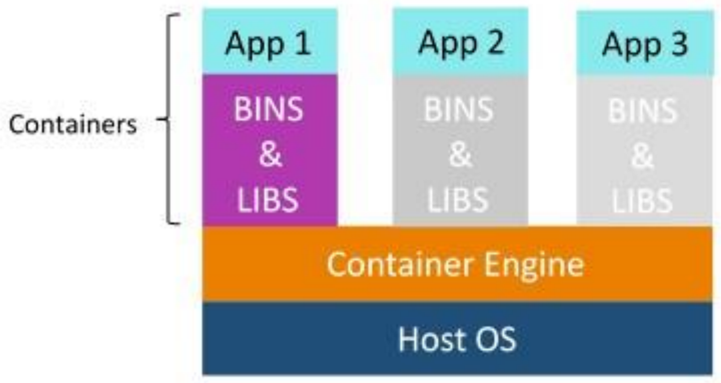
HUMMINGBIRD
Knowledge Hub

Containerization and Virtualization

Q1. What are containers?

My suggestion is to explain the need for containerization first, containers are used to provide consistent computing environment from a developer's laptop to a test environment, from a staging environment into production.

Now give a definition of containers, a container consists of an entire runtime environment: an application, plus all its dependencies, libraries and other binaries, and configuration files needed to run it, bundled into one package. Containerizing the application platform and its dependencies removes the differences in OS distributions and underlying infrastructure.



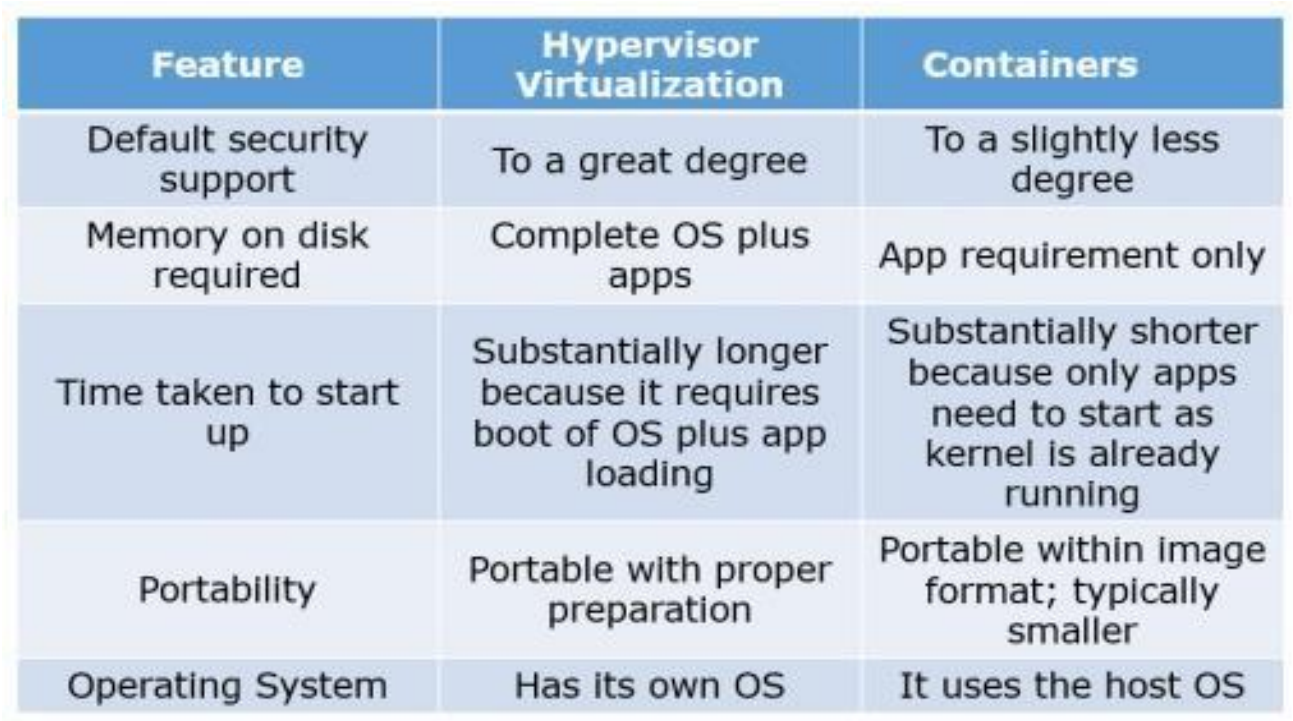
Q2. What are the advantages that Containerization provides over virtualization?

Below are the advantages of containerization over virtualization:

- Containers provide real-time provisioning and scalability but VMs provide slow provisioning
- Containers are lightweight when compared to VMs
- VMs have limited performance when compared to containers
- Containers have better resource utilization compared to VMs

Q3. How exactly are containers (Docker in our case) different from hypervisor virtualization (vSphere)? What are the benefits?

Given below are some differences. Make sure you include these differences in your answer:



Feature	Hypervisor Virtualization	Containers
Default security support	To a great degree	To a slightly less degree
Memory on disk required	Complete OS plus apps	App requirement only
Time taken to start up	Substantially longer because it requires boot of OS plus app loading	Substantially shorter because only apps need to start as kernel is already running
Portability	Portable with proper preparation	Portable within image format; typically smaller
Operating System	Has its own OS	It uses the host OS

Q4. What is Docker image?

I suggest that you go with the below mentioned flow:
Docker image is the source of Docker container. In other words, Docker images are used to create containers. Images are created with the build command, and they'll produce a container when started with run. Images are stored in a Docker registry such as registry.hub.docker.com because they can become quite large, images are designed to be composed of layers of other images, allowing a minimal amount of data to be sent when transferring images over the network.
Tip: Be aware of Dockerhub in order to answer questions on pre-available images.

Q5. What is Docker container?

This is a very important question so just make sure you don't deviate from the topic. I advise you to follow the below mentioned format:
Docker containers include the application and all of its dependencies but share the kernel with other containers, running as isolated processes in user space on the host operating

system. Docker containers are not tied to any specific infrastructure: they run on any computer, on any infrastructure, and in any cloud.
Now explain how to create a Docker container, Docker containers can be created by either creating a Docker image and then running it or you can use Docker images that are present on the Dockerhub.

Docker containers are basically runtime instances of Docker images.

Q6. What is Docker hub?

Answer to this question is pretty direct. Docker hub is a cloud-based registry service which allows you to link to code repositories, build your images and test them, stores manually pushed images, and links to Docker cloud so you can deploy images to your hosts. It provides a centralized resource for container image discovery, distribution and change management, user and team collaboration, and workflow automation throughout the development pipeline.

Q7. How is Docker different from other container technologies?

According to me, below points should be there in your answer:
Docker containers are easy to deploy in a cloud. It can get more applications running on the same hardware than other technologies, it makes it easy for developers to quickly create, ready-to-run containerized applications and it makes managing and deploying applications much easier. You can even share containers with your applications. If you have some more points to add you can do that but make sure the above the above explanation is there in your answer.

Q8. What is Docker Swarm?

You should start this answer by explaining Docker Swarn. It is native clustering for Docker which turns a pool of Docker hosts into a single, virtual Docker host. Docker Swarm serves the standard Docker API, any tool that already communicates with a
Docker daemon can use Swarm to transparently scale to multiple hosts.
I will also suggest you to include some supported tools:

- Dokku
- Docker Compose

- Docker Machine
- Jenkins

Q9. What is Dockerfile used for?

This answer according to me should begin by explaining the use of Dockerfile. Docker can build images automatically by reading the instructions from a Dockerfile.

Now I suggest you to give a small definition of Dockerfle. A Dockerfile is a text document that contains all the commands a user could call on the command line to assemble an image. Using docker build users can create an automated build that executes several command-line instructions in succession.

Now expect a few questions to test your experience with Docker.

Q10. Can I use json instead of yaml for my compose file in Docker?

You can use json instead of yaml for your compose file, to use json file with compose, specify the filename to use for eg: **docker-compose -f docker-compose.json up**

Q11. Tell us how you have used Docker in your past position?

Explain how you have used Docker to help rapid deployment. Explain how you have scripted Docker and used Docker with other tools like Puppet, Chef or Jenkins. If you have no past practical experience in Docker and have past experience with other tools in similar space, be honest and explain the same. In this case, it makes sense if you can compare other tools to Docker in terms of functionality.

Q12. How to create Docker container?

I will suggest you to give a direct answer to this. We can use Docker image to create Docker container by using the below command: **docker run -t -i <image name> <command name>** This command will create and start container.
You should also add, If you want to check the list of all running container with status on a host use the below command: **docker ps -a**

Q13. How to stop and restart the Docker container?

In order to stop the Docker container you can use the below command: **docker stop <container ID>**

Now to restart the Docker container you can use: **docker restart <container ID>**

Q14. How far do Docker containers scale?

Large web deployments like Google and Twitter, and platform providers such as Heroku and dotCloud all run on container technology, at a scale of hundreds of thousands or even millions of containers running in parallel.

Q15. What platforms does Docker run on?

I will start this answer by saying Docker runs on only Linux and Cloud platforms and then I will mention the below vendors of Linux:

- Ubuntu 12.04, 13.04 et al
- Fedora 19/20+
- RHEL 6.5+
- CentOS 6+
- Gentoo
- ArchLinux
- openSUSE 12.3+ CRUX 3.0+

Cloud:

- Amazon EC2
- Google Compute Engine
- Microsoft Azure
- Rackspace

Note that Docker does not run on Windows or Mac.

Q16. Do I lose my data when the Docker container exits?

You can answer this by saying, no I won't loose my data when Dcoker container exits. Any data that your application writes to disk gets preserved in its container until you explicitly delete the container. The file system for the container persists even after the container halts.

1. How does HTTP work?

The HTTP protocol works in a client and server model like most other protocols. A web browser using which a request is initiated is called as a client and a web server software which responds to that request is called a server. World Wide Web Consortium and the Internet Engineering Task Force are two important spokes in the standardization of the HTTP protocol. HTTP allows improvement of its request and response with the help of intermediates, for example a gateway, a proxy, or a tunnel. The resources that can be requested using the HTTP protocol, are made available using a certain type of URI (Uniform Resource Identifier) called a URL (Uniform Resource Locator). TCP (Transmission Control Protocol) is used to establish a connection to the application layer port 80 used by HTTP.

2. Explain your understanding and expertise on both the software development side and the technical operations side of an organization you've worked for in the past.

DevOps engineers almost always work in a 24/7 business critical online environment. I was adaptable to on-call duties and able to take up real-time, live-system responsibility. I successfully automated processes to support continuous software deployments. I have experience with public/private clouds, tools like Chef or Puppet, scripting and automation with tools like Python and PHP, and a background in Agile.

3. Discuss your experience building bridges between IT Ops, QA and development.

DevOps is all about effective communication and collaboration. I've been able to deal with production issues from the development and operations sides, effectively straddling the two worlds. I'm less interested in finding blame or playing the hero than I am with ensuring that all of the moving parts come together.

4. What types of testing are needed?

Software teams will often look for the “fair weather” path to system completion; that is, they start from an assumption that software will usually work and only occasionally fail.

I believe to practice defensive programming in a pragmatic way, which often means assuming that the code will fail and planning for those failures. I try to incorporate unit test strategy, use of test harnesses, early load testing; network simulation, A/B and multi-variate testing etc.

5. Give me an example of how you would handle projects?

As a professional with managerial responsibilities, I would demonstrate a clear understanding of DevOps project management tactics and also work with teams to set objectives, streamline workflow, maintain scope, research and introduce new tools or frameworks, translate requirements into workflow and follow up. I would resort to CI, release management and other tools to keep interdisciplinary projects on track.

6. What's your career objective in your role as a DevOps engineer?

My passion is breaking down the barriers and building and improving processes, so that the engineering and operations teams work better and smarter. That's why I love DevOps. It's an opportunity to be involved in the entire delivery system from start to finish.

7. How would you make software deployable?

The ability to script the installation and reconfiguration of software systems is essential towards controlled and automated change. Although there is an increasing trend for new software to enable this, older systems and products suffer from the assumption that changes would be infrequent and minor, and so make automated changes difficult. As a professional who appreciates the need to expose configuration and settings in a manner accessible to automation, I will work with concepts like Inversion of Control (IoC) and Dependency Injection, scripted installation, test harnesses, separation of concerns, command-line tools, and infrastructure as code.

8. What is the one most important thing DevOps helps do?

The most important thing DevOps helps do is to get the changes into production as quickly as possible while minimizing risks in software quality assurance and compliance. That is the primary objective of DevOps. However, there are many other positive sideeffects to DevOps. For example, clearer communication and better working relationships between teams which creates a less stressful working environment.

9. Which scripting languages do you think are most important for a DevOps engineer?

As far as scripting languages go, the simpler the better. In fact, the language itself isn't as important as understanding design patterns and development paradigms such as procedural, object-oriented, or functional programming.

10. How do you expect you would be required to multitask as a DevOps professional?

I believe I'll be expected to:

1. Focus attention on bridging communication gaps between Development and Operations teams.
2. Understand system design from an architect's perspective, software development from a developer's perspective,operations and infrastructure from the perspective of a seasoned Systems Administrator.
3. Execute – to be able to actually do what needs to be done.

11. What testing is necessary to ensure that a new service is ready for production?

DevOps is all about continuous testing throughout the process, starting with development through to production. Everyone shares the testing responsibility. This ensures that developers are delivering code that doesn't have any errors and is of high quality, and it also helps everyone leverage their time most effectively.

12. What's a PTR in DNS?

Pointer records are used to map a network interface (IP) to a host name. These are primarily used for reverse DNS. Reverse DNS is setup very similar to how normal (forward) DNS is setup. When you delegate the DNS forward, the owner of the domain tells the registrar to let your domain use specific name servers.

13. Describe two-factor authentication?

Two-factor authentication is a security process in which the user provides two means of identification from separate categories of credentials; one is typically a physical token, such as a card, and the other is typically something memorized, such as a security code.

14. Tell us about the CI tools that you are familiar with?

The premise of CI is to get feedback as early as possible because the earlier you get feedback, the less things cost to fix. Popular open source tools include Hudson, Jenkins, CruiseControl and CruiseControl.NET. Commercial tools include ThoughtWorks' Go, Urbancode's Anthill Pro, Jetbrains' Team City and Microsoft's Team Foundation Server.

15. What are the advantages of NoSQL database over RDBMS?

The advantages are:

1. Less need for ETL
2. Support for unstructured text
3. Ability to handle change over time
4. Breadth of functionality
5. Ability to scale horizontally
6. Support for multiple data structures
7. Choice of vendors

16. What is an MX record in DNS?

MX records are mail exchange records used for determining the priority of email servers for a domain. The lowest priority email server is the first destination for email. If the lowest priority email server is unavailable, mail will be sent to the higher priority email servers.

17. What is the difference between RAID 0 and RAID 1?

RAID 1 offers redundancy through mirroring, i.e., data is written identically to two drives. RAID 0 offers no redundancy and instead uses striping, i.e., data is split across all the drives. This means RAID 0 offers no fault tolerance; if any of the constituent drives fails, the RAID unit fails.

18. How would you prepare for a migration?

Tips to answer: This question evaluates your experience of real projects with all the awkwardness and complexity they bring. Include terms like cut-over, dress rehearsals, roll-back and roll-forward, DNS solutions, feature toggles, branch by abstraction, and automation in your answer. Developing greenfield systems with little or no existing technology in place is always easier than having to deal with legacy components and configuration. As a candidate if you appreciate that any interesting software system will in effect be under constant migration, you will appear suitable for the role.

19. What's your systems background?

Tips to answer: Some DevOps jobs require extensive systems knowledge, including server clustering and highly concurrent systems. As a DevOps engineer, you need to analyze system capabilities and implement upgrades for efficiency, scalability and stability, or resilience. It is recommended that you have a solid knowledge of OSes and supporting technologies, like network security, virtual private networks and proxy server configuration.

DevOps relies on virtualization for rapid workload provisioning and allocating compute resources to new VMs to support the next rollout, so it is useful to have in-depth knowledge around popular hypervisors. This should ideally include backup, migration and lifecycle management tactics to protect, optimize and eventually recover computing resources. Some environments may emphasize microservices software development tailored for virtual containers. Operations expertise must include extensive knowledge of systems management tools like Microsoft System Center, Puppet, Nagios and Chef. DevOps jobs with an emphasis on operations require detailed problem-solving, troubleshooting and analytical skills.

20. What DevOp tools have you worked with?

Tips to answer: Software configuration management and build/release (version control) tools, including Apache Subversion, Mercurial, Fossil and others, help document change requests. Developers can more easily follow the company's best practices and policies while software changes.

Continuous integration (CI) tools such as Rational Build Forge, Jenkins and Semaphore merge all developer copies of the working code into a central version. These tools are important for larger groups where teams of developers work on the same codebase simultaneously. QA experts use code analyzers to test software for bugs, security and performance. If you've used HP's Fortify Static Code Analyzer, talk about how it identified security vulnerabilities in coding languages. Also speak about tools like GrammaTech's CodeSonar that you used to identify memory leaks, buffer underruns and other defects for C/C++ and Java code. It is essential that you have adequate command of the principal languages like Ruby, C#, .NET, Perl, Python, Java, PHP, Windows PowerShell, and are comfortable with the associated OS environments Windows, Linux and Unix.

21. How much have you interacted with cloud based software development?

Tips to answer: Share your knowledge around use of cloud platforms, provisioning new instances, coding new software iterations with the cloud provider's APIs or software development kits, configuring clusters to scale computing capacity, managing workload lifecycles and so on. This is the perfect opportunity to discuss container-based cloud instances as an alternative to conventional VMs. Event-based cloud computing, such as AWS Lambda offers another approach to software development, a boon for experienced DevOps candidates. In your interview, mention experience handling big data, which uses highly scalable cloud infrastructures to tackle complex computing tasks.

22. What other tools are you familiar with that might help you in this role?

Tips to answer: DevOps is so diverse and inclusive that it rarely ends with coding, testing and systems. A DevOps project might rely on database platforms like SQL or NoSQL, data structure servers like Redis, or configuration and management issue tracking systems like Redmine. Web applications are popular for modern enterprises, making a background with Web servers, like Microsoft Internet Information Services, Apache Tomcat or other Web servers, beneficial. Make sure to bring across that you are familiar with Agile application lifecycle management techniques and tools.

23. Are you familiar with just Linux or have you worked with Windows environments as well?

Tips to answer: Demonstrate as much as you can, a clear understanding of both the environments including the key tools.

24. How can you reduce load time of a dynamic website?

Tips to answer: Talk about Webpage optimization, cached web pages, quality web hosting , compressed text files, Apache fine tuning.

25. Describe your experience implementing continuous deployment?

Tips to answer: Answer with a comprehensive list of all the tools that you used. Include inferences of the challenges you faced and how you tackled them.

26. How would you ensure traceability?

Tips to answer: This question probes your attitude to metrics, logging, transaction journeys, and reporting. You should be able to identify that metric, monitoring and logging needs to be a core part of the software system, and that without them, the software is essentially not going to be able to appear maintained and diagnosed. Include words like SysLog, Splunk, error tracking, Nagios, SCOM, Avicode in your answer.

27. What was your greatest achievement on a recent project?

Tips to answer: Make sure you demonstrate your perfect understanding of both development and operations. Do not let your answer lean towards one particular skillset ignoring the other. Even if you have worked in an environment wherein you had to work more with one skillset, assure the intervewer that you are agile according to the needs of your organization.

28. What problems did you face and how did you solve them in a way that met the team's goals?

Tips to answer: This questions aims to find out how much you can handle stress and nonconformity at work. Talk about your leadership skills to handle and motivate the team to solve problems together.Talk about CI, release management and other tools to keep interdisciplinary projects on track.

29. Are you more Dev or Ops?

Tips to answer: This is probably the trickiest question that you might face in the interview. Emphasize the fact that this depends a lot on the job, the company you are working for and the skills of people involved. You really have to be able to alternate between both sides of the fence at any given time. Talk about your experience and demonstrate how you are agile with both.

30. What special training or education did it require for you to become a DevOps engineer?

Tips to answer: DevOps is more of a mind-set or philosophy rather than a skill-set. The typical technical skills associated with DevOps Engineers today is Linux systems administration, scripting, and experience with one of the many continuous integration or configuration management tools like Jenkins and Chef. What it all boils down to is that whatever skill-sets you have, while important, are not as important as having the ability to learn new skills quickly to meet the needs. It’s all about pattern recognition, and having the ability to merge your experiences with current requirements.Proficiency in Windows and Linux systems administration, script development, an understanding