Q12. What are the three main variables that affect recursion and inheritance in Nagios?

According to me the proper format for this answer should be:
First name the variables and then a small explanation of each of these variables:

- Name
- Use
- Register

Then give a brief explanation for each of these variables. Name is a placeholder that is used by other objects. Use defines the "parent" object whose properties should be used. Register can have a value of 0 (indicating its only a template) and 1 (an actual object). The register value is never inherited.

Q13. What is meant by saying Nagios is Object Oriented?

Answer to this question is pretty direct. I will answer this by saying, "One of the features of Nagios is object configuration format in that you can create object definitions that inherit properties from other object definitions and hence the name. This simplifies and clarifies relationships between various components."

Q14. What is State Stalking in Nagios?

I will advise you to first give a small introduction on State Stalking. It is used for logging purposes. When Stalking is enabled for a particular host or service, Nagios will watch that host or service very carefully and log any changes it sees in the output of check results.
Depending on the discussion between you and interviewer you can also add, "It can be very helpful in later analysis of the log files. Under normal circumstances, the result of a host or service check is only logged if the host or service has changed state since it was last checked."

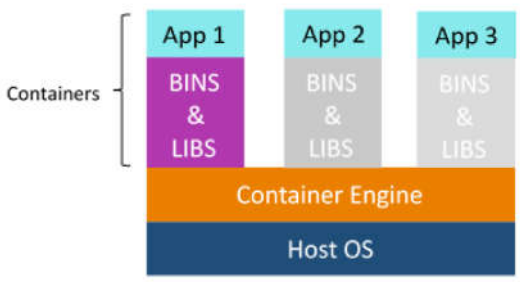
Want to get trained in monitoring tools like Nagios? Want to certified as a DevOps Engineer? Make sure you check out our DevOps Masters Program.

Containerization and Virtualization Interview Questions

Let's see how much you know about containers and VMs.

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.