple design that is very hard to compromise as long as you take the following precautions:

The application server and credit card server sit in two different security zones with only web services traffic from the application server being allowed into the credit card processor zone.

• Credit card numbers are encrypted using a customer-specific encryption key.
• The credit card processor has no access to the encryption key, except for a short period of time (in memory) while it is processing a transaction on that card.
• The application server never has the ability to read the credit card number from the credit card server
• No person has administrative access to both servers.

FIGURE 4-5. Host credit card data behind a web service that encrypts credit card data



Under this architecture, a hacker has no use for the data on any individual server; he must hack both servers to gain access to credit card data. Of course, if your web application is poorly written, no amount of structure will protect you against that failing. You therefore need to minimize the ability of a hacker to use one server to compromise the other. Because this problem applies to general cloud security.

For now, I'll just list a couple rules of thumb:

• Make sure the two servers have different attack vectors. In other words, they should not be running the same software. By following this guideline, you guarantee that whatever exploit compromised the first server is not available to compromise the second server.

• Make sure that neither server contains credentials or other information that will make it possible to compromise the other server. In other words, don't use passwords for user logins and don't store any private SSH keys on either server.

Managing the credit card encryption

In order to charge a credit card, you must provide the credit card number, an expiration date, and a varying number of other data elements describing the owner of the credit card. You may also be required to provide a security code.

This architecture separates the basic capture of data from the actual charging of the credit card. When a person first enters her information, the system stores contact info and some basic credit card profile information with the e-commerce application and sends the credit card number over to the credit card processor for encryption and storage.

The first trick is to create a password on the e-commerce server and store it with the customer record. It's not a password that any user will ever see or use, so you should generate something complex using the strongest password guidelines. You should also create a credit card record on the e-commerce server that stores everything except the credit card number. Figure 4-6 shows a sample e-commerce data model.

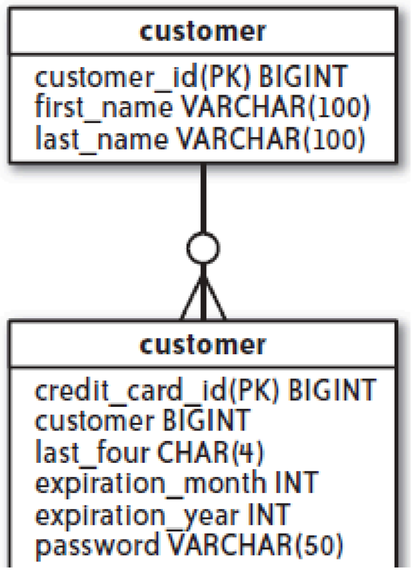
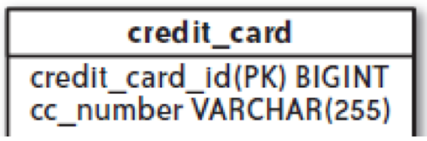


FIGURE 4-6. The e-commerce system stores everything but the credit card number and security code



credit_card
credit_card_id(PK) BIGINT
cc_number VARCHAR(255)

FIGURE 4-7. The credit card processor stores the encrypted credit card number and associates it with the e-commerce credit card ID

With that data stored in the e-commerce system database, the system then submits the credit card number, credit card password, and unique credit card ID from the e-commerce system to the credit card processor.

The credit card processor does not store the password. Instead, it uses the password as salt to encrypt the credit card number, stores the encrypted credit card number, and associates it with the credit card ID. Figure 4-7 shows the credit card processor data model.

Neither system stores a customer's security code, because the credit card companies do not allow you to store this code.

Processing a credit card transaction

When it comes time to charge the credit card, the e-commerce service submits a request to the credit card processor to charge the card for a specific amount. The e-commerce system refers to the credit card on the credit card processor using the unique ID that was created when the credit card was first inserted. It passes over the credit card password, the security code, and the amount to be charged. The credit card processor then decrypts the credit card number for the specified credit card using the specified password. The unencrypted credit card number, security code, and amount are then passed to the bank to complete the transaction.

If the e-commerce application is compromised

If the e-commerce application is compromised, the attacker has access only to the non-sensitive customer contact info. There is no mechanism by which he can download that database and access credit card information or otherwise engage in identity theft. That would require compromising the credit card processor separately.

Having said all of that, if your e-commerce application is insecure, an attacker can still assume the identity of an existing user and place orders in their name with deliveries to their address.

In other words, you still need to worry about the design of each component of the system.

If the credit card processor is compromised

Compromising the credit card processor is even less useful than compromising the e-commerce application. If an attacker gains access to the credit card database, all he has are random unique IDs and strongly encrypted credit card numbers—each encrypted with a unique encryption key. As a result, the attacker can take the database offline and attempt to brute-force decrypt the numbers, but each number will take a lot of time to crack and, ultimately, provide the hacker with a credit card number that has no individually identifying information to use in identity theft.

Another attack vector would be to figure out how to stick a Trojan application on the compromised server and listen for decryption passwords. However, if you are running intrusion detection software as suggested in Chapter 5, even this attack vector becomes unmanageable.

6.a)Explain briefly about storage-as-a-Service.. [CO4 ,K2, 6M]

Storage-as-a-Service

Storage-as-a-service is the ability to leverage storage that physically exists remotely but is logically a local storage resource to any application that requires storage .This is the most primitive component of cloud computing and is leveraged by most of the other cloud computing components.

There are a few **core benefits**. **First**, you can expand the amount of disk space available as you need it and pay only for what you use. You can reduce the amount of disk space—and thereby cost—as the need declines. This makes storage-as-a-service solutions cost effective only for larger volumes of data, typically more than 500 gigabytes, either through direct access or by using the disk as if it were local to your client computer. You can also use the storage-as-a-service provider as a redundant backup for critical files.
Second, you do not have to maintain the hardware. Drives can go down and you do not have to replace them; it is all a part of the service. When compared with an on-premise solution where you have to physically repair the drive, storage-as-a-service removes you from having to deal with that issue.
Finally, the storage-as-a-service provider provides the disaster recovery system for you, and getting back deleted files or entire directories is part of the service. The provider backs up and restores the file system as you require.

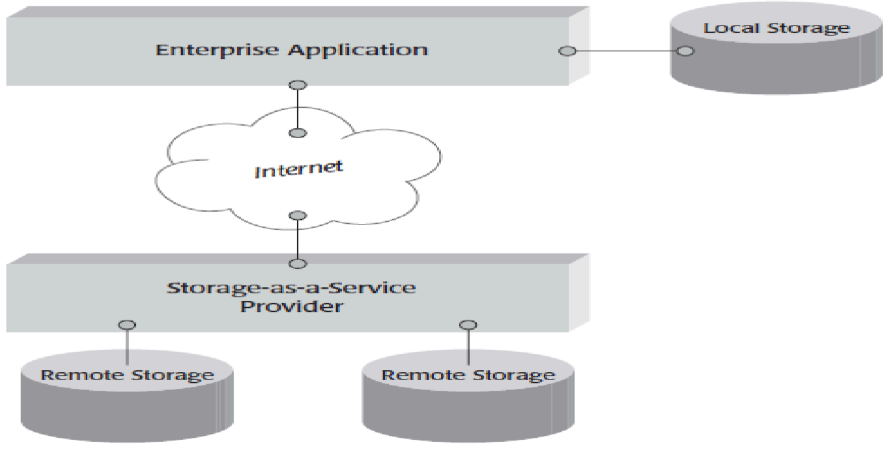


Figure: *Storage-as-a-service allows you to store information on a remote disk drive as if it were local.*
There are some **drawbacks** to storage-as-a-service. **First**, you are dependent on the Internet as the mechanism to connect to your storage as- a-service provider, and if the network goes down, you lose that connection. In many instances, those who leverage storage-as-a-service are surprised to find that they cannot access their shared disk space when not connected to the Internet, such as when on a plane.
Second, performance can be an issue. When compared to on-premise storage, where the disks are physically located near the applications that leverage them, storage-as-a-service does not provide the same performance. Thus, if performance is a critical success factor, storage-as-a-service may not be the approach you want to leverage. Of course, you can use faster connections, but the cost of implementing a higher speed network connection quickly diminishes the cost savings of storage-as-a-service.
Finally, the cost of the storage-as-a-service provider can be prohibitive when compared with an on-premise solution. While SOA using cloud computing is cost effective in some instances, in many instances it is not. The cost effectiveness of cloud computing is enterprise and domain dependent.

However, if the employees or applications are in the same building, the benefits of storage-as-a-service versus on-premise storage solutions are not as compelling.

b)**Explain about platform-as-a-Service** [CO4 ,K2, 6M]

Platform-as-a-service is a complete platform, including application development, interface development, database development, storage, and testing, delivered through a remotely hosted platform to subscribers. Based on the traditional time-sharing model, modern platform-as-a-service providers offer the ability to create enterprise-class applications for use locally or on demand for a small subscription price or for free.

Platform-as-a-service is one-stop shopping for those looking to build and deploy applications. Platform-as-a-service provides self-contained platforms with everything you need for application development and operational hosting. Platforms such as Google App Engine and Force.com are popular ways to approach application development on the cloud.

Platform-as-a-service having a few major components:

Design, development, deployment, integration, storage, and operations.

Design is the ability to design your application and user interfaces.

Development is the ability to design, develop, and test applications right out of the platform, on demand, using development tools that are delivered on demand. We have seen the Salesforce.com Apex language provide these services, with a few smaller players providing similar capabilities.

Deployment is the ability to test, bundle, and deliver the platform-as-a service– created applications. This means hosting the applications, typically accessing them visually, through a browser, or as Web services.

Integration is the ability to integrate the applications developed on your platform-as-a-service provider with software-as-a-service applications or applications that may exist within your enterprise.

Storage**,** the ability to provide persistence for the application, means an on-demand database or on-demand file storage.

Finally, ***operations*** is the ability to run the application over a long period of time, dealing with backup, restore, exception handling, and other things that add value to operations.

Platform-as-a-service is going to deliver only a subset of the existing features and functions most of us look for in a platform. Platforms are costly, and the ability to create a platform through a subscription service is compelling. Many professionals in the Global 2000 companies see platform-as-a-service as a way to develop, deploy, and maintain critical applications on the cheap.

The **advantage** of platform-as-a-service is that you can access a complete enterprise-class development environment at a low cost and build complete enterprise applications, from the data to the user interface.

The **disadvantage** is that many of the platform-as-a-service vendors leverage proprietary programming languages and interfaces; thus, once your application is there, it may be difficult to move it to an on-premise server or another platform-as-a service provider.

7. (a) Explain disaster recovery planning [CO5,K2, 6M]

Disaster recovery deals with catastrophic failures that are extremely unlikely to occur during the lifetime of a system. If they are reasonably expected failures, they fall under the auspices of traditional availability planning. Although each single disaster is unexpected over the lifetime of a system, the possibility of some disaster occurring over time is reasonably nonzero.

"Disaster Recovery Planning is identifies an acceptable recovery state and develops processes and procedures to achieve the recovery state in the event of a disaster."

Defining a disaster recovery plan involves **two key metrics**:

Recovery Point Objective (RPO)

"The recovery point objective identifies how much data you are willing to lose in the event of a disaster"**.** This value is typically specified in a number of hours or days of data. **For example**, if you determine that it is OK to lose 24 hours of data, you must make sure that the backups you'll use for your disaster recovery plan are never more than 24 hours old.

Recovery Time Objective (RTO)

"The recovery time objective identifies how much downtime is acceptable in the event of a disaster". If your RTO is 24 hours, you are saying that up to 24 hours may elapse between the point when your system first goes offline and the point at which you are fully operational again.

Accomplishing that level of redundancy is expensive. It would also come with a nontrivial performance penalty. The cold reality for most businesses is likely that the cost of losing 24 hours of data is less than the cost of maintaining a zero downtime/zero loss of data infrastructure.

Determining an appropriate RPO and RTO is ultimately a financial calculation: at what point does the cost of data loss and downtime exceed the cost of a backup strategy that will prevent that level of data loss and downtime? The right answer is radically different for different businesses.

The final element of disaster recovery planning understands the catastrophic scenario. A good disaster recovery plan can describe that scenario so that all stakeholders can understand and accept the risk.

Recovery Point Objective (RPO)

Any software system should be able to attain an RPO between 24 hours for a simple disaster to one week for a significant disaster without incurring absurd costs. Losing 24 hours of banking transactions would never be acceptable, much less one week.

RPO is typically governed by the way in which you save and back up data:

• Weekly off-site backups will survive the loss of your data center with a week of data loss. Daily off-site backups are even better.

• Daily on-site backups will survive the loss of your production environment with a day of data loss plus replicating transactions during the recovery period after the loss of the system. Hourly on-site backups are even better.

• A NAS (Network Attached System) /SAN (Storage Area Network) will survive the loss of any individual server, except for instances of data corruption with no data loss.

• A clustered database will survive the loss of any individual data storage device or database node with no data loss.
• A clustered database across multiple data centers will survive the loss of any individual data center with no data loss.

(b) Discuss about clod centers in detail [CO5,K2, 6M]

***