Configuration Management Interview Questions

Now let’s check how much you know about Configuration Management.

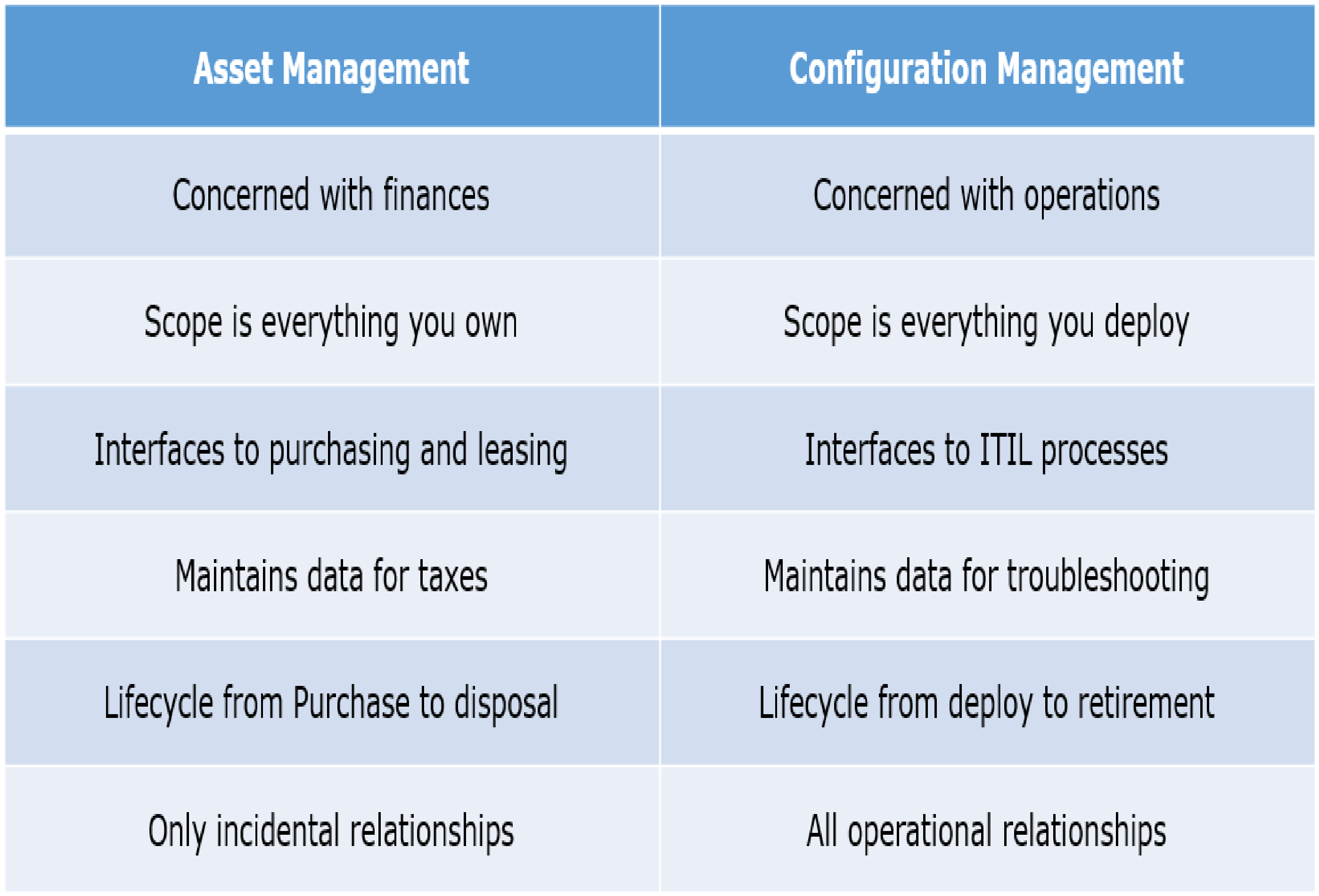
Q1. What are the goals of Configuration management processes?

The purpose of Configuration Management (CM) is to ensure the integrity of a product or system throughout its life-cycle by making the development or deployment process controllable and repeatable, therefore creating a higher quality product or system. The CM process allows orderly management of system information and system changes for purposes such as to:

- Revise capability,
- Improve performance,
- Reliability or maintainability,
- Extend life,
- Reduce cost,
- Reduce risk and
- Liability, or correct defects.

Q2. What is the difference between Asset management and Configuration Management?

Given below are few differences between Asset Management and Configuration Management:



Asset Management	Configuration Management
Concerned with finances	Concerned with operations
Scope is everything you own	Scope is everything you deploy
Interfaces to purchasing and leasing	Interfaces to ITIL processes
Maintains data for taxes	Maintains data for troubleshooting
Lifecycle from Purchase to disposal	Lifecycle from deploy to retirement
Only incidental relationships	All operational relationships

Q3. What is the difference between an Asset and a Configuration Item?

According to me, you should first explain Asset. It has a financial value along with a depreciation rate attached to it. IT assets are just a sub-set of it. Anything and everything that has a cost and the organization uses it for its asset value calculation and related benefits in tax calculation falls under Asset Management, and such item is called an asset. Configuration Item on the other hand may or may not have financial values assigned to it. It will not have any depreciation linked to it. Thus, its life would not be dependent on its financial value but will depend on the time till that item becomes obsolete for the organization.

Now you can give an example that can showcase the similarity and differences between both:
1) Similarity:
Server – It is both an asset as well as a CI.
2) Difference:
Building – It is an asset but not a CI.
Document – It is a CI but not an asset

Q4. What do you understand by “Infrastructure as code”? How does it fit into the DevOps methodology? What purpose does it achieve?

Infrastructure as Code (IAC) is a type of IT infrastructure that operations teams can use to automatically manage and provision through code, rather than using a manual process.

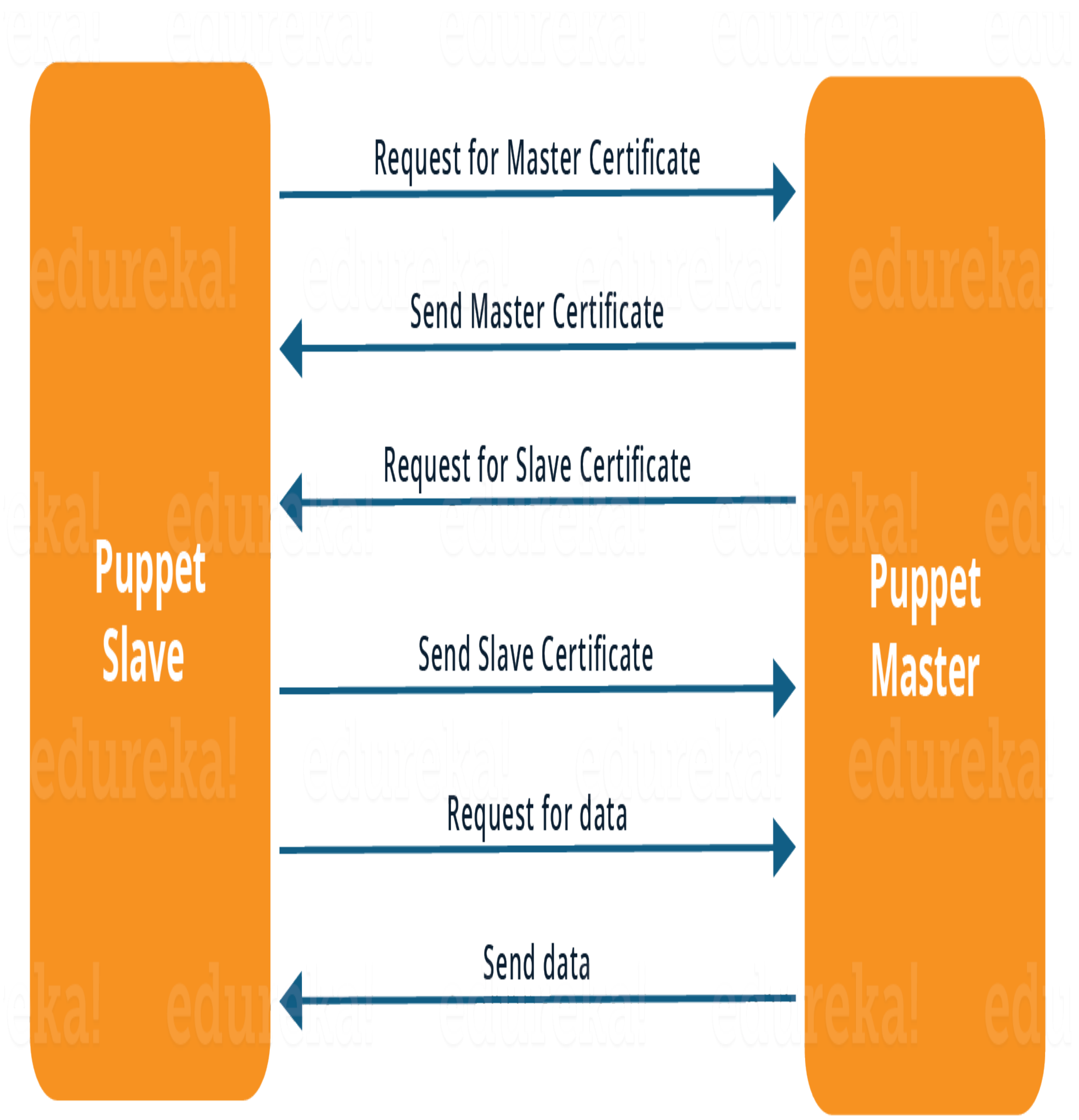
Companies for faster deployments treat infrastructure like software: as code that can be managed with the DevOps tools and processes. These tools let you make infrastructure changes more easily, rapidly, safely and reliably.

Q5. Which among Puppet, Chef, SaltStack and Ansible is the best Configuration Management (CM) tool? Why?

This depends on the organization's need so mention few points on all those tools:
Puppet is the oldest and most mature CM tool. Puppet is a Ruby-based Configuration Management tool, but while it has some free features, much of what makes Puppet great is only available in the paid version. Organizations that don't need a lot of extras will find Puppet useful, but those needing more customization will probably need to upgrade to the paid version.
Chef is written in Ruby, so it can be customized by those who know the language. It also includes free features, plus it can be upgraded from open source to enterprise-level if necessary. On top of that, it's a very flexible product.
Ansible is a very secure option since it uses Secure Shell. It's a simple tool to use, but it does offer a number of other services in addition to configuration management. It's very easy to learn, so it's perfect for those who don't have a dedicated IT staff but still need a configuration management tool.
SaltStack is python based open source CM tool made for larger businesses, but its learning curve is fairly low.

Q6. What is Puppet?

I will advise you to first give a small definition of Puppet. It is a Configuration Management tool which is used to automate administration tasks.
Now you should describe its architecture and how Puppet manages its Agents. Puppet has a Master-Slave architecture in which the Slave has to first send a Certificate signing request to Master and Master has to sign that Certificate in order to establish a secure connection between Puppet Master and Puppet Slave as shown on the diagram below. Puppet Slave sends request to Puppet Master and Puppet Master then pushes configuration on Slave.
Refer the diagram below that explains the above description.



Q7. Before a client can authenticate with the Puppet Master, its certs need to be signed and accepted. How will you automate this task?

The easiest way is to enable auto-signing in puppet.conf. Do mention that this is a security risk. If you still want to do this:

- Firewall your puppet master – restrict port tcp/8140 to only networks that you trust.

- Create puppet masters for each ‘trust zone’, and only include the trusted nodes in that Puppet masters manifest.
- Never use a full wildcard such as *.

Q8. Describe the most significant gain you made from automating a process through Puppet.

For this answer, I will suggest you to explain you past experience with Puppet. you can refer the below example:
I automated the configuration and deployment of Linux and Windows machines using Puppet. In addition to shortening the processing time from one week to 10 minutes, I used the roles and profiles pattern and documented the purpose of each module in README to ensure that others could update the module using Git. The modules I wrote are still being used, but they’ve been improved by my teammates and members of the community

Q9. Which open source or community tools do you use to make Puppet more powerful?

Over here, you need to mention the tools and how you have used those tools to make Puppet more powerful. Below is one example for your reference:
Changes and requests are ticketed through Jira and we manage requests through an internal process. Then, we use Git and Puppet’s Code Manager app to manage Puppet code in accordance with best practices. Additionally, we run all of our Puppet changes through our continuous integration pipeline in Jenkins using the beaker testing framework.

Q10. What are Puppet Manifests?

It is a very important question so make sure you go in a correct flow. According to me, you should first define Manifests. Every node (or Puppet Agent) has got its configuration details in Puppet Master, written in the native Puppet language. These details are written in the language which Puppet can understand and are termed as Manifests. They are composed of Puppet code and their filenames use the .pp extension.
Now give an exampl. You can write a manifest in Puppet Master that creates a file and installs apache on all Puppet Agents (Slaves) connected to the Puppet Master.

Q11. What is Puppet Module and How it is different from Puppet Manifest?

For this answer, you can go with the below mentioned explanation:
A Puppet Module is a collection of Manifests and data (such as facts, files, and templates), and they have a specific directory structure. Modules are useful for organizing your Puppet code, because they allow you to split your code into multiple Manifests. It is considered best practice to use Modules to organize almost all of your Puppet Manifests.
Puppet programs are called Manifests which are composed of Puppet code and their file names use the .pp extension.

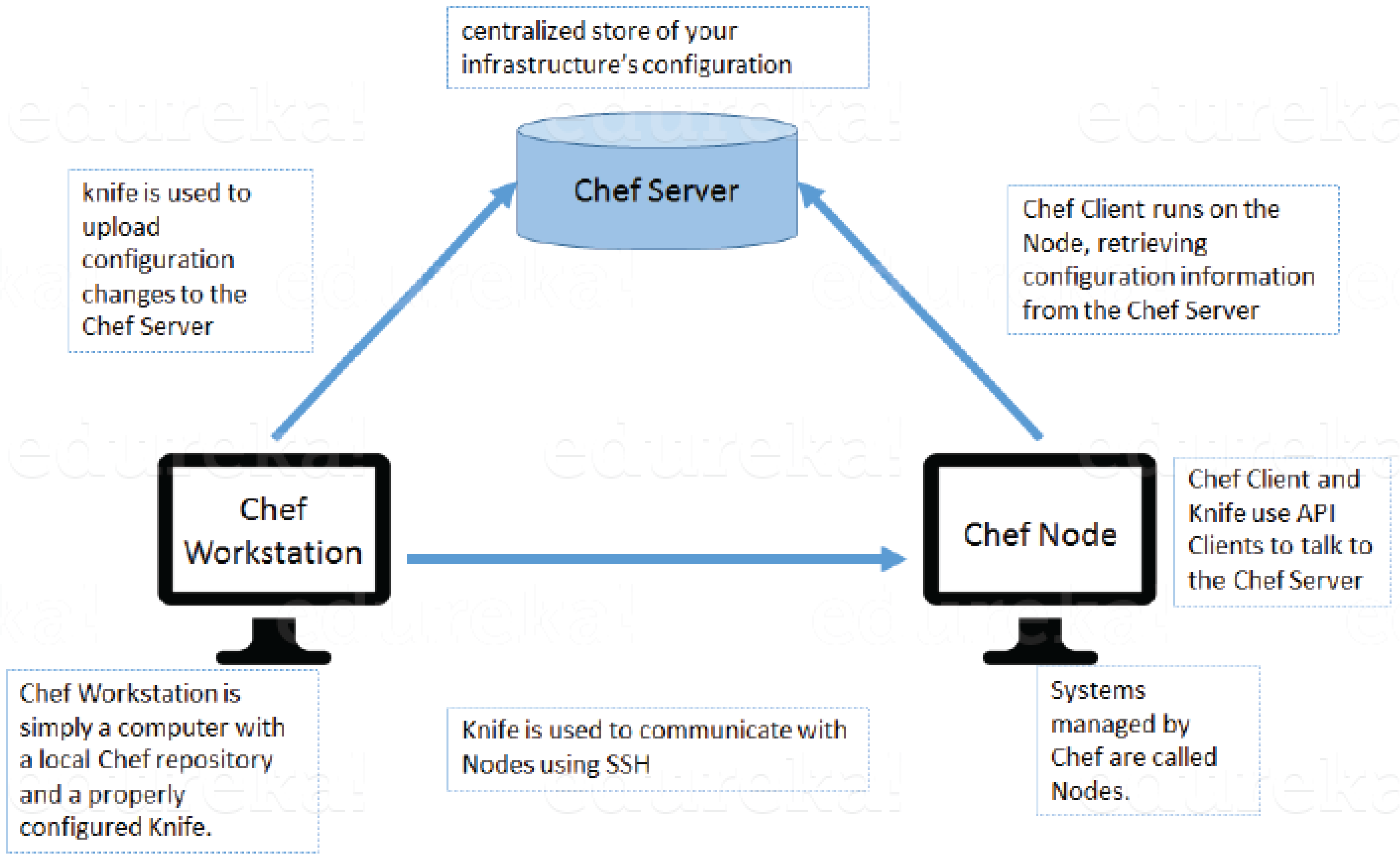
Q12. What is Facter in Puppet?

You are expected to answer what exactly Facter does in Puppet so according to me, you should say, “Facter gathers basic information (facts) about Puppet Agent such as hardware details, network settings, OS type and version, IP addresses, MAC addresses, SSH keys, and more. These facts are then made available in Puppet Master's Manifests as variables."

Q13. What is Chef?

Begin this answer by defining Chef. It is a powerful automation platform that transforms infrastructure into code. Chef is a tool for which you write scripts that are used to automate processes. What processes? Pretty much anything related to IT.
Now you can explain the architecture of Chef, it consists of:

- **Chef Server:** The Chef Server is the central store of your infrastructure's configuration data. The Chef Server stores the data necessary to configure your nodes and provides search, a powerful tool that allows you to dynamically drive node configuration based on data.
- **Chef Node:** A Node is any host that is configured using Chef-client. Chef-client runs on your nodes, contacting the Chef Server for the information necessary to configure the node. Since a Node is a machine that runs the Chef-client software, nodes are sometimes referred to as "clients".
- **Chef Workstation:** A Chef Workstation is the host you use to modify your cookbooks and other configuration data.



centralized store of your infrastructure's configuration
Chef Server
knife is used to upload configuration changes to the Chef Server
Chef Client runs on the Node, retrieving configuration information from the Chef Server
Chef Workstation
Chef Node
Chef Client and Knife use API Clients to talk to the Chef Server
Chef Workstation is simply a computer with a local Chef repository and a properly configured Knife.
Knife is used to communicate with Nodes using SSH
Systems managed by Chef are called Nodes.

Q14. What is a resource in Chef?

My suggestion is to first define Resource. A Resource represents a piece of infrastructure and its desired state, such as a package that should be installed, a service that should be running,

or a file that should be generated.
You should explain about the functions of Resource for that include the following points:

- Describes the desired state for a configuration item.
- Declares the steps needed to bring that item to the desired state.
- Specifies a resource type such as package, template, or service.
- Lists additional details (also known as resource properties), as necessary.
- Are grouped into recipes, which describe working configurations.

Q15. What do you mean by recipe in Chef?

For this answer, I will suggest you to use the above mentioned flow: first define Recipe. A Recipe is a collection of Resources that describes a particular configuration or policy. A Recipe describes everything that is required to configure part of a system.
After the definition, explain the functions of Recipes by including the following points:

- Install and configure software components.
- Manage files.
- Deploy applications.
- Execute other recipes.

Q16. How does a Cookbook differ from a Recipe in Chef?

The answer to this is pretty direct. You can simply say, “a Recipe is a collection of Resources, and primarily configures a software package or some piece of infrastructure. A Cookbook groups together Recipes and other information in a way that is more manageable than having just Recipes alone.”

Q17. What happens when you don’t specify a Resource’s action in Chef?

My suggestion is to first give a direct answer: when you don’t specify a resource’s action, Chef applies the default action.
Now explain this with an example, the below resource:
file ‘C:\Users\Administrator\chef-repo\settings.ini’ do
content ‘greeting=hello world’
end
is same as the below resource:
file ‘C:\Users\Administrator\chef-repo\settings.ini’ do
action :create
content ‘greeting=hello world’
end
because: create is the file Resource’s default action.

Q18. What is Ansible module?

Modules are considered to be the units of work in Ansible. Each module is mostly standalone and can be written in a standard scripting language such as Python, Perl, Ruby, bash, etc.. One of the guiding properties of modules is idempotency, which means that even if an operation is repeated multiple times e.g. upon recovery from an outage, it will always place the system into the same state.

Q19. What are playbooks in Ansible?

Playbooks are Ansible’s configuration, deployment, and orchestration language. They can describe a policy you want your remote systems to enforce, or a set of steps in a general IT process. Playbooks are designed to be human-readable and are developed in a basic text language.
At a basic level, playbooks can be used to manage configurations of and deployments to remote machines.

Q20. How do I see a list of all of the ansible_ variables?

Ansible by default gathers “facts” about the machines under management, and these facts can be accessed in Playbooks and in templates. To see a list of all of the facts that are available about a machine, you can run the “setup” module as an ad-hoc action:
Ansible -m setup hostname
This will print out a dictionary of all of the facts that are available for that particular host.

Q21. How can I set deployment order for applications?

WebLogic Server 8.1 allows you to select the load order for applications. See the Application MBean Load Order attribute in Application. WebLogic Server deploys server-level resources (first JDBC and then JMS) before deploying applications. Applications are deployed in this order: connectors, then EJBs, then Web Applications. If the application is an EAR, the individual components are loaded in the order in which they are declared in the application.xml deployment descriptor.

Q22. Can I refresh static components of a deployed application without having to redeploy the entire application?

Yes, you can use weblogic.Deployer to specify a component and target a server, using the following syntax:
java weblogic.Deployer -adminurl http://admin:7001 -name appname -targets server1,server2 -deploy jsps/*.jsp

Q23. How do I turn the auto-deployment feature off?

The auto-deployment feature checks the applications folder every three seconds to determine whether there are any new applications or any changes to existing applications and then dynamically deploys these changes.

The auto-deployment feature is enabled for servers that run in development mode. To disable auto-deployment feature, use one of the following methods to place servers in production mode:

- In the Administration Console, click the name of the domain in the left pane, then select the Production Mode checkbox in the right pane.
- At the command line, include the following argument when starting the domain’s Administration Server:
  -Dweblogic.ProductionModeEnabled=true

- Production mode is set for all WebLogic Server instances in a given domain.

Q24. When should I use the external_stage option?

Set -external_stage using weblogic.Deployer if you want to stage the application yourself, and prefer to copy it to its target by your own means.