DevOps Interview questions and answers

CONTENTS

DevOps

34. What is the difference between a Container and a Virtual Machine?
35. What is Serverless architecture?
36. What are the main principles of DevOps?
37. Are you more Dev or more Ops?
38. What is a REST service?
39. What are the Three Ways of DevOps?
40. How do you apply DevOps principles to make system Secure?
41. What is Self-testing Code?
42. What is a Deployment Pipeline?
43. What are the main features of Docker Hub?
44. What are the security benefits of using Container based system?
45. How many heads can you create in a GIT repository?
46. What is a Passive check in Nagios?
47. What is a Docker container?
48. How will you remove an image from Docker?
49. What are the common use cases of Docker?
50. Can we lose our data when a Docker Container exits?

Docker Questions

51. What is Docker?
52. What is the difference between Docker image and Docker container?
53. How is a Docker container different from a hypervisor?
54. Can we write compose file in json file instead of yaml?
55. Can we run multiple apps on one server with Docker?
56. What are the main features of Docker-compose?
57. What is the most popular use of Docker?
58. What is the role of open source development in the popularity of Docker?
59. What is the difference between Docker commands: up, run and start?
60. What is Docker Swarm?
61. What are the features of Docker Swarm?
62. What is a Docker Image?
63. What is a Docker Container?
64. What is Docker Machine?
65. Why do we use Docker Machine?
66. How will you create a Container in Docker?
67. Do you think Docker is Application-centric or Machine-centric?
68. Can we run more than one process in a Docker container?
69. What are the objects created by Docker Cloud in Amazon Web Services (AWS) EC2?
70. How will you take backup of Docker container volumes in AWS S3?
71. What are the three main steps of Docker Compose?

72. What is Pluggable Storage Driver architecture in Docker based containers?
73. What are the main security concerns with Docker based containers?
74. How can we check the status of a Container in Docker?
75. What are the main benefits of using Docker?
76. How does Docker simplify Software Development process?
77. What is the basic architecture behind Docker?
78. What are the popular tasks that you can do with Docker Command line tool?
79. What type of applications- Stateless or Stateful are more suitable for Docker Container?
80. How can Docker run on different Linux distributions?
81. Why do we use Docker on top of a virtual machine?
82. How can Docker container share resources?
83. What is the difference between Add and Copy command in a Dockerfile?
84. What is Docker Entrypoint?
85. What is ONBUILD command in Docker?
86. What is Build cache in Docker?
87. What are the most common instructions in Dockerfile?
88. What is the purpose of EXPOSE command in Dockerfile?
89. What are the different kinds of namespaces available in a Container?
90. How will you monitor Docker in production?
91. What are the Cloud platforms that support Docker?
92. How can we control the startup order of services in Docker compose?
93. Why Docker compose does not wait for a container to be ready before moving on to start next service in dependency order?
94. How will you customize Docker compose file for different environments?

Cloud Computing Questions

95. What are the benefits of Cloud Computing?
96. What is On-demand computing in Cloud Computing?
97. What are the different layers of Cloud computing?
98. What resources are provided by Infrastructure as a Service (IAAS) provider?
99. What is the benefit of Platformas a Service?
100. What are the main advantages of PaaS?
101. What is the main disadvantage of PaaS?
102. What are the different deployment models in Cloud computing?
103. What is the difference between Scalability and Elasticity?
104. What is Software as a Service?
105. What are the different types of Datacenters in Cloud computing?
106. Explain the various modes of Software as a Service (SaaS) cloud environment?
107. What are the important things to care about in Security in a cloud environment?
108. Why do we use API in cloud computing environment?
109. What are the different areas of Security Management in cloud?

110. What are the main cost factors of cloud based data center?
111. Howcan we measure the cloud-based services?
112. Howa traditional datacenter is different froma cloud environment?
113. Howwill you optimize availability of your application in a Cloud environment?
114. What are the requirements for implementing IaaS strategy in Cloud?
115. What is the scenario in which public cloud is preferred over private cloud?
116. Do you think Cloud Computing is a software application or a hardware service?
117. Why companies nowprefer Cloud Computing architecture over Client Server Architecture?
118. What are the main characteristics of Cloud Computing architecture?
119. Howdatabases in Cloud computing are different fromtraditional databases?
120. What is Virtual Private Network (VPN)?
121. What are the main components of a VPN?
122. Howwill you secure the application data for transport in a cloud environment?
123. What are the large-scale databases available in Cloud?
124. What are the options for open source NoSQL database in a Cloud environment?
125. What are the important points to consider before selecting cloud computing?
126. What is a Systemintegrator in Cloud computing?
127. What is virtualization in cloud computing?
128. What is Eucalyptus in a cloud environment?
129. What are the main components of Eucalyptus cloud architecture?
130. What is Auto-scaling in Cloud computing?
131. What are the benefits of Utility Computing model?
132. What is a Hypervisor in Cloud Computing?
133. What are the different types of Hypervisor in Cloud Computing?
134. Why Type-1 Hypervisor has better performance than Type-2 Hypervisor?
135. What is CaaS?
136. Howis Cloud computing different fromcomputing for mobile devices?
137. Why automation of deployment is very important in Cloud architecture?
138. What are the main components in Amazon Cloud?
139. What are main components in Google Cloud?
140. What are the major offerings of Microsoft Azure Cloud?
141. What are the reasons of popularity of Cloud Computing architecture?
142. What are the Machine Learning options from Google Cloud?
143. Howwill you optimize the Cloud Computing environment?
144. Do you think Regulations and Legal Compliance is an important aspect of Cloud Computing?

Unix Questions

145. Howwill you remove all files in current directory? Including the files that are two levels down in a sub-directory.
146. What is the difference between the –v and –x options in Bash shell scripts?
147. What is a Filter in Unix command?

148. What is Kernel in Unix operating system?
149. What is a Shell in Unix OS?
150. What are the different shells in Unix that you knowabout?
151. What is the first character of the output in ls –l command ?
152. What is the difference between Multi-tasking and Multi-user environment?
153. What is an Inode in Unix?
154. What is the difference between absolute path and relative path in Unix file system?
155. What are the main responsibilities of a Unix Shell?
156. What is a Shell variable?
157. What are the important Shell variables that are initialized on starting a Shell?
158. Howwill you set the value of Environment variables in Unix?
159. What is the difference between a System Call and a library function?
160. What are the networking commands in Unix that you have used?
161. What is a Pipeline in Unix?
162. What is the use of tee command in Unix?
163. Howwill you count the number of lines and words in a file in Unix?
164. What is Bash shell?
165. Howwill you search for a name in Unix files?
166. What are the popular options of grep command in Unix?
167. What is the difference between whoami and who ami commands in Unix?
168. What is a Superuser in Unix?
169. Howwill you check the information about a process in Unix?
170. What is the use of more command with cat command?
171. What are the File modes in Unix?
172. We wrote a shell script in Unix but it is not doing anything. What could be the reason?
177. What is the significance of 755 in chmod 755 command?
178. Howcan we run a process in background in Unix? Howcan we kill a process running in background?
179. Howwill you create a read only file in Unix?
180. Howdoes alias work in Unix?
181. Howcan you redirect I/O in Unix?
182. What are the main steps taken by a Unix Shell for processing a command?
183. What is a Sticky bit in Unix?
184. What are the different outputs from Kill command in Unix?
185. Howwill you customize your environment in Unix?
186. What are the popular commands for user management in Unix?
187. Howwill you debug a shell script in Unix?
188. What is the difference between a Zombie and Orphan process in Unix?
189. Howwill you check if a remote host is still alive?
190. Howwill you get the last executed command in Unix?

191. **What is the meaning of "2>&1" in a Unix shell?**
192. **How will you find which process is taking most CPU time in Unix?**
193. **What is the difference between Soft link and Hard link in Unix?**
194. **How will you find which processes are using a file?**
195. **What is the purpose of nohup in Unix?**
196. **How will you remove blank lines from a file in Unix?**
197. **How will you find the remote hosts that are connecting to your system on a specific port in Unix?**
198. **What is xargs in Unix?**

DevOps Interview Questions

DevOps

1. What are the popular DevOps tools that you use?

We use following tools for work in DevOps:

I. **Jenkins** : This is an open source automation server used as a continuous integration tool. We can build, deploy and run automated tests with Jenkins.
II. **GIT** : It is a version control tool used for tracking changes in files and software.
III. **Docker** : This is a popular tool for containerization of services. It is very useful in Cloud based deployments.
IV. **Nagios** : We use Nagios for monitoring of IT infrastructure.
V. **Splunk** : This is a powerful tool for log search as well as monitoring production systems.
VI. **Puppet** : We use Puppet to automate our DevOps work so that it is reusable.

2. What are the main benefits of DevOps?

DevOps is a very popular trend in Software Development. Some of the main benefits of DevOps are as follows:

I. **Release Velocity** : DevOps practices help in increasing the release velocity. We can release code to production more often and with more confidence.

II. **Development Cycle** : With DevOps, the complete Development cycle from initial design to production deployment becomes shorter.

III. **Deployment Rollback** : In DevOps, we plan for any failure in deployment rollback due to a bug in code or issue in production. This gives confidence in releasing feature without worrying about downtime for rollback.

IV. **Defect Detection** : With DevOps approach, we can catch defects much earlier than releasing to production.

It improves the quality of the software.

V. **Recovery from Failure** : In case of a failure, we can recover very fast with DevOps process.

VI. **Collaboration** : With DevOps, collaboration between development and operations professionals increases.

VII. **Performance-oriented** : With DevOps, organization follows performance-oriented culture in which teams become more productive and more innovative.

3. What is the typical DevOps workflow you use in your organization?

The typical DevOps workflow in our organization is as follows:

I. We use Atlassian Jira for writing requirements and tracking tasks.
II. Based on the Jira tasks, developers checkin code into GIT version control system.
III. The code checked into GIT is built by using Apache Maven.
IV. The build process is automated with Jenkins.
V. During the build process, automated tests run to validate the code checked in by developer.
VI. Code built on Jenkins is sent to organization's Artifactory.
VII. Jenkins automatically picks the libraries from Artifactory and deploys it to Production.
VIII. During Production deployment Docker images are used to deploy same code on multiple hosts.
IX. Once code is deployed to Production, we use Nagios to monitor the health of production servers.
X. Splunk based alerts inform us of any issues or exceptions in production.

4. How do you take DevOps approach with Amazon Web Services?

Amazon Web Services (AWS) provide many tools and features to deploy and manage applications in AWS. As per DevOps, we treat infrastructure as code. We mainly use following two services from AWS for DevOps:

I. **CloudFormation** : We use AWS CloudFormation to create and deploy AWS resources by using templates. We can describe our dependencies and pass special parameters in these templates. CloudFormation can read these templates and deploy the application and resources in AWS cloud.

II. **OpsWorks** : AWS provides another service called OpsWorks that is used for configuration management by utilizing Chef framework. We can automate server configuration, deployment and management by using OpsWorks. It helps in managing EC2 instances in AWS as well as any on-premises servers.

5. How will you run a script automatically when a developer commits a change into GIT?

GIT provides the feature to execute customscripts when certain event occurs in GIT. This feature is called hooks.

We can write two types of hooks.

I. Client-side hooks
II. Server-side hooks

For this case, we can write a Client-side post-commit hook. This hook will execute a custom script in which we can add the message and code that we want to run automatically with each commit.

6. What are the main features of AWS OpsWorks Stacks?

Some of the main features of AWS OpsWorks Stacks are as follows:

I. **Server Suppo** rt: AWS OpsWorks Stacks we can automate operational tasks on any server in AWS as well as our own data center.
II. **Scalable Automation** : We get automated scaling support with AWS OpsWorks Stacks. Each new instance in AWS can read configuration from OpsWorks. It can even respond to system events in same way as other instances do.

III. **Dashboard** : We can create dashboards in OpsWorks to display the status of all the stacks in AWS.
IV. **Configuration as Code** : AWS OpsWorks Stacks are built on the principle of "Configuration as Code". We can define and maintain configurations like application source code. Same configuration can be replicated on multiple servers and environments.
V. **Application Support** : OpsQorks supports almost all kinds of applications. So it is universal in nature.

7. How does CloudFormation work in AWS?

AWS CloudFormation is used for deploying AWS resources.
In CloudFormation, we have to first create a template for a resource. A template is a simple text file that contains information about a stack on AWS. A stack is a collection of AWS resourced that we want to deploy together in an AWS as a group.

Once the template is ready and submitted to AWS, CloudFormation will create all the resources in the template. This helps in automation of building new environments in AWS.

8. What is CICD in DevOps?

CICD stands for Continuous Integration and Continuous Delivery. These are two different concepts that are complementary to each other.

Continuous Integration (CI) : In CI all the developer work is merged to main branch several times a day. This helps in reducing integration problems.

In CI we try to minimize the duration for which a branch remains checked out. A developer gets early feedback on the new code added to main repository by using CI.

Continuous Delivery (CD) : In CD, a software team plans to deliver software in short cycles. They perform development, testing and release in such a short time that incremental changes can be easily delivered to production.

In CD, as a DevOps we create a repeatable deployment process that can help achieve the objective of Continuous Delivery.

9. What are the best practices of Continuous Integration (CI)?

Some of the best practices of Continuous Integration (CI) are as follows:

I. **Build Automation** : In CI, we create such a build environment that even with one command build can be triggered. This automation is done all the way up to deployment to Production environment.
II. **Main Code Repository** : In CI, we maintain a main branch in code repository that stores all the Production ready code. This is the branch that we can deploy to Production any time.
III. **Self-testing build** : Every build in CI should be self-tested. It means with every build there is a set of tests that runs to ensure that changes are of high quality.
IV. **Every day commits to baseline** : Developers will commit all of theirs changes to baseline everyday. This ensures that there is no big pileup of code waiting for integration with the main repository for a long time.
V. **Build every commit to baseline** : With Automated Continuous Integration, every time a commit is made into baseline, a build is triggered. This helps in confirming that every change integrates correctly.
VI. **Fast Build Process** : One of the requirements of CI is to keep the build process fast so that we can quickly identify any problem.
VII. **Production like environment testing** : In CI, we maintain a production like environment also known as pre-production or staging environment, which is very close to Production environment. We perform testing in this environment to check for any integration issues.
VIII. **Publish Build Results** : We publish build results on a common site so that everyone can see these and take corrective actions.
IX. **Deployment Automation** : The deployment process is automated to the extent that in a build process we can add the step of deploying the code to a test environment. On this test environment all the stakeholders can access and test the latest delivery.

10. What are the benefits of Continuous Integration (CI)?

The benefits of Continuous Integration (CI) are as follows:

I. CI makes the current build constantly available for testing, demo and release purpose.
II. With CI, developers write modular code that works well with frequent code check-ins.
III. In case of a unittest failure or bug, developer can easily revert back to the bug-free state of the code.
IV. There is drastic reduction in chaos on release day with CI practices.
V. With CI, we can detect Integration issues much earlier in the process.
VI. Automated testing is one very useful side effect of implementing CI.
VII. All the stakeholders including business partners can see the small changes deployed into pre-production environment. This provides early feedback on the changes to software.
VIII. Automated CI and testing generates metrics like code-coverage, code complexity that help in improving the development process.

11. What are the options for security in Jenkins?

In Jenkins, it is very important to make the system secure by setting user authentication and authorization. To do this we have to do following:

I. First we have to set up the Security Realm. We can integrate Jenkins with LDAP server to create user authentication.
II. Second part is to set the authorization for users. This determines which user has access to what resources.

In Jenkins some of the options to setup security are as follows:

I. We can use Jenkins' own User Database.
II. We can use LDAP plugin to integrate Jenkins with LDAP server.
III. We can also setup Matrix based security on Jenkins.

12. What are the main benefits of Chef?

Chef is an automation tool for keeping infrastructure as code. It has many benefits. Some of these are as follows:

I. **Cloud Deployment** : We can use Chef to perform automated deployment in Cloud environment.

II. **Multi-cloud support** : With Chef we can even use multiple cloud providers for our infrastructure.

III. **Hybrid Deployment** : Chef supports both Cloud based as well as datacenter-based infrastructure.

IV. **High Availability** : With Chef automation, we can create high availability environment. In case of hardware failure, Chef can maintain or start new servers in automated way to maintain highly available environment.

13. What is the architecture of Chef?

Chef is composed of many components like Chef Server, Client etc. Some of the main components in Chef are as follows:

I. **Client** : These are the nodes or individual users that communicate with Chef server.
II. **Chef Manage** : This is the web console that is used for interacting with Chef Server.
III. **Load Balancer** : All the Chef server API requests are routed through Load Balancer. It is implemented in Nginx.
IV. **Bookshelf** : This is the component that stores cookbooks. All the cookbooks are stored in a repository. It is separate storage from the Chef server.
V. **PostgreSQL** : This is the data repository for Chef server.
VI. **Chef Server** : This is the hub for configuration data. All the cookbooks and policies are stored in it. It can scale to the size of any enterprise.

14. What is a Recipe in Chef?

In any organization, Recipe is the most fundamental configuration element.
It is written in Ruby language. It is a collection of resources defined by using patterns.

A Recipe is stored in a Cookbook and it may have dependency on other Recipe.

We can tag Recipe to create some kind of grouping.

We have to add a Recipe in run-list before using it by chef-client.

It always maintains the execution order specified in run-list.

15. What are the main benefits of Ansible?

Ansible is a powerful tool for IT Automation for large scale and complex deployments. It increases the productivity of team. Some of the main benefits of Ansible are as follows:

I. **Productivity** : It helps in delivering and deploying with speed. It increases productivity in an organization.

II. **Automation** : Ansible provides very good options for automation. With automation, people can focus on delivering smart solutions.

III. **Large-scale** : Ansible can be used in small as well as very large-scale organizations.

IV. **Simple DevOps** : With Ansible, we can write automation in a human-readable language. This simplifies the task of DevOps.

16. What are the main use cases of Ansible?

Some of the popular use cases of Ansible are as follows:

I. **App Deployment** : With Ansible, we can deploy apps in a reliable and repeatable way.

II. **Configuration Management** : Ansible supports the automation of configuration management across multiple environments.

III. **Continuous Delivery** : We can release updates with zero downtime with Ansible.

IV. **Security** : We can implement complex security policies with Ansible.

V. **Compliance** : Ansible helps in verifying and organization's systems in comparison with the rules and regulations.

VI. **Provisioning** : We can provide new systems and resources to other users with Ansible.

VII. **Orchestration** : Ansible can be used in orchestration of complex deployment in a simple way.

17. What is Docker Hub?

Docker Hub is a cloud-based registry. We can use Docker Hub to link code repositories. We can even build images and store them in Docker Hub. It also provides links to Docker Cloud to deploy the images to our hosts.

Docker Hub is a central repository for container image discovery, distribution, change management, workflow automation and

teamcollaboration.

18. What is your favorite scripting language for DevOps?

In DevOps, we use different scripting languages for different purposes. There is no single language that can work in all the scenarios. Some of the popular scripting languages that we use are as follows:

I. **Bash** : On Unix based systems we use Bash shell scripting for automating tasks.

II. **Python** : For complicated programming and large modules we use Python. We can easily use a wide variety of standard libraries with Python.

III. **Groovy** : This is a Java based scripting language. We need JVM installed in an environment to use Groovy. It is very powerful and it provides very powerful features.

IV. **Perl** : This is another language that is very useful for text parsing. We use it in web applications.

19. What is Multi-factor authentication?

In security implementation, we use Multi-factor authentication (MFA). In MFA, a user is authenticated by multiple means before giving access to a resource or service. It is different from simple user/password based authentication.

The most popular implementation of MFA is Two-factor authentication. In most of the organizations, we use username/password and an RSA token as two factors for authentication.

With MFA, the systembecomes more secure and it cannot be easily hacked.

20. What are the main benefits of Nagios?

Nagios is open source software to monitor systems, networks and infrastructure. The main benefits of Nagios are as follows:

I. **Monitor** : DevOps can configure Nagios to monitor IT infrastructure components, system metrics and network protocols.

II. **Alert** : Nagios will send alerts when a critical component in infrastructure fails.

III. **Response** : DevOps acknowledges alerts and takes corrective actions.

IV. **Report** : Periodically Nagios can publish/send reports on outages, events and SLAs etc.

V. Maintenance: During maintenance windows, we can also disable alerts.

VI. **Planning** : Based on past data, Nagios helps in infrastructure planning and upgrades.

21. What is State Stalking in Nagios?

State Stalking is a very useful feature. Though all the users do not use it all the time, it is very helpful when we want to investigate an issue.

In State Stalking, we can enable stalking on a host. Nagios will monitor the state of the host very carefully and it will log any changes in the state.

By this we can identify what changes might be causing an issue on the host.

22. What are the main features of Nagios?

Some of the main features of Nagios are as follows:

I. **Visibility** : Nagios provides a centralized view of the entire IT infrastructure.

II. **Monitoring** : We can monitor all the mission critical infrastructure components with Nagios.

III. **Proactive Planning** : With Capacity Planning and Trending we can proactively plan to scale up or scale down the infrastructure.

IV. **Extendable** : Nagios is extendable to a third party tools in APIs.

V. **Multi-tenant** : Nagios supports multi-tenants architecture.

23. What is Puppet?

Puppet Enterprise is a DevOps software platform that is used for automation of infrastructure operations. It runs on Unix as well as on Windows.

We can define systemconfiguration by using Puppet's language or Ruby DSL.

The systemconfiguration described in Puppet's language can be distributed to a target systemby using REST API calls.

24. What is the architecture of Puppet?

Puppet is Open Source software. It is based on Client-server architecture. It is a Model Driven system. The client is also called Agent. And server is called Master.

It has following architectural components:

I. **Configuration Language** : Puppet provides a language that is used to configure Resources. We have to specify what Action has to be applied to which Resource.

The Action has three items for each Resource: type, title and list of attributes of a resource. Puppet code is written in Manifests files.

II. **Resource Abstraction** : We can create Resource Abstraction in Puppet so that we can configure resources on different platforms. Puppet agent uses a Facter for passing the information of an environment to Puppet server. In Facter we have information about IP, hostname, OS etc of the environment.

III. **Transaction** : In Puppet, Agent sends Facter to Master server. Master sends back the catalog to Client. Agent applies any configuration changes to system. Once all changes are applied, the result is sent to Server.

25. What are the main use cases of Puppet Enterprise?

We can use Puppet Enterprise for following scenarios:

I. **Node Management** : We can manage a large number of nodes with Puppet.

II. **Code Management** : With Puppet we can define Infrastructure as code. We can review, deploy, and test the environment configuration for Development, Testing and Production environments.

III. **Reporting & Visualization** : Puppet provides Graphical tools to visualize and see the exact status of infrastructure configuration.

IV. **Provisioning Automation** : With Puppet we can automate deployment and creation of new servers and resources. So users and business can get their infrastructure requirements completed very fast with Puppet.

V. **Orchestration** : For a large Cluster of nodes, we can orchestrate the complete process by using Puppet. It

can follow the order in which we want to deploy the infrastructure environments.

VI. **Automation of Configuration** : With Configuration automation, the chances of manual errors are reduced. The process becomes more reliable with this.

26. What is the use of Kubernetes?

We use Kubernetes for automation of large-scale deployment of Containerized applications.

It is an open source system based on concepts similar to Google's deployment process of millions of containers.

It can be used on cloud, on-premise datacenter and hybrid infrastructure.

In Kubernetes we can create a cluster of servers that are connected to work as a single unit. We can deploy a containerized application to all the servers in a cluster without specifying the machine name.

We have to package applications in such a way that they do not depend on a specific host.

27. What is the architecture of Kubernetes?

The architecture of Kubernetes consists of following components:

Master : There is a master node that is responsible for managing the cluster. Master performs following functions in a cluster.

I. Scheduling Applications
II. Maintaining desired state of applications
III. Scaling applications
IV. Applying updates to applications

Nodes : A Node in Kubernetes is responsible for running an application. The Node can be a Virtual Machine or a Computer in the cluster. There is software called Kubelet on each node. This software is used for managing the node and communicating with the Master node in cluster.

There is a Kubernetes API that is used by Nodes to communicate with the Master. When we deploy an application on Kubernetes, we request Master to start application containers on Nodes.

28. How does Kubernetes provide high availability of applications in a Cluster?

In a Kubernetes cluster, there is a Deployment Controller. This controller monitors the instances created by Kubernetes in a cluster. Once a node or the machine hosting the node goes down, Deployment Controller will replace the node.

It is a self-healing mechanism in Kubernetes to provide high availability of applications.

Therefore in Kubernetes cluster, Kubernetes Deployment Controller is responsible for starting the instances as well as replacing the instances in case of a failure.

29. Why Automated Testing is a must requirement for DevOps?

In DevOps approach we release software with high frequency to production. We have to run tests to gain confidence on the quality of software deliverables.

Running tests manualy is a time taking process. Therefore, we first prepare automation tests and then deliver software. This ensures that we catch any defects early in our process.

30. What is Chaos Monkey in DevOps?

Chaos Monkey is a concept made popular by Netflix. In Chaos Monkey, we intentionally try to shut down the services or create failures. By failing one or more services, we test the reliability and recovery mechanism of the Production architecture.

It checks whether our applications and deployment have survival strategy built into it or not.

31. How do you perform Test Automation in DevOps?

We use Jenkins to create automated flows to run Automation tests. The first part of test automation is to develop test strategy and test cases. Once automation test cases are ready for an application, we have to plug these into each Build run.
In each Build we run Unit tests, Integration tests and Functional tests.

With a Jenkins job, we can automate all these tasks. Once all the automated tests pass, we consider the build as green. This helps in deployment and release processes to build confidence on the application software.

32. What are the main services of AWS that you have used?

We use following main services of AWS in our environment:

I. **EC2** : This is the Elastic Compute Cloud by Amazon. It is used to for providing computing capability to a system. We can use it in places of our standalone servers. We can deploy different kinds of applications on C2.
II. **S3** : We use S3 in Amazon for our storage needs.
III. **DynamoDB** : We use DynamoDB in AWS for storing data in NoSQL database form.
IV. **Amazon CloudWatch** : We use CloudWatch to monitor our application in Cloud.
V. **Amazon SNS** : We use Simple Notification Service to inform users about any issues in Production environment.

33. Why GIT is considered better than CVS for version control system?

GIT is a distributed system. In GIT, any person can create its own branch and start checking in the code. Once the code is tested, it is merged into main GIT repo. IN between, Dev, QA and product can validate the implementation of that code.

In CVS, there is a centralized system that maintains all the commits and changes.
GIT is open source software and there are plenty of extensions in GIT for use by our teams.

34. What is the difference between a Container and a Virtual Machine?

We need to select an Operating System (OS) to get a specific Virtual Machine (VM). VM provides full OS to an application for running in a virtualized environment.

A Container uses APIs of an Operating System(OS) to provide runtime environment to an application.

A Container is very lightweight in comparison with a VM.

VM provides higher level of security compared to a Container.

A Container just provides the APIs that are required by the application.

35. What is Serverless architecture?

Serverless Architecture is a term that refers to following:

I. An Application that depends on a third-party service.
II. An Application in which Code is run on ephemeral containers.

In AWS, Lambda is a popular service to implement Serverless architecture.

Another concept in Serverless Architecture is to treat code as a service or Function as a Service (FAAS). We just write code that can be run on any environment or server without the need of specifying which server should be used to run this code.

36. What are the main principles of DevOps?

DevOps is different from Technical Operations. It has following main principles:

I. **Incremental** : In DevOps we aim to incrementally release software to production. We do releases to production more often than Waterfall approach of one large release.

II. **Automated** : To enable use to make releases more often, we automate the operations from Code Check in to deployment in Production.

III. **Collaborative** : DevOps is not only responsibility of Operations team. It is a collaborative effort of Dev, QA, Release and DevOps teams.

IV. **Iterative** : DevOps is based on Iterative principle of using a process that is repeatable. But with each iteration we aim to make the process more efficient and better.

V. **Self-Service** : In DevOps, we automate things and give self-service options to other teams so that they are empowered to deliver the work in their domain.

37. Are you more Dev or more Ops?

This is a tricky question. DevOps is a new concept and in any organization the maturity of DevOps varies from highly Operations oriented to highly DevOps oriented. In some projects teams are very mature and practice DevOps in it true form. In some projects, teams rely more on Operations team.

As a DevOps person I give first priority to the needs of an organization and project. At some times I may have to perform a lot of operations work. But with each iteration, I aim to bring DevOps changes incrementally to an organization.

Over time, organization/project starts seeing results of DevOps practices and embraces it fully.

38. What is a REST service?

REST is also known as Representational State Transfer. A REST service is a simple software functionality that is available over HTTP protocol. It is a lightweight service that is widely available due to the popularity of HTTP protocol.

Sine REST is lightweight; it has very good performance in a software system. It is also one of the foundations for creating highly scalable systems that provide a service to large number of clients.

Another key feature of a REST service is that as long as the interface is kept same, we can change the underlying implementation. E.g. Clients of REST service can keep calling the same service while we change the implementation from php to Java.

39. What are the Three Ways of DevOps?

Three Ways of DevOps refers to three basic principles of DevOps culture. These are as follows:

I. **The First Way: Systems Thinking** : In this principle we see the DevOps as a flow of work from left to right. This is the time taken from Code check in to the feature being released to End customer. In DevOps culture we try to identify the bottlenecks in this.

II. **The Second Way: Feedback Loops** : Whenever there is an issue in production it is a feedback about the whole development and deployment process. We try to make the feedback loop more efficient so that teams can get the feedback much faster. It is a way of catching defect much earlier in process than it being reported by customer.

III. **The Third Way: Continuous Learning** : We make use of first and second way principles to keep on making improvements in the overall process. This is the third principle in which over the time we make the process and our operations highly efficient, automated and error free by continuously improving them.

40. How do you apply DevOps principles to make system Secure?

Security of a system is one of the most important goals for an organization. We use following ways to apply DevOps to security.

I. **Automated Security Testing** : We automate and integrate Security testing techniques for Software Penetration testing and Fuzz testing in software development process.

II. **Early Security Checks** : We ensure that teams know about the security concerns at the beginning of a project, rather than at the end of delivery. It is achieved by conducting Security trainings and knowledge sharing sessions.

III. **Standard Process** : At DevOps we try to follow standard deployment and development process that has already gone through security audits. This helps in minimizing the introduction of any new security loopholes due to change in the standard process.

41. What is Self-testing Code?

Self-testing Code is an important feature of DevOps culture. In DevOps culture, development team members are expected to write self-testing code. It means we have to write code along with the tests that can test this code. Once the test passes, we feel confident to release the code.

If we get an issue in production, we first write an automation test to validate that the issue happens in current release. Once the issue in release code is fixed, we run the same test to validate that the defect is not there. With each release we keep running these tests so that the issue does not appear anymore.

One of the techniques of writing Self-testing code is Test Driven Development (TDD).

42. What is a Deployment Pipeline?

A Deployment Pipeline is an important concept in Continuous Delivery. In Deployment Pipeline we break the build process into distinct stages. In each stage we get the feedback to move onto the next stage.

It is a collaborative effort between various groups involved in delivering software development.
Often the first stage in Deployment Pipeline is compiling the code and converting into binaries.

After that we run the automated tests. Depending on the scenario, there are stages like performance testing, security check, usability testing etc in a Deployment Pipeline.

In DevOps, our aim is to automate all the stages of Deployment Pipeline. With a smooth running Deployment Pipeline, we can achieve the goal of Continuous Delivery.

43. What are the main features of Docker Hub?

Docker Hub provides following main features:

I. **Image Repositories** : In Docker Hub we can push, pull, find and manage Docker Images. It is a big library that has images from community, official as well as private sources.

II. **Automated Builds** : We can use Docker Hub to create new images by making changes to source code repository of the image.

III. **Webhooks** : With Webhooks in Docker Hub we can trigger actions that can create and build new images by pushing a change to repository.

IV. **Github/Bitbucket integration** : Docker Hub also provides integration with Github and Bitbucket systems.

44. What are the security benefits of using Container based system?

Some of the main security benefits of using a Container based systemare as follows:

I. **Segregation** : In a Container based system we segregate the applications on different containers. Each application may be running on same host but in a separate container. Each application has access to ports, files and other resources that are provided to it by the container.

II. **Transient** : In a Container based system, each application is considered as a transient system. It is better than a static system that has fixed environment which can be exposed overtime.

III. **Control:** We use repeatable scripts to create the containers. This provides us tight control over the software application that we want to deploy and run. It also reduces the risk of unwanted changes in setup that can cause security loopholes.

IV. **Security Patch:** In a Container based system; we can deploy security patches on multiple containers in a uniform way. Also it is easier to patch a Container with an application update.

45. How many heads can you create in a GIT repository?

There can be any number of heads in a GIT repository.

By default there is one head known as HEAD in each repository in GIT.

46. What is a Passive check in Nagios?

In Nagios, we can monitor hosts and services by active checks. In addition, Nagios also supports Passive checks that are initiated by external applications.

The results of Passive checks are submitted to Nagios. There are two main use cases of Passive checks:

I. We use Passive checks to monitor asynchronous services that do not give positive result with Active checks at regular intervals of time.

II. We can use Passive checks to monitor services or applications that are located behind a firewall.

47. What is a Docker container?

ADocker Container is a lightweight system that can be run on a Linux operating system or a virtual machine. It is a package of an application and related dependencies that can be run independently.

Since Docker Container is very lightweight, multiple containers can be run simultaneously on a single server or virtual machine.

With a Docker Container we can create an isolated system with restricted services and processes. A Container has private view of the operating system. It has its own process ID space, file system, and network interface.

Multiple Docker Containers can share same Kernel.

48. How will you remove an image from Docker?

We can use docker rmi command to delete an image from our local system.

Exact command is:

% docker rmi <Image Id>

If we want to find IDs of all the Docker images in our local system, we can user docker images command.

% docker images

If we want to remove a docker container then we use docker rmcommand.

% docker rm<Container Id>

49. What are the common use cases of Docker?

Some of the common use cases of Docker are as follows:

I. **Setting up Development Environment** : We can use Docker to set the development environment with the applications on which our code is dependent.
II. **Testing Automation Setup** : Docker can also help in creating the Testing Automation setup. We can setup different services and apps with Docker to create the automation-testing environment.
III. **Production Deployment** : Docker also helps in implementing the Production deployment for an application. We can use it to create the exact environment and process that will be used for doing the production deployment.

50. Can we lose our data when a Docker Container exits?

A Docker Container has its own file-system. In an application running on Docker Container we can write to this file-system. When the container exits, data written to file-system still remains. When we restart the container, same data can be accessed again.

Only when we delete the container, related data will be deleted.

Docker Questions

51. What is Docker?

Docker is Open Source software. It provides the automation of Linux application deployment in a software container.

We can do operating system level virtualization on Linux with Docker.

Docker can package software in a complete file system that contains software code, runtime environment, system tools, &

libraries that are required to install and run the software on a server.

52. What is the difference between Docker image and Docker container?

Docker container is simply an instance of Docker image.

A Docker image is an immutable file, which is a snapshot of container. We create an image with **build** command.

When we use run command, an Image will produce a container.

In programming language, an Image is a Class and a Container is an instance of the class.

53. How is a Docker container different from a hypervisor?

In a Hypervisor environment we first create a Virtual Machine and then install an Operating System on it. After that we deploy the application. The virtual machine may also be installed on different hardware configurations.

In a Docker environment, we just deploy the application in Docker. There is no OS layer in this environment. We specify libraries, and rest of the kernel is provided by Docker engine.

In a way, Docker container and hypervisor are complementary to each other.

54. Can we write compose file in json file instead of yaml?

Yes. Yaml format is a superset of json format. Therefore any json file is also a valid Yaml file.

If we use a json file then we have to specify in docker command that we are using a json file as follows:

% docker-compose -f docker-compose.json up

55. Can we run multiple apps on one server with Docker?

Yes, theoretically we can run multiples apps on one Docker server. But in practice, it is better to run different components on separate containers.

With this we get cleaner environment and it can be used for multiple uses.

56. What are the main features of Docker-compose?

Some of the main features of Docker-compose are as follows:

I. **Multiple environments on same Host** : We can use it to create multiple environments on the same host server.

II. **Preserve Volume Data on Container Creation** : Docker compose also preserves the volume data when we create a container.
III. **Recreate the changed Containers** : We can also use compose to recreate the changed containers.
IV. **Variables in Compose file** : Docker compose also supports variables in compose file. In this way we can create variations of our containers.

57. What is the most popular use of Docker?

The most popular use of Docker is in build pipeline. With the use of Docker it is much easier to automate the development to deployment process in build pipeline.

We use Docker for the complete build flow from development work, test run and deployment to production environment.

58. What is the role of open source development in the popularity of Docker?

Since Linux was an open source operating system, it opened new opportunities for developers who want to contribute to open source systems.

One of the very good outcomes of open source software is Docker. It has very powerful features.

Docker has wide acceptance due to its usability as well as its open source approach of integrating with different systems.

59. What is the difference between Docker commands: up, run and start?

We have up and start commands in docker-compose. The run command is in docker.

a. **Up** : We use this command to build, create, start or restart all the services in a docker-compose.yml file. It also attaches to containers for a service.

   This command can also start linked services.

b. **Run** : We use this command for adhoc requests. It just starts the service that we specifically want to start. We generally use it run specific tests or any administrative tasks.
c. **Start** : This command is used to start the container that were previously created but are not currently running. This command does not create new containers.

60. What is Docker Swarm?

Docker Swarm is used to create a cluster environment. It can turn a group of Docker engines into a Single virtual Docker Engine. This creates a system with pooled resources. We can use Docker Swarm to scale our application.

61. What are the features of Docker Swarm?

Some of the key features of Docker Swarm are as follows:

I. **Compatible** : Docker Swarm is compatible with standard Docker API.
II. **High Scalability** : Swarm can scale up to as much as 1000 nodes and 50000 containers. There is almost no performance degradation at this scale in Docker Swarm.
III. **Networking** : Swarm comes with support for Docker Networking.
IV. **High Availability** : We can create a highly available system with Docker Swarm. It allows use to create

multiple master nodes so that in case of a failure, another node can take over.

V. **Node Discovery** : In Docker Swarm, we can add more nodes and the new nodes can be found with any discovery service like etcd or zookeeper etc.

62. What is a Docker Image?

Docker Image is the blue print that is used to create a Docker Container. Whenever we want to run a container we have to specify the image that we want to run.

There are many Docker images available online for standard software. We can use these images directly from the source.

The standard set of Docker Images is stored in Docker Hub Registry. We can download these from this location and use it in our environment.

We can also create our own Docker Image with the software that we want to run as a container.

63. What is a Docker Container?

ADocker Container is a lightweight system that can be run on a Linux operating system or a virtual machine. It is a package of an application and related dependencies that can be run independently.

Since Docker Container is very lightweight, multiple containers can be run simultaneously on a single server or virtual machine.

With a Docker Container we can create an isolated system with restricted services and processes. A Container has private view of the operating system. It has its own process ID space, file system, and network interface.

Multiple Docker Containers can share same Kernel.

64. What is Docker Machine?

We can use Docker Machine to install Docker Engine on virtual hosts. It also provides commands to manage virtual hosts.

Some of the popular Docker machine commands enable us to start, stop, inspect and restart a managed host.

Docker Machine provides a Command Line Interface (CLI), which is very useful in managing multiple hosts.

65. Why do we use Docker Machine?

There are two main uses of Docker Machine:

I. **Old Desktop** : If we have an old desktop and we want to run Docker then we use Docker Machine to run Docker. It is like installing a virtual machine on an old hardware system to run Docker engine.

II. **Remote Hosts** : Docker Machine is also used to provision Docker hosts on remote systems. By using Docker Machine you can install Docker Engine on remote hosts and configure clients on them.

66. How will you create a Container in Docker?

To create a Container in Docker we have to create a Docker Image. We can also use an existing Image from Docker Hub Registry.

We can run an Image to create the container.

67. Do you think Docker is Application-centric or Machine-centric?

Docker is an Application-centric solution. It is optimized for deployment of an application. It does not replace a machine by creating a virtual machine. Rather, it focuses on providing ease of use features to run an application.

68. Can we run more than one process in a Docker container?

Yes, a Docker Container can provide process management that can be used to run multiple processes. There are process supervisors like runit, s6, daemontools etc that can be used to fork additional processes in a Docker container.

69. What are the objects created by Docker Cloud in Amazon Web Services (AWS) EC2?

Docker Cloud creates following objects in AWS EC2 instance:

I. **VPC** : Docker Cloud creates a Virtual Private Cloud with the tag name dc-vpc. It also creates Class Less Inter-Domain Routing (CIDR) with the range of 10.78.0.0/16 .

II. **Subnet** : Docker Cloud creates a subnet in each Availability Zone (AZ). In Docker Cloud, each subnet is tagged with dc-subnet.

III. **Internet Gateway** : Docker Cloud also creates an internet gateway with name dc-gateway and attaches it to the VPC created earlier.

IV. **Routing Table** : Docker Cloud also creates a routing table named dc-route-table in Virtual Private Cloud. In this Routing Table Docker Cloud associates the subnet with the Internet Gateway.

70. How will you take backup of Docker container volumes in AWS S3?

We can use a utility named Dockup provided by Docker Cloud to take backup of Docker container volumes in S3.

71. What are the three main steps of Docker Compose?

Three main steps of Docker Compose are as follows:

I. **Environment** : We first define the environment of our application with a Dockerfile. It can be used to recreate the environment at a later point of time.

II. **Services** : Then we define the services that make our app in docker-compose.yml. By using this file we can define how these services can be run together in an environment.

III. **Run** : The last step is to run the Docker Container. We use docker-compose up to start and run the application.

72. What is Pluggable Storage Driver architecture in Docker based containers?

Docker storage driver is by default based on a Linux file system. But Docker storage driver also has provision to plug in any

other storage driver that can be used for our environment.

In Pluggable Storage Driver architecture, we can use multiple kinds of file systems in our Docker Container. In Docker info command we can see the Storage Driver that is set on a Docker daemon.

We can even plug in shared storage systems with the Pluggable Storage Driver architecture.

73. What are the main security concerns with Docker based containers?

Docker based containers have following security concerns:

I. **Kernel Sharing** : In a container-based system, multiple containers share same Kernel. If one container causes Kernel to go down, it will take down all the containers. In a virtual machine environment we do not have this issue.

II. **Container Leakage** : If a malicious user gains access to one container, it can try to access the other containers on the same host. If a container has security vulnerabilities it can allow the user to access other containers on same host machine.

III. **Denial of Service** : If one container occupies the resources of a Kernel then other containers will starve for resources. It can create a Denial of Service attack like situation.

IV. **Tampered Images** : Sometimes a container image can be tampered. This can lead to further security concerns. An attacker can try to run a tampered image to exploit the vulnerabilities in host machines and other containers.

V. **Secret Sharing** : Generally one container can access other services. To access a service it requires a Key or Secret. A malicious user can gain access to this secret. Since multiple containers share the secret, it may lead to further security concerns.

74. How can we check the status of a Container in Docker?

We can use docker ps –a command to get the list of all the containers in Docker. This command also returns the status of these containers.

75. What are the main benefits of using Docker?

Docker is a very powerful tool. Some of the main benefits of using Docker are as follows:

I. **Utilize Developer Skills** : With Docker we maximize the use of Developer skills. With Docker there is less need of build or release engineers. Same Developer can create software and wrap it in one single file.

II. **Standard Application Image** : Docker based system allows us to bundle the application software and Operating system files in a single Application Image that can be deployed independently.

III. **Uniform deployment** : With Docker we can create one package of our software and deploy it on different platforms seamlessly.

76. How does Docker simplify Software Development process?

Prior to Docker, Developers would develop software and pass it to QA for testing and then it is sent to Build & Release team for deployment.

In Docker workflow, Developer builds an Image after developing and testing the software. This Image is shipped to Registry. From Registry it is available for deployment to any system. The development process is simpler since steps for QA and Deployment etc take place before the Image is built. So Developer gets the feedback early.

77. What is the basic architecture behind Docker?

Docker is built on client server model. Docker server is used to run the images. We use Docker client to communicate with Docker server.

Clients tell Docker server via commands what to do.

Additionally there is a Registry that stores Docker Images. Docker Server can directly contact Registry to download images.

78. What are the popular tasks that you can do with Docker Command line tool?

Docker Command Line (DCL) tool is implemented in Go language. It can compile and run on most of the common operating systems. Some of the tasks that we can do with Docker Command Line tool are as follows:

I. We can download images from Registry with DCL.

II. We can start, stop or terminate a container on a Docker server by DCL.

III. We can retrieve Docker Logs via DCL.

IV. We can build a Container Image with DCL.

79. What type of applications- Stateless or Stateful are more suitable for Docker Container?

It is preferable to create Stateless application for Docker Container. We can create a container out of our application and take out the configurable state parameters from application. Now we can run same container in Production as well as QA environments with different parameters. This helps in reusing the same Image in different scenarios. Also a stateless application is much easier to scale with Docker Containers than a stateful application.

80. How can Docker run on different Linux distributions?

Docker directly works with Linux kernel level libraries. In every Linux distribution, the Kernel is same. Docker containers share same kernel as the host kernel.

Since all the distributions share the same Kernel, the container can run on any of these distributions.

81. Why do we use Docker on top of a virtual machine?

Generally we use Docker on top of a virtual machine to ensure isolation of the application. On a virtual machine we can get the advantage of security provided by hypervisor. We can implement different security levels on a virtual machine. And Docker can make use of this to run the application at different security levels.

82. How can Docker container share resources?

We can run multiple Docker containers on same host. These containers can share Kernel resources. Each container runs on its own Operating System and it has its own user-space and libraries.

So in a way Docker container does not share resources within its own namespace. But the resources that are not in isolated namespace are shared between containers. These are the Kernel resources of host machine that have just one copy.

So in the back-end there is same set of resources that Docker Containers share.

83. What is the difference between Add and Copy command in a Dockerfile?

Both Add and Copy commands of Dockerfile can copy new files from a source location to a destination in Container's file path.

They behave almost same.

The main difference between these two is that Add command can also read the files from a URL.

As per Docker documentation, Copy command is preferable. Since Copy only supports copying local files to a Container, it is preferred over Add command.

84. What is Docker Entrypoint?

We use Docker Entrypoint to set the starting point for a command in a Docker Image.

We can use the entrypoint as a command for running an Image in the container.

E.g. We can define following entrypoint in docker file and run it as following command:

ENTRYPOINT ["mycmd"]

% docker run mycmd

85. What is ONBUILD command in Docker?

We use ONBUILD command in Docker to run the instructions that have to execute after the completion of current Dockerfile build.

It is used to build a hierarchy of images that have to be build after the parent image is built.

A Docker build will execute first ONBUILD command and then it will execute any other command in Child Dockerfile.

86. What is Build cache in Docker?

When we build an Image, Docker will process each line in Dockerfile. It will execute the commands on each line in the order that is mentioned in the file.

But at each line, before running any command, Docker will check if there is already an existing image in its cache that can be reused rather than creating a new image.

This method of using cache in Docker is called Build cache in Docker.

We can also specify the option –no-cache=true to let Docker know that we do not want to use cache for Images. With this option, Docker will create all new images.

87. What are the most common instructions in Dockerfile?

Some of the common instructions in Dockerfile are as follows:

I. **FROM** : We use FROM to set the base image for subsequent instructions. In every valid Dockerfile, FROM is the first instruction.

II. **LABEL** : We use LABEL to organize our images as per project, module, licensing etc. We can also use LABEL to help in automation.

In LABEL we specify a key value pair that can be later used for programmatically handling the Dockerfile.

III. **RUN** : We use RUN command to execute any instructions in a new layer on top of the current image. With each RUN command we add something on top of the image and use it in subsequent steps in Dockerfile.

IV. **CMD** : We use CMD command to provide default values of an executing container. In a Dockerfile, if we include multiple CMD commands, then only the last instruction is used.

88. What is the purpose of EXPOSE command in Dockerfile?

We use EXPOSE command to inform Docker that Container will listen on a specific network port during runtime.

But these ports on Container may not be accessible to the host. We can use –p to publish a range of ports from Container.

89. What are the different kinds of namespaces available in a Container?

In a Container we have an isolated environment with namespace for each resource that a kernel provides. There are mainly six types of namespaces in a Container.

I. **UTS Namespace** : UTS stands for Unix Timesharing System. In UTS namespace every container gets its own hostname and domain name.

II. **Mount Namespace** : This namespace provides its own file systemwithin a container. With this namespace we get root like / in the file systemon which rest of the file structure is based.

III. **PID Namespace** : This namespace contains all the processes that run within a Container. We can run ps command to see the processes that are running within a Docker container.

IV. **IPC Namespace** : IPC stands for Inter Process Communication. This namespace covers shared memory, semaphores, named pipes etc resources that are shared by processes. The items in this namespace do not cross the container boundary.

V. **User Namespace** : This namespace contains the users and groups that are defined within a container.

VI. **Network Namespace** : With this namespace, container provides its own network resources like- ports, devices etc. With this namespace, Docker creates an independent network stack within each container.

90. How will you monitor Docker in production?

Docker provides tools like docker stats and docker events to monitor Docker in production.

We can get reports on important statistics with these commands.

Docker stats : When we call docker stats with a container id, we get the CPU, memory usage etc of a container. It is similar to top command in Linux.

Docker events : Docker events are a command to see the stream of activities that are going on in Docker daemon.

Some of the common Docker events are: attach, commit, die, detach, rename, destroy etc.

We can also use various options to limit or filter the events that we are interested in.

91. What are the Cloud platforms that support Docker?

Some of the popular cloud platforms that support Docker are:

I. Amazon AWS

II. Google Cloud Platform

III. Microsoft Azure

IV. IBM Bluemix

92. How can we control the startup order of services in Docker compose?

In Docker compose we can use the depends_on option to control the startup order of services.

With compose, the services will start in the dependency order. Dependencies can be defined in the options like- depends_on, links, volumes_from, network_mode etc.

But Docker does not wait for until a container is ready.

93. Why Docker compose does not wait for a container to be ready before moving on to start next service in dependency order?

The problem with waiting for a container to be ready is that in a Distributed system, some services or hosts may become unavailable sometimes. Similarly during startup also some services may also be down.

Therefore, we have to build resiliency in our application. So that even if some services are down we can continue our work or wait for the service to become available again.

We can use wait-for-it or dockerize tools for building this kind of resiliency.

94. How will you customize Docker compose file for different environments?

In Docker compose there are two files docker-compose.yml and docker-compose.override.yml. We specify our base configuration in docker-compose.yml file. For any environment specific customization we use docker-compose.override.yml file.

We can specify a service in both the files. Docker compose will merge these files based on following rules:

For single value options, new value replaces the old value.

For multi-value options, compose will concatenate the both set of values.

We can also use extends field to extend a service configuration to multiple environments. With extends, child services can use the common configuration defined by parent service.

Cloud Computing Questions

95. What are the benefits of Cloud Computing?

There are ten main benefits of Cloud Computing:

I. **Flexibility** : The businesses that have fluctuating bandwidth demands need the flexibility of Cloud Computing. If you need high bandwidth, you can scale up your cloud capacity. When you do not need high bandwidth, you can just scale down. There is no need to be tied into an inflexible fixed capacity infrastructure.

II. **Disaster Recovery** : Cloud Computing provides robust backup and recovery solutions that are hosted in cloud. Due to this there is no need to spend extra resources on homegrown disaster recovery. It also saves time in setting up disaster recovery.

III. **Automatic Software Updates** : Most of the Cloud providers give automatic software updates. This reduces the extra task of installing new software version and always catching up with the latest software installs.

IV. **Low Capital Expenditure** : In Cloud computing the model is Pay as you Go. This means there is very less upfront capital expenditure. There is a variable payment that is based on the usage.

V. **Collaboration:** In a cloud environment, applications can be shared between teams. This increases collaboration and communication among team members.

VI. **Remote Work:** Cloud solutions provide flexibility of working remotely. There is no on site work. One can just connect from anywhere and start working.

VII. **Security:** Cloud computing solutions are more secure than regular onsite work. Data stored in local servers and computers is prone to security attacks. In Cloud Computing, there are very few loose ends. Cloud providers give a secure working environment to its users.

VIII. **Document Control:** Once the documents are stored in a common repository, it increases the visibility and transparency among companies and their clients. Since there is one shared copy, there are fewer chances of discrepancies.

IX. **Competitive Pricing:** In Cloud computing there are multiple players, so they keep competing among themselves and provide very good pricing. This comes out much cheaper compared to other options.

X. **Environment Friendly:** Cloud computing saves precious environmental resources also. By not blocking the resources and bandwidth.

96. What is On-demand computing in Cloud Computing?

On-demand Computing is the latest model in enterprise systems. It is related to Cloud computing. It means IT resources can be provided on demand by a Cloud provider.

In an enterprise systemdemand for computing resources varies from time to time. In such a scenario, On-demand computing makes sure that servers and IT resources are provisioned to handle the increase/decrease in demand.

A cloud provider maintains a pol of resources. The pool of resources contains networks, servers, storage, applications and services. This pool can serve the varying demand of resources and computing by various enterprise clients.

There are many concepts like- grid computing, utility computing, autonomic computing etc. that are similar to on-demand computing.

This is the most popular trend in computing model as of now.

97. What are the different layers of Cloud computing?

Three main layers of Cloud computing are as follows:

I. **Infrastructure as a Service (IAAS):** IAAS providers give low-level abstractions of physical devices. Amazon Web Services (AWS) is an example of IAAS. AWS provides EC2 for computing, S3 buckets for storage etc. Mainly the resources in this layer are hardware like memory, processor speed, network bandwidth etc.

II. **Platformas a Service (PAAS):** PAAS providers offer managed services like Rails, Django etc. One good example of PAAS is Google App Engineer. These are the environments in which developers can develop sophisticated software with ease.

Developers just focus on developing software, whereas scaling and performance is handled by PAAS provider.

III. **Software as a Service (SAAS)** : SAAS provider offer an actual working software application to clients. Salesforce and Github are two good examples of SAAS. They hide the underlying details of the software and just provide an interface to work on the system. Behind the scenes the version of Software can be easily changed.

98. What resources are provided by Infrastructure as a Service (IAAS) provider?

An IAAS provider can give physical, virtual or both kinds of resources. These resources are used to build cloud.

IAAS provider handles the complexity of maintaining and deploying these services.

IAAS provider also handles security and backup recovery for these services. The main resources in IAAS are servers, storage, routers, switches and other related hardware etc.

99. What is the benefit of Platform as a Service?

Platform as a service (PaaS) is a kind of cloud computing service. A PaaS provider offers a platform on which clients can develop, run and manage applications without the need of building the infrastructure.

In PAAS clients save time by not creating and managing infrastructure environment associated with the app that they want to develop.

100. What are the main advantages of PaaS?

The advantages of PaaS are:

I. It allows development work on higher level programming with very less complexity.
II. Teams can focus on just the development of the application that makes the application very effective.
III. Maintenance and enhancement of the application is much easier.
IV. It is suitable for situations in which multiple developers work on a single project but are not co-located.

101. What is the main disadvantage of PaaS?

Biggest disadvantage of PaaS is that a developer can only use the tools that PaaS provider makes available. A developer cannot use the full range of conventional tools.

Some PaaS providers lock in the clients in their platform. This also decreases the flexibility of clients using PaaS.

102. What are the different deployment models in Cloud computing?

Cloud computing supports following deployment models:

I. **Private Cloud:** Some companies build their private cloud. A private cloud is a fully functional platform that is owned, operated and used by only one organization.

   Primary reason for private cloud is security. Many companies feel secure in private cloud. The other reasons for building private cloud are strategic decisions or control of operations.

   There is also a concept of Virtual Private Cloud (VPC). In VPC, private cloud is built and operated by a hosting company. But it is exclusively used by one organization.

II. **Public Cloud:** There are cloud platforms by some companies that are open for general public as well as big companies for use and deployment. E.g. Google Apps, Amazon Web Services etc.

   The public cloud providers focus on layers and application like- cloud application, infrastructure management etc. In this model resources are shared among different organizations.

III. **Hybrid Cloud:** The combination of public and private cloud is known as Hybrid cloud. This approach provides benefits of both the approaches- private and public cloud. So it is very robust platform.

A client gets functionalities and features of both the cloud platforms. By using Hybrid cloud an organization can create its own cloud as well as they can pass the control of their cloud to another third party.

103. What is the difference between Scalability and Elasticity?

Scalability is the ability of a system to handle the increased load on its current hardware and software resources. In a highly scalable system it is possible to increase the workload without increasing the resource capacity. Scalability supports any sudden surge in the demand/traffic with current set of resources.

Elasticity is the ability of a system to increase the workload by increasing the hardware/software resources dynamically. Highly elastic systems can handle the increased demand and traffic by dynamically commission and decommission resources. Elasticity is an important characteristic of Cloud Computing applications. Elasticity means how well your architecture is adaptable to workload in real time.

E.g. If in a system, one server can handle 100 users, 2 servers can handle 200 users and 10 servers can handle 1000 users. But in case for adding every X users, if you need 2X the amount of servers, then it is not a scalable design.

Let say, you have just one user login every hour on your site. Your one server can handle this load. But, if suddenly, 1000 users login at once, can your system quickly start new web servers on the fly to handle this load? Your design is elastic if it can handle such sudden increase in traffic so quickly.

104. What is Software as a Service?

Software as Service is a category of cloud computing in which Software is centrally hosted and it is licensed on a subscription basis. It is also known as On-demand software. Generally, clients access the software by using a thin-client like a web browser.

Many applications like Google docs, Microsoft office etc. provide SaaS model for their software.

The benefit of SaaS is that a client can add more users on the fly based on its current needs. And client does not need to install or maintain any software on its premises to use this software.

105. What are the different types of Datacenters in Cloud computing?

Cloud computing consists of different types of Datacenters linked in a grid structure. The main types of Datacenters in Cloud computing are:

I. **Containerized Datacenter**

As the name suggests, containerized datacenter provides high level of customization for an organization. These are traditional kind of datacenters. We can choose the different types of servers, memory, network and other infrastructure resources in this datacenter. Also we have to plan temperature control, network management and power management in this kind of datacenter.

II. **Low-Density Datacenters**

In a Low-density datacenter, we get high level of performance. In such a datacenter if we increase the density of servers, the issue with power comes. With high density of servers, the area gets heated. In such a scenario, effective heat and power management is done. To reach high level of performance, we have to optimize the number of servers' in the datacenter.

106. Explain the various modes of Software as a Service (SaaS) cloud environment?

Software as a Service (SaaS) is used to offer different kinds of software applications in a Cloud environment. Generally these are offered on subscription basis. Different modes of SaaS are:

I. **Simple multi-tenancy** : In this setup, each client gets its own resources. These resources are not shared with other clients. It is more secure option, since there is no sharing of resources. But it an inefficient option, since for each client more money is needed to scale it with the rising demands. Also it takes time to scale up the application in this mode.

II. **Fine grain multi-tenancy** : In this mode, the feature provided to each client is same. The resources are shared among multiple clients. It is an efficient mode of cloud service, in which data is kept private among different clients but computing resources are shared. Also it is easier and quicker to scale up the SaaS implementation for different clients.

107. What are the important things to care about in Security in a cloud environment?

In a cloud-computing environment, security is one of the most important aspects.

With growing concern of hacking, every organization wants to make its software system and data secure. Since in a cloud computing environment, Software and hardware is not on the premises of an organization, it becomes more important to implement the best security practices.

Organizations have to keep their Data most secure during the transfer between two locations. Also they have to keep data secure when it is stored at a location. Hackers can hack into application or they can get an unauthorized copy of the data. So it becomes important to encrypt the data during transit as well as during rest to protect it from unwanted hackers.

108. Why do we use API in cloud computing environment?

Application Programming Interfaces (API) is used in cloud computing environment for accessing many services. APIs are very easy to use. They provide a quick option to create different set of applications in cloud environment.

An API provides a simple interface that can be used in multiple scenarios.

There are different types of clients for cloud computing APIs. It is easier to serve different needs of multiple clients with APIs in cloud computing environment.

109. What are the different areas of Security Management in cloud?

Different areas of Security management in cloud are as follows:

I. **Identity Management** : This aspect creates different level of users, roles and their credentials to access the services in cloud.

II. **Access Control** : In this area, we create multiple levels of permissions and access areas that can be given to a user or role for accessing a service in cloud environment.

III. **Authentication** : In this area, we check the credentials of a user and confirm that it is the correct user. Generally this is done by user password and multi-factor authentication like-verification by a one-time use code on cell phone.

IV. **Authorization** : In this aspect, we check for the permissions that are given to a user or role. If a user is authorized to access a service, they are allowed to use it in the cloud environment.

110. What are the main cost factors of cloud based data center?

Costs in a Cloud based data center are different from a traditional data center. Main cost factors of cloud based data center are as follows:

I. **Labor cost** : We need skilled staff that can work with the cloud-based datacenter that we have selected for our operation. Since cloud is not a very old technology, it may get difficult to get the right skill people for handling cloud based datacenter.

II. **Power cost** : In some cloud operations, power costs are borne by the client. Since it is a variable cost, it can increase with the increase in scale and usage.

III. **Computing cost** : The biggest cost in Cloud environment is the cost that we pay to Cloud provider for giving us computing resources. This cost is much higher compared to the labor or power costs.

111. How can we measure the cloud-based services?

In a cloud-computing environment we pay for the services that we use. So main criteria to measure a cloud based service its usage.

For computing resource we measure by usage in terms of time and the power of computing resource.

For a storage resource we measure by usage in terms of bytes (giga bytes) and bandwidth used in data transfer.

Another important aspect of measuring a cloud service is its availability. A cloud provider has to specify the service level agreement (SLA) for the time for which service will be available in cloud.

112. How a traditional datacenter is different from a cloud environment?

In a traditional datacenter the cost of increasing the scale of computing environment is much higher than a Cloud computing environment. Also in a traditional data center, there are not much benefits of scaling down the operation when demand decreases. Since most of the expenditure is in capital spent of buying servers etc., scaling down just saves power cost, which is very less compared to other fixed costs.

Also in a Cloud environment there is no need to higher a large number of operations staff to maintain the datacenter. Cloud provider takes care of maintaining and upgrading the resources in Cloud environment.

With a traditional datacenter, people cost is very high since we have to hire a large number of technical operation people for in-house datacenter.

113. How will you optimize availability of your application in a Cloud environment?

In a Cloud environment, it is important to optimize the availability of an application by implementing disaster recovery strategy. For disaster recovery we create a backup application in another location of cloud environment. In case of complete failure at a data center we use the disaster recovery site to run the application.

Another aspect of cloud environment is that servers often fail or go down. In such a scenario it is important to implement the application in such a way that we just kill the slow server and restart another server to handle the traffic seamlessly.

114. What are the requirements for implementing IaaS strategy in Cloud?

Main requirements to implement IAAS are as follows:

I. **Operating System(OS):** We need an OS to support hypervisor in IaaS. We can use open source OS like Linux for this purpose.

II. **Networking** : We have to define and implement networking topology for IaaS implementation. We can use public or private network for this.

III. **Cloud Model** : We have to select the right cloud model for implementing IaaS strategy. It can be SaaS, PaaS or CaaS.

115. What is the scenario in which public cloud is preferred over private cloud?

In a startup mode often we want to test our idea. In such a scenario it makes sense to setup application in public cloud. It is much faster and cheaper to use public cloud over private cloud.

Remember security is a major concern in public cloud. But with time and changes in technology, even public cloud is very secure.

116. Do you think Cloud Computing is a software application or a hardware service?

Cloud Computing is neither a software application nor a hardware service. Cloud computing is a system architecture that can be used to implement software as well as hardware strategy of an organization.

Cloud Computing is a highly scalable, highly available and cost effective solution for software and hardware needs of an application.

Cloud Computing provides great ease of use in running the software in cloud environment. It is also very fast to implement compared with any other traditional strategy.

117. Why companies now prefer Cloud Computing architecture over Client Server Architecture?

In Client Server architecture there is one to one communication between client and server. Server is often at in-house datacenter and client can access same server from anywhere. If client is at a remote location, the communication can have high latency.

In Cloud Computing there can be multiple servers in the cloud. There will be a Cloud controller that directs the requests to right server node. In such a scenario clients can access cloud-based service from any location and they can be directed to the one nearest to them.

Another reason for Cloud computing architecture is high availability. Since there are multiple servers behind the cloud, even if one server is down, another server can serve the clients seamlessly.

118. What are the main characteristics of Cloud Computing architecture?

Main characteristics of Cloud Computing architecture are as follows:

I. **Elasticity** : In Cloud Computing system is highly elastic in the sense that it can easily adapt itself to increase or decrease in load. There is no need to take urgent actions when there is surge in traffic requests.

II. **Self-service provisioning** : In Cloud environment users can provision new resources on their own by just calling some APIs. There is no need to fill forms and order actual hardware from vendors.

III. **Automated de-provisioning** : In case demand/load decreases, extra resources can be automatically shut down in Cloud computing environment.

IV. **Standard Interface** : There are standard interfaces to start, stop, suspend or remove an instance in Cloud environment. Most of the services are accessible via public and standard APIs in Cloud computing.

V. **Usage based Billing** : In a Cloud environment, users are charged for their usage of resources. They can forecast their bill and costs based on the growth they are expecting in their load.

119. How databases in Cloud computing are different from traditional databases?

In a Cloud environment, companies often use different kind of data to store. There are data like email, images, video, pdf, graph etc. in a Cloud environment. To store this data often NoSQL databases are used.

A NoSQL database like MongoDB provides storage and retrieval of data that cannot be stored efficiently in a traditional RDBMS.

Database like Neo4J provides features to store graph data like Facebook, LinkedIn etc. in a cloud environment.

Hadoop like database help in storing Big Data based information. It can handle very large-scale information that is generated in a large-scale environment.

120. What is Virtual Private Network (VPN)?

In a Cloud environment, we can create a virtual private network (VPM) that can be solely used by only one client. This is a secure network in which data transfer between servers of same VPN is very secure.

By using VPN, an organization uses the public network in a private manner. It increases the privacy of an organization's data transfer in a cloud environment.

121. What are the main components of a VPN?

Virtual Private Network (VPN) consists of following main components:

I. **Network Access Server (NAS):** A NAS server is responsible for setting up tunnels in a VPN that is accesses remotely. It maintains these tunnels that connect clients to VPN.

II. **Firewall** : It is the software that creates barrier between VPN and public network. It protects the VPN from malicious activity that can be done from the outside network.

III. **AAA Server** : This is an authentication and authorization server that controls the access and usage of VPN. For each request to use VPN, AAA server checks the user for correct permissions.

IV. **Encryption** : In a VPN, encryption algorithms protect the important private data from malicious users.

122. How will you secure the application data for transport in a cloud environment?

With ease of use in Cloud environment comes the important aspect of keeping data secure. Many organizations have data that is transferred from their traditional datacenter to Cloud datacenter.

During the transit of data it is important to keep it secure. Once of the best way to secure data is by using HTTPS protocol over Secure Socket Layer (SSL).

Another important point is to keep the data always encrypted. This protects data from being accessed by any unauthorized user during transit.

123. What are the large-scale databases available in Cloud?

In Cloud computing scale is not a limit. So there are very large-scale databases available from cloud providers. Some of these are:

I. **Amazon DynamoDB** : Amazon Web Services (AWS) provides a NoSQL web service called DynamoDB that provides highly available and partition tolerant database system. It has a multi-master design. It uses synchronous replication across multiple datacenters. We can easily integrate it with MapReduce and Elastic MapReduce of AWS.

II. **Google Bigtable** : This is a very large-scale high performance cloud based database option from Google. It is available on Google Cloud. It can be scaled to peta bytes. It is a Google proprietary implementation. In Bigtable, two arbitrary string values, row key and column key, and timestamp are mapped to an arbitrary byte array. In Bigtable MapReduce algorithm is used for modifying and generating the data.

III. **Microsoft Azure SQL Database** : Microsoft Azure provides cloud based SQL database that can be scaled very easily for increased demand. It has very good security features and it can be even used to build multi-tenant apps to service multiple customers in cloud.

124. What are the options for open source NoSQL database in a Cloud environment?

Most of the cloud-computing providers support Open Source NoSQL databases. Some of these databases are:

I. **Apache CouchDB** : It is a document based NoSQL database from Apache Open Source. It is compatible with Couch Replication Protocol. It can communicate in native JSON and can store binary data very well.

II. **HBase** : It is a NoSQL database for use with Hadoop based software. It is also available as Open Source from Apache. It is a scalable and distributed Big Data database.

III. **MongoDB** : It is an open source database system that offers a flexible data model that can be used to store various kinds of data. It provides high performance and always-on user experience.

125. What are the important points to consider before selecting cloud computing?

Cloud computing is a very good option for an organization to scale and outsource its software/hardware needs. But before selecting a cloud provider it is important to consider following points:

I. **Security** : One of the most important points is security of the data. We should ask the cloud provider about the options to keep data secure in cloud during transit and at rest.

II. **Data Integrity** : Another important point is to maintain the integrity of data in cloud. It is essential to keep data accurate and complete in cloud environment.

III. **Data Loss** : In a cloud environment, there are chances of data loss. So we should know the provisions to minimize the data loss. It can be done by keeping backup of data in cloud. Also there should be reliable data recovery options in case of data loss.

IV. **Compliance** : While using a cloud environment one must be aware of the rules and regulations that have to be followed to use the cloud. There compliance issues with storing data of a user in an external provider's location/servers.

V. **Business Continuity** : In case of any disaster, it is important to create business continuity plans so that we can provide uninterrupted service to our end users.

VI. **Availability** : Another important point is the availability of data and services in a cloud-computing environment. It is very important to provide high availability for a good customer experience.

VII. **Storage Cost** : Since data is stored in cloud, it may be very cheap to store the data. But the real cost can come in transfer of data when we have to pay by bandwidth usage. So storage cost of data in cloud should also include the access cost of data transfer.

VIII. **Computing Cost** : One of the highest costs of cloud is computing cost. It can be very high cost with the increase of scale. So cloud computing options should be wisely considered in conjunction with computing cost charged for them.

126. What is a System integrator in Cloud computing?

Often an organization does not know all the options available in a Cloud computing environment. Here comes the role of a System Integrator (SI) who specializes in implementing Cloud computing environment.

SI creates the strategy of cloud setup. It designs the cloud platform for the use of its client. It creates the cloud architecture for the business need of client.

SI oversees the overal implementation of cloud strategy and plan. It also guides the client while choosing the right options in cloud computing platform.

127. What is virtualization in cloud computing?

Virtualization is the core of cloud computing platform. In cloud we can create a virtual version of hardware, storage and operating system that can be used to deploy the application.

A cloud provider gives options to create virtual machines in cloud that can be used by its clients. These virtual machines are much cheaper than buying a few high end computing machines.

In cloud we can use multiple cheap virtual machines to implement a resilient software system that can be scaled very easily in quick time. Where as buying an actual high-end machine to scale the system is very costly and time taking.

128. What is Eucalyptus in a cloud environment?

Eucalyptus is an open source software to build private and hybrid cloud in Amazon Web Services (AWS).

It stands for Elastic Utility Computing Architecture for Linking Your Programs To Useful Systems.

We can create our own datacenter in a private cloud by using Eucalyptus. It makes use of pooling the computing and storage resources to scale up the operations.

In Eucalyptus, we create images of software applications. These images are deployed to create instances. These instances are used for computing needs.

A Eucalyptus instance can have both public and private ip addresses.

129. What are the main components of Eucalyptus cloud architecture?

The main components of Eucalyptus cloud architecture are as follows:

I. **Cloud Controller (CLC)** : This is the controler that manages virtual resources like servers, network and storage. It is at the highest level in hierarchy. It is a Java programwith web interface for outside world. It can do resource scheduling as wel as

systemaccounting. There is only one CLC per cloud. It can handle authentication, accounting, reporting and quota management in cloud.

II. **Walrus** : This is another Java program in Eucalyptus that is equivalent to AWS S3 storage. It provides persistent storage. It also contains images, volumes and snapshots similar to AWS. There is only one Walrus in a cloud.

III. **Cluster Controller (CC)** : It is a C programthat is the front end for a Eucalyptus cloud cluster. It can communicate with Storage controller and Node controller. It manages the instance execution in cloud.

IV. **Storage Controller (SC)** : It is a Java programequivalent to EBS in AWS. It can interface with Cluster Controller and Node Controller to manage persistent data via Walrus.

V. **Node Controller (NC)** : It is a C programthat can host a virtual machine instance. It is at the lowest level in Eucalyptus cloud. It downloads images from Walrus and creates an instance for computing requirements in cloud.

VI. **VMWare Broker** : It is an optional component in Eucalyptus. It provides AWS compatible interface to VMWare environment.

130. What is Auto-scaling in Cloud computing?

Amazon Web Services (AWS) provides an important feature called Auto-scaling in the cloud. With Auto-scaling setup we can automatically provision and start new instances in AWS cloud without any human intervention.

Auto-scaling is triggered based on load and other metrics.

Let say if the load reaches a threshold we can setup auto-scaling to kick in and start a new server to handle additional load.

131. What are the benefits of Utility Computing model?

Utility computing is a cloud service model in which provider gives computing resources to users for using on need basis.

Some of the main benefits of Utility computing are:

I. **Pay per use** : Since a user pays for only usage, the cost of Utility computing is pay per use. We pay for the number of servers of instances that we use in cloud.

II. **Easy to Scale** : It is easier to scale up the operations in Utility computing. There is no need to plan for time consuming and costly hardware purchase.

III. **Maintenance** : In Utility computing maintenance of servers is done by cloud provider. So a user can focus on its core business. It need not spend time and resources on maintenance of servers in cloud.

Utility computing is also known as On-demand computing.

132. What is a Hypervisor in Cloud Computing?

Hypervisor is also known as virtual machine monitor (VMM). It is a computer software/hardware that can create and run virtual machines.

Hypervisor runs on a host machine. Each virtual machine is called Guest machine.

Hypervisor derives its name from termsupervisor, which is a traditional name for the kernel of an operating system.

Hypervisor provides a virtual operating platform to the guest operating system. It manages the execution of guest OS.

133. What are the different types of Hypervisor in Cloud Computing?

Hypervisors come in two main types:

I. **Type-1, native or bare-metal hypervisors** : Type 1 hypervisor runs directly on the hardware of host machine. It controls the guest operating system from host machine. It is also called bare metal hypervisor or native hypervisor.

Examples of Type-1 are: Xen, Oracle VM Server for SPARC, Oracle VM Server for x86, the Citrix XenServer, Microsoft Hyper-V and VMware ESX/ESXi.

II. **Type-2, hosted hypervisors:** Type 2 hypervisor runs like a regular computer programon an operating system. The guest operating system runs like a process on the host machine. It creates an abstract guest operating systemdiferent from the host operating system.

Examples of Type-2 are: VMware Workstation, VMware Player, VirtualBox, Parallels Desktop for Mac and QEMU are examples of type-2 hypervisors.

134. Why Type-1 Hypervisor has better performance than Type-2 Hypervisor?

Type-1 Hypervisor has better performance than Type-2 hypervisor because Type-1 hypervisor skips the host operating systemand it runs directly on host hardware. So it can utilize all the resources of host machine.

In cloud computing Type-1 hypervisors are more popular since Cloud servers may need to run multiple operating system images.

135. What is CaaS?

CaaS is also known as Communication as a Service. It is available in Telecom domain. One of the examples for CaaS is Voice Over IP (VoIP).

CaaS offers business features like desktop call control, unified messaging, and fax via desktop.

CaaS also provides services for Call Center automation like- IVR, ACD, call recording, multimedia routing and screen sharing.

136. How is Cloud computing different from computing for mobile devices?

Since Mobile devices are getting connected to the Internet in large numbers, we often use Cloud computing for Mobile devices.

In mobile applications, there can be sudden increase in traffic as well as usage. Even some applications become viral very soon. This leads to very high load on application.
In such a scenario, it makes sense to use Cloud Computing for mobile devices.

Also mobile devices keep changing over time, it requires standard interfaces of cloud computing for handling multiple mobile devices.

137. Why automation of deployment is very important in Cloud architecture?

One of the main reasons for selecting Cloud architecture is scalability of the system. In case of heavy load, we have to scale up the systemso that there is no performance degradation.

While scaling up the systemwe have to start new instances. To provision new instances we have to deploy our application on them.

In such a scenario, if we want to save time, it makes sense to automate the deployment process. Another term for this is Auto-scaling.

With a fully automated deployment process we can start new instances based on automated triggers that are raised by load reaching a threshold.

138. What are the main components in Amazon Cloud?

Amazon provides a wide range of products in Amazon Web Services for implementing Cloud computing architecture. In AWS some of the main components are as follows:

I. **Amazon EC2** : This is used for creating instances and getting computing power to run applications in AWS.

II. **Amazon S3** : This is a Simple Storage Service from AWS to store files and media in cloud.

III. **Amazon DynamoDB** : It is the database solution by AWS in cloud. It can store very large-scale data to meet needs of even BigData computing.

IV. **Amazon Route53** : This is a cloud based Domain Name System(DNS) service from AWS.

V. **Amazon Elastic Load Balancing (ELB):** This component can be used to load balance the various nodes in AWS cloud.

VI. **Amazon CodeDeploy** : This service provides feature to automate the code deployment to any instance in AWS.

139. What are main components in Google Cloud?

Google is a newer cloud alternative than Amazon. But Google provides many additional features than AWS. Some of the main components of Google Cloud are as follows:

I. **Compute Engine** : This component provides computing power to Google Cloud users.

II. **Cloud Storage** : As the name suggests this is a cloud storage solution from Google for storing large files for application use or just serving over the Internet.

III. **Cloud Bigtable** : It is a Google proprietary database from Google in Cloud. Now users can use this unique database for creating their applications.

IV. **Cloud Load Balancing** : This is a cloud-based load balancing service from Google.

V. **BigQuery** : It is a data-warehouse solution from Google in Cloud to perform data analytics of large scale.

VI. **Cloud Machine Learning Platform** : It is a powerful cloud based machine learning product from Google to perform machine learning with APIs like- Job Search, Text Analysis, Speech Recognition, Dynamic translation etc.

VII. **Cloud IAM** : This is an Identity and Access management tool from Google to help administrators run the security and authorization/authentication policies of an organization.

140. What are the major offerings of Microsoft Azure Cloud?

Microsoft is a relatively new entrant to Cloud computing with Azure cloud offering. Some of the main products of Microsoft cloud are as follows:

I. **Azure Container Service** : This is a cloud computing service from Microsoft to run and manage Docker based containers.

II. **StorSimple** : It is a Storage solution from Microsoft for Azure cloud.

III. **App Service** : By using App Services, users can create Apps for mobile devices as well as websites.

IV. **SQL Database** : It is a Cloud based SQL database from Microsoft.

V. **DocumentDB** : This is a NoSQL database in cloud by Microsoft.

VI. **Azure Bot Service** : We can use Azure Bot Service to create serverless bots that can be scaled up on demand.

VII. **Azure IoT Hub** : It is a solution for Internet of Things services in cloud by Microsoft.

141. What are the reasons of popularity of Cloud Computing architecture?

These days Cloud Computing is one of the most favorite architecture among organizations for their systems. Following are some of the reasons for popularity of Cloud Computing architecture:

I. **IoT** : With the Internet of Things, there are many types of machines joining the Internet and creating various types of interactions. In such a scenario, Cloud Computing serves well to provide scalable interfaces to communicate between the machines in IoT.

II. **Big Data** : Another major trend in today's computing is Big Data. With Big Data there is very large amount of user / machine data that is generated. Using in-house solution to handle Big Data is very costly and capital intensive. In Cloud Computing we can handle Big Data very easily since we do not have to worry about capital costs.

III. **Mobile Devices** : A large number of users are going to Mobile computing. With a mobile device users can access a service from any location. To handle wide-variety of mobile devices, standard interfaces of Cloud Computing are very useful.

IV. **Viral Content** : With growth of Social Media, content and media is getting viral i.e. It takes very short time to increase the traffic exponentially on a server. In such a scenario Auto-scaling of Cloud Computing architecture can handle such spikes very easily.

142. What are the Machine Learning options from Google Cloud?

Google provides a very rich library of Machine Learning options in Google Cloud. Some of these API are:

I. **Google Cloud ML** : This is a general purpose Machine Learning API in cloud. We can use pre-trained models or generate new models for machine learning with this option.

II. **Google Cloud Jobs API** : It is an API to link Job Seekers with Opportunities. It is mainly for job search based on skills, demand and location.

III. **Google Natural Language API** : This API can do text analysis of natural language content. We can use it for analyzing the content of blogs, websites, books etc.

IV. **Google Cloud Speech API** : It is a Speech Recognition API from Google to handle spoken text. It can recognize more than 80 languages and their related variants. It can even transcribe the user speech into written text.

V. **Google Cloud Translate API** : This API can translate content from one language to another language in cloud.

VI. **Google Cloud Vision API** : It is a powerful API for Image analysis. It can recognize faces and objects in an image. It can even categorize images in multiple relevant categories with a simple REST API call.

143. How will you optimize the Cloud Computing environment?

In a Cloud Computing environment we pay by usage. In such a scenario our usage costs are much higher. To optimize the Cloud Computing environment we have to keep a balance between our usage costs and usage.

If we are paying for computing instances we can choose options like Lambda in AWS, which is a much cheaper options for computing in cloud.

In case of Storage, if the data to be stored is not going to be accesses frequently we can go for Glacier option in AWS.

Similarly when we pay for bandwidth usage, it makes sense to implement a caching strategy so that we use less bandwidth for the content that is accessed very frequently.

It is a challenging task for an architect in cloud to match the options available in cloud with the budget that an organization has to run its applications.

Optimizations like server-less computing, load balancing, and storage selection can help in keeping the Cloud computing costs low with no degradation in User experience.

144. Do you think Regulations and Legal Compliance is an important aspect of Cloud Computing?

Yes, in Cloud Computing we are using resources that are owned by the Cloud provider. Due to this our data resides on the servers that can be shared by other users of Cloud.

There are regulations and laws for handling user data. We have to ensure that these regulations are met while selecting and implementing a Cloud computing strategy.

Similarly, if we are in a contract with a client to provide certain Service Level Agreement (SLA) performance, we have to implement the cloud

solution in such a way that there is no breach of SLA agreement due to Cloud provider's failures.

For security there are laws that have to be followed irrespective of Cloud or Co-located Data center. This is in the interest of our end-customer as well as for the benefit of business continuity.

With Cloud computing architecture we have to do due diligence in selecting Security and Encryption options in Cloud.

Unix Questions

145. How will you remove all files in current directory? Including the files that are two levels down in a sub-directory.

In Unix we have rmcommand to remove files and sub-directories. With rmcommand we have –r option that stands for recursive. The –r option can delete all files in a directory recursively.

It means if we our current directory structure is as follows:

My_dir

->Level_1_dir

-> Level_1_dir ->Level_2_dir

-> Level_1_dir ->Level_2_dir->a.txt

With rm–r * command we can delete the file a.txt as well as sub-directories Level_1_dir and Level_2_dir.

Command:

rm– r *

The asterisk (*) is a wild card character that stands for all the files with any name.

146. What is the difference between the –v and –x options in Bash shell scripts?

In a BASH Unix shell we can specify the options –v and –x on top of a script as follows:

#!/bin/bash -x –v

With –x option BASH shell will echo the commands like for, select, case etc. after substituting the arguments and variables. So it will be an expanded form of the command that shows all the actions of the script. It is very useful for debugging a shell script.

With –v option BASH shell will echo every command before substituting the values of arguments and variables. In –v option Unix will print each line as it reads.

In –v option, If we run the script, the shell prints the entire file and then executes. If we run the script interactively, it shows each command after pressing enter.

147. What is a Filter in Unix command?

In Unix there are many Filter commands like- cat, awk, grep, head, tail cut etc.

A Filter is a software programthat takes an input and produces an output, and it can be used in a streamoperation.

E.g. cut -d : -f 2 /etc/passwd | grep abc

We can mix and match multiple filters to create a complex command that can solve a problem.
Awk and Sed are complex filters that provide fully programmable features.

Even Data scientists use Unix filters to get the overview of data stored in the files.

148. What is Kernel in Unix operating system?

Kernel is the central core component of a Unix operating system (OS).

A Kernel is the main component that can control everything within Unix OS.

It is the first programthat is loaded on startup of Unix OS. Once it is loaded it will manage the rest of the startup process.

Kernel manages memory, scheduling as well as communication with peripherals like printers, keyboards etc.

But Kernel does not directly interact with a user. For a new task, Kernel will spawn a shell and user will work in a shell.

Kernel provides many systemcalls. A software program interacts with Kernel by using systemcalls.

Kernel has a protected memory area that cannot be overwritten accidentally by any process.

149. What is a Shell in Unix OS?

Shell in Unix is a user interface that is used by a user to access Unix services.

Generally a Unix Shell is a command line interface (CLI) in which users enter commands by typing or uploading a file.

We use a Shell to run different commands and programs on Unix operating system.

A Shell also has a command interpreter that can take our commands and send these to be executed by Unix operating system.

Some of the popular Shells on Unix are: Korn shell, BASH, C shell etc.

150. What are the different shells in Unix that you know about?

Unix has many flavors of Shell. Some of these are as follows:

- Bourne shell: We use sh for Bourne shell.
- Bourne Again shell: We use bash to run this shell.
- Korn shell: We can use ksh to for Korn shell.
- Z shell: The command to use this is zsh
- C shell: We use csh to run C shell.
- Enhanced C shell: tcsh is the command for enhanced C shell.

151. What is the first character of the output in ls –l command ?

We use ls -l command to list the files and directories in a directory. With -l option we get long listing format.

In this format the first character identifies the entry type. The entry type can be one of the following:

- b Block special file
- c Character special file
- d Directory
- l Symbolic link
- s Socket link
- p FIFO
- \- Regular file

In general we see d for directory and - for a regular file.

152. What is the difference between Multi-tasking and Multi-user environment?

In a Multi-tasking environment, same user can submit more than one tasks and operating system will execute them at the same time.

In a Multi-user environment, more than one user can interact with the operating system at the same time.

3. What is Command Substitution in Unix?

Command substitution is a mechanism by which Shell passes the output of a command as an argument to another command. We can even use it to set a variable or use an argument list in a for loop.

E.g. rm`cat files_to_delete`

In this example files_to_delete is a file containing the list of files to be deleted. cat command outputs this file and gives the output to rm command. rm command deletes the files.

In general Command Substitution is represented by back quotes `.

153. What is an Inode in Unix?

An Inode is a Data Structure in Unix that denotes a file or a directory on file system. It contains information about file like- location of file on the disk, access mode, ownership, file type etc.

Each Inode has a number that is used in the index table. Unix kernel uses Inode number to access the contents of an Inode.

We can use ls -i command to get the inode number of a file.

154. What is the difference between absolute path and relative path in Unix file system?

Absolute path is the complete path of a file or directory from the root directory. In general root directory is represented by / symbol. If we are in a directory and want to know the absolute path, we can use pwd command.

Relative path is the path relative the current location in directory.

E.g. In a directory structure /var/user/kevin/mail if we are in kevin directory then pwd command will give absolute path as /var/user/kevin.

Absolute path of mail folder is /var/user/kevin/mail. For mail folder ./mail is the relative path of mail directory from kevin folder.

155. What are the main responsibilities of a Unix Shell?

Some of the main responsibilities of a Unix Shell are as follows:

1. Program Execution: A shell is responsible for executing the commands and script files in Unix. User can either interactively enter the commands in Command Line Interface called terminal or they can run a script file containing a program.
2. Environment Setup: A shell can define the environment for a user. We can set many environment variables in a shell and use the value of these variables in our program.

3. Interpreter: A shell acts as an interpreter for our scripts. It has a built in programming language that can be used to implement the logic.

4. Pipeline: A shell also can hookup a pipeline of commands. When we run multiple commands separated by | pipe character, the shell takes the output of a command and passes it to next one in the pipeline.

5. I/O Redirection: Shell is also responsible for taking input from command line interface (CLI) and sending the output back to CLI. We use >, <, >> characters for this purpose.

156. What is a Shell variable?

A Unix Shell variable is an internal variable that a shell maintains. It is local to that Shell. It is not made available to the parent shell or child shell.

We generally use lower case names for shell variables in C shell.

We can set the value of a shell variable by set command.

E.g. % set max_threads=10

To delete a Shell variable we can use unset command.

To use a Shell variable in a script we use $ sign in front of the variable name.

E.g. echo $max_threads

157. What are the important Shell variables that are initialized on starting a Shell?

There are following important Shell variables that are automatically initialized when a Shell starts:

user:

term:

home:

path:

These Shell variables take values from environment variables.

If we change the value of these Shell variables then the corresponding environment variable value is also changed.

158. How will you set the value of Environment variables in Unix?

We can use 'setenv' command to set the value of environment variables.

E.g. % setenv [Name] [value]

% setenv MAX_TIME 10

To print the value of environment variable we can use 'printenv' command.

E.g. % printenv MAX_TIME

If we just use printenv then it lists all the environment variables and their values.

To unset or delete an environment variable we use unsetenv command.

E.g. % unsetenv MAX_TIME

To use an environment variable in a command we use the prefix $ with the name of variable.

What is the special rule about Shell and Environment variable in Bourne Shell?

In Bourne Shell, there is not much diff erence between Shell variable and Environment variable.

Once we start a Bourne Shell, it gets the value of environment variables and defines a corresponding Shell variable. Fromthat time onwards the shell only refers to Shell variable. But if a change is made to a Shell variable, then we have to explicitly export it to environment so that other shell or child processes can use it.

Also for Shell variables we use set and unset commands.

159. What is the difference between a System Call and a library function?

Systemcalls are low-level kernel calls. These are handled by the kernel. Systemcalls are implemented in kernel of Unix. An application has to execute special hardware and systemdependent instruction to run a Systemcall.
A library function is also a low level call but it is implemented in user space. A library call is a regular function call whose code resides in a shared library.

160. What are the networking commands in Unix that you have used?

Some of the popular networking commands in Unix that we use are as follows:

I. **ping** : We use this command to test the reachability of a host on an Internet Protocol (IP) network.

II. **telnet** : This is another useful command to access another machine on the network. This is command uses Telnet protocol.

III. **tracert** : This is short for Traceroute. It is a diagnostic command to display the route and transit delays of packets across Internet Protocol.

IV. **ftp** : We use ftp commands to transfer files over the network. ftp uses File Transfer Protocol.

V. **su** : This unix command is used to execute commands with the privileges of another user. It is also known as switch user, substitute user.

VI. **ssh** : This is a secure command that is preferred over Telnet for connecting to another machine. It creates a secure channel over an unsecured network. It uses cryptographic protocol to make the communication secure.

161. What is a Pipeline in Unix?

A Pipeline in Unix is a chain of commands that are connected through a stream in such a way that output of one command becomes input for another command.

E.g. ls –l | grep "abc" | wc –l

In the above example we have created pipeline of three commands ls, grep and wc.

First ls –l command is executed and gives the list of files in a directory. Then grep command searches for any line with word "abc" in it. Finally wc –l command counts the number of lines that are returned by grep command.

In general a Pipeline is uni-directional. The data flows from left to right direction.

162. What is the use of tee command in Unix?

We use tee command in a shell to read the input by user (standard input) and write it to screen (standard output) as well as to a file.

We can use tee command to split the output of a programso that it is visible on command line interface (CLI) as well as stored on a file for later use.

Syntax is tee [-a] [-i] [file …]

163. How will you count the number of lines and words in a file in Unix?

We can use wc (word count) command for counting the number of lines and words in a file. The wc command provides very good options for

collecting statistics of a file. Some of these options are:

l: This option gives line count

m: This option gives character count

c : This option gives byte count

w : This option gives word count

L: This option gives the length of the longest line

In case we give more than one files as input to wc command then it gives statistics for individual files as well as the total statistics for all files.

164. What is Bash shell?

Bash stands for Bourne Again Shell. It is free software written to replace Bourne shell.

We can see following line in shell scripts for Bash shell.

#!/bin/bash

In Bash we use ~/.profile at login to set environment variables.

In Bash we can execute commands in batch mode or concurrent mode.

In batch mode commands are separated by semi colon.

% command1; command2

In concurrent mode we separate commands by & symbol.

% command1 & command2

165. How will you search for a name in Unix files?

We can use grep command to search for a name or any text in a Unix file.

Grep stands for Global**l**y search a Regular Expression and Print.

Grep command can search for a text in one file as we**l** as multiple files.

We can also specify the text to be searched in regular expression pattern.

% grep ^z *.txt

Above command searches for lines starting with letter z in a**l** the .txt files in current directory.

166. What are the popular options of grep command in Unix?

In Unix, grep is one of the very useful commands. It provides many useful options. Some of the popular options are:

% grep –i : This option ignores case while doing search.

% grep –x : This option is used to search exact word in a file.

% grep –v: We use this option to find the lines that do not have the text we are searching.

% grep –A 10: This option displays 10 lines after the match is found.

% grep –c: We can use it to count the number of matching lines.

167. What is the difference between whoami and who am i commands in Unix?

Both the commands whoami and who ami are used to get the user information in Unix.

When we login as root user on the network, then both whoami and who ami commands will show the user as root.

But when any other user let say john logs in remotely and runs su –root, whoami will show root, but who ami will show the original user john.

168. What is a Superuser in Unix?

Superuser is a special user account. It is used for Unix systemadministration. This user can access all files on the file system. Also Superuser can also run any command on a system.

Generally Superuser permission is given to root user.

Most of the users work on their own user accounts. But when they need to run some additional commands, they can use su to switch to Superuser account.

It is a best practice to not use Superuser account for regular operations.

169. How will you check the information about a process in Unix?

We can use ps command to check the status of a process in Unix. It is short for Process Status.

On running ps command we get the list of processes that are executing in the Unix environment.

Generally we use ps –ef command. In this e stands for every process and f stands for full format.

This command gives us id of the process. We can use this id to kill the process.

170. What is the use of more command with cat command?

We generally use cat command to display the contents of a file.

If a file is very big then the contents of the file will not fit in screen, therefore screen will scroll forward and in the end we just see the last page of information from a file.

With more command we can pause the scrolling of data from a file in display. If we use cat command with more then we just see the first page of a file first. On pressing enter button, more command will keep changing the page. In this way it is easier to view information in a file.

When using the cat command to display file contents, large data that does not fit on the screen would scroll off without pausing, therefore making it difficult to view. On the other hand, using the more command is more appropriate in such case because it will display file contents one screen page at a time.

171. What are the File modes in Unix?

In Unix, there are three main permissions for a File.

I. r = It means a user can read the file
II. w = It means that a user can write to this file
III. x = It means the a user can execute a file like a shell script

Further there are three permission sets.

I. Owner: User who created the file
II. Group: This applies to user of a group to which owner belongs
III. Other: This is rest of the users in Unix system

With the combination of these three sets permissions of file in Unix are specified.

E.g. If a file has permissions –rwxr-xr-- , it means that owner has read, write, execute access. Group has read and execute access. Others have just read access. So the owner or admin has to specifically grant access to Others to execute the file.

172. We wrote a shell script in Unix but it is not doing anything. What could be the reason?

After writing a shell script we have to give it execute permission so that it can be run in Unix shell.

We can use chmod command to change the permission of a file in Unix. In general we use chmod +x to give execute permission to users for executing the shell script.

E.g. chmod +x abc.txt will give execute permission to users for executing the file abc.txt.

With chmod command we can also specify to which user/group the permission should be granted. The options are:

173. u is the owner user
174. g is the owner group
175. o is others
176. a is all users

177. What is the significance of 755 in chmod 755 command?

We use chmod command to change the permissions of a file in Unix. In this command we can pass the file permissions in the form of a three-digit number.

In this number 755, first digit 7 is the permissions given to owner, second digit 5 is the permissions of group and third digit 5 is the permissions of all others.

Also the numbers 7 and 5 are made from following rules:

4 = read permission

2 = write permission

1 = execute permission

So 7 = 4 + 2 + 1 = Read + Write + Execute permission

5 = 4 + 1 = Read + Execute permission

In out example 755 means, owner has read, write and execute permissions. Group and others have read and execute permissions.

178. How can we run a process in background in Unix? How can we kill a process running in background?

In Unix shell we can use symbol & to run a command in background.

E.g. % ls –lrt &

Once we use & option it runs the process in background and prints the process ID. We cannot down this process ID for using it in kill command.

We can also use ps –ef command to get the process ID of processes running in background.
Once we know the process ID of a process we can kill it by following command:

% kill -9 processId

179. How will you create a read only file in Unix?

We can create a file with Vi editor, cat or any other command. Once the file is created we have to give read only permissions to file. To change file permission to read only we use following command:

% chmod 400 filename

180. How does alias work in Unix?

We use alias in Unix to give a short name to a long command. We can even use it to combine multiple commands and give a short convenient name.

E.g. alias c='clear'

With this alias we just need to type c for running clear command.

In bash we store alias in .bash_profile file.

To get the list of all active alias in a shell we can run the alias command without any argument on command line.

% alias

alias h='history'

alias ki='kill -9'

alias l='last'

181. How can you redirect I/O in Unix?

In Unix we can redirect the output of command or operation to a file instead of command line interface (CLI). For this we sue redirection pointers. These are symbols > and >>.

If we want to write the output of ls –lrt command to a file we use following:

% ls –lrt > fileList.txt

If we want to copy one file to another file we use following:

% cat srcFile > copyFile

If we want to append the contents of one file at the end of another file we use following:

% cat srcFile >> appendToFile

182. What are the main steps taken by a Unix Shell for processing a command?

A Unix Shell takes following main steps to process a command:

I. **Parse** : First step is to parse the command or set of commands given in a Command Line Interface (CLI). In this step multiple consecutive spaces are replaced by single space. Multiple commands that are delimited by a symbol are divided into multiple individual actions.

II. **Variable** : In next step Shell identifies the variables mentioned in commands. Generally any word prefixed by $ sign is a variable.

III. **Command Substitution** : In this step, Shell executes the commands that are surrounded by back quotes and replaces that section with the output from the command.

IV. **Wild Card** : Once these steps are done, Shell replaces the Wild card characters like asterisk * with the relevant substitution.

V. **Execute** : Finally, Shell executes all the commands and follows the sequence in which Commands are given in CLI.

183. What is a Sticky bit in Unix?

A Sticky bit is a file/directory permission feature in Unix.

Sometimes when we give write permission to another user then that user can delete the file without the owner knowing about it. To prevent such an accidental deletion of file we use sticky bit.

When we mark a file/directory with a sticky bit, no user other than owner of file/directory gets the privilege to delete a file/directory.
To set the sticky bit we use following command:

% chmod +t filename

When we do ls for a file or directory, the entries with sticky bit are listed with letter t in the end of permissions.

E.g. % ls –lrt

-rwxrwxrwt 5 abc abc 4096 Jan 1 10:10 abc.txt

To remove the sticky bit we use following command:

% chmod –t filename

184. What are the different outputs from Kill command in Unix?

Kill command in Unix can return following outputs:

I. 0: It means Kill command was successful
II. -1: When we get -1 from Kill command it shows that there was some error. In addition to -1 we get EPERM or ESRCH in output.

EPERM denotes that system does not permit the process to be killed.

ESRCH denotes that process with PID mentioned in Kill command does not exist anymore. Or due to security restrictions we cannot access that process.

185. How will you customize your environment in Unix?

In Unix, almost all the popular shells provide options to customize the environment by using environment variables. To make these customizations permanent we can write these to special files that are specific to a user in a shell.

Once we write our customizations to these files, we keep on getting same customization when we open a new shell with same user account.

The special files for storing customization information for different shells at login time are:

I. C shell: /etc/.login or ~/.cshrc
II. TC shell: /etc/.login or ~/.tshrc
III. Korn shell: ~etc/ksh.kshrc
IV. Bash: ~/.bash_profile

186. What are the popular commands for user management in Unix?

In Unix we use following commands for User Management:

I. **id** : This command gives the active user id with login and groups to which user belongs.
II. **who** : This command gives the user that is currently logged on system. It also gives the time of login.
III. **last** : This command shows the previous logins to the system in a chronological order.
IV. **adduser** : We use this command to add a new user.
V. **groupadd** : We use this command to add a new group in the system.
VI. **usermod** : We user usermod command to add/remove a user to a group in Unix.

187. How will you debug a shell script in Unix?

A shell script is a programthat can be executed in Unix shell. Sometimes a shell script does not work as intended. To debug and find the problem in shell script we can use the options provided by shell to debug the script.

In bash shell there are x and v options that can be used while running a script.

% bash –xv <scriptName>

With option v all the input lines are printed by shell. With option x all the simple commands are printed in expanded format. We can see all the arguments passed to a command with –x option.

188. What is the difference between a Zombie and Orphan process in Unix?

Zombie is a defunct child process in Unix that still has entry in process table.

Sometimes a child process is terminated in Unix, but the parent process still waits on it.

A Zombie process is different from an Orphan process. An orphan process is a child process whose parent process had died. Once a process is orphan it is adopted by init process. So effectively it is not an orphan.

Therefore if a process exits without cleaning its child processes, they do not become Zombie. Instead init process adopts these child processes.

Zombie processes are the ones that are not yet adopted by init process.

189. How will you check if a remote host is still alive?

We can use one of the networking commands in Unix. It is called ping. With ping command we can ping a remote host.

Ping utility sends packets in an IP network with ICMP protocol. Once the packet goes from source to destination and comes back it records the time.

We can even specify the number of packets we want to send so that we collect more statistics to confirm the result.

% ping www.google.com

Another option is to use telnet to remote host to check its status.

190. How will you get the last executed command in Unix?

We can use history command to get the list commands that were executed in Unix. Since we are only interested in the last executed command we have to use tail to get the last entry.

Exact command would be as follows:

% history | tail -2

191. What is the meaning of "2>&1" in a Unix shell?

In Unix shell file descriptor 1 is for standard output.

File description 2 is for standard error.

We can use "2>&1" in a command so that all the errors from standard error go to standard output.

% cat file 2>&1

192. How will you find which process is taking most CPU time in Unix?

In Unix, we can use top command to list the CPU time and memory used by various processes. The top command lists the process IDs and CPU time, memory etc used by top most processes.

Top command keeps refreshing the screen at a specified interval. So we can see over the time which process is always appearing on the top most row in the result of top command.

This is the process that is consuming most CPU time.

193. What is the difference between Soft link and Hard link in Unix?

A soft link is a pointer to a file, directory or a program located in a different location. A hard link can point to a program or a file but not to a directory.

If we move, delete or rename a file, the soft link will be broken. But a hard link still remains after moving the file/program.

We use the command ln –s for creating a soft link. But a hard link can be created by ln command without –s option.

194. How will you find which processes are using a file?

We can use lsof command to find the list of Process IDs of the processes that are accessing a file in Unix.

Lsof stands for List Open Files.

Sample command is:

% lsof /var
It will list the processes that are accessing /var directory in current unix system.

We can use options –i, -n and –P for different uses.

% lsof –i will only list IP sockets.

195. What is the purpose of nohup in Unix?

In Unix, nohup command can be used to run a command in background. But it is different from & option to run a process in background.

Nohup stands for No Hangup. A nohup process does not stop even if the Unix user that started the process has logged out from the system.

But the process started with option & will stop when the user that started the process logs off.

196. How will you remove blank lines from a file in Unix?

We can use grep command for this option. Grep command gives –v option to exclude lines that do not match a pattern.

In an empty line there is nothing from start to end. In Grep command, ^ denotes that start of line and $ denotes the end of line.

% grep –v '^$' lists the lines that are empty from start to the end.

Once we get this result, we can use > operator to write the output to a new file. So exact command will be:

% grep –v '^$' file1.txt > file2.txt

197. How will you find the remote hosts that are connecting to your system on a specific port in Unix?

We can use netstat command for this purpose. Netstat command lists the statistics about network connections. We can grep for the port in which we are interested.
Exact command will be:

% netstst –a | grep "port number"

198. What is xargs in Unix?

We use xargs command to build and execute commands that take input from standard input. It is generally used in chaining of commands.

Xargs breaks the list of arguments into small sub lists that can be handled by a command.

Following is a sample command:

% find /path -type f -print | xargs rm

The above command uses find to get the list of all files in /path directory. Then xargs command passes this list to rmcommand so that they can be deleted.

Thanks!!