Chef Infrastructure Automation Quiz (100 Questions)

DevOps Learning Module

This quiz covers fundamental concepts related to Chef infrastructure automation. Choose the best answer for each question.

1. What is the command-line tool used to interact with a Chef server, manage nodes, and upload cookbooks?

   A. `chef-client`

   B. `chef-solo`

   C. `knife`

   D. `ohai`

   Answer: C
   Explanation: `knife` is the primary tool for interacting with the Chef server from your workstation. It's used for managing nodes, roles, data bags, and uploading cookbooks.

2. Which file within a cookbook is used to define its metadata, such as its name, version, and dependencies on other cookbooks?

   A. `README.md`

   B. `attributes/default.rb`

   C. `recipes/default.rb`

   D. `metadata.rb`

   Answer: D
   Explanation: `metadata.rb` is a Ruby file that contains all the metadata for a cookbook, including its name, version, dependencies, and which operating systems it supports.

3. In Chef, what is the term for a system (like a server or virtual machine) that is managed by the Chef server?

   A. A Node

   B. A Client

   C. A Resource

   D. A Recipe

   Answer: A
   Explanation: A "Node" is any machine (physical, virtual, cloud) that is registered with the Chef server and configured by `chef-client`.

4. Which Chef resource is used to manage files that are transferred from a cookbook's `files/default/` directory to a node?

   A. `file`

   B. `template`

   C. `cookbook_file`

   D. `remote_file`

   Answer: C
   Explanation: The `cookbook_file` resource is used for transferring static files. It looks for the file in the cookbook's `files/default/` directory (or a platform-specific subdirectory).

5. Which resource is used to manage files that are generated from an `.erb` (Embedded Ruby) template?

   A. `file`

   B. `template`

   C. `cookbook_file`

   D. `content_file`

   Answer: B
   Explanation: The `template` resource is used for dynamic content. It processes an `.erb` (Embedded Ruby) file, allowing you to use variables and logic to generate the final file.

6. Inside an `.erb` template file, how do you correctly insert the value of a node attribute, such as `node['ipaddress']`?

   A. `${node['ipaddress']}`

   B. `<%= node['ipaddress'] %>`

   C. `<% node['ipaddress'] %>`

   D. `#{node['ipaddress']}`

   Answer: B
   Explanation: In `.erb` templates, the `<%= ... %>` tag is used to evaluate a Ruby expression and insert its string result into the file. `<% ... %>` executes code without inserting the result.

7. What is the term for the Chef component that gathers data about the current state of a node (e.g., CPU, memory, IP addresses)?

   A. `chef-client`

   B. `ohai`

   C. `knife`

   D. `node-inspector`

Answer: B
Explanation: `ohai` is run at the beginning of every `chef-client` run to collect detailed information about the node's current state (OS, kernel, network, etc.), which is then available as node attributes.

8. What is a Chef "Recipe"?

   A. A file that defines the dependencies for a cookbook.

   B. A Ruby file that primarily contains a collection of resources, defining the configuration state of a node.

   C. A JSON file defining a node's attributes.

   D. The executable that runs on the Chef server.

   Answer: B
   Explanation: A recipe is the core of Chef configuration. It's a Ruby file that declares resources (like packages, files, or services) and the desired state for them.

9. What does it mean for a Chef resource to be "idempotent"?

   A. The resource can be run on any operating system.

   B. The resource always runs, even if not needed.

   C. The resource can be run multiple times, but the result will be the same as running it once (it only makes changes if necessary).

   D. The resource must be run as the root user.

   Answer: C
   Explanation: Idempotency is a core principle of Chef. It means a configuration can be applied over and over, but it will only make a change if the system is not already in the desired state.

10. What is the purpose of a "Run List" in Chef?

    A. A list of all cookbooks available on the Chef server.

    B. A list of all nodes managed by the Chef server.

    C. A file containing a list of software to be installed.

    D. An ordered list of recipes and/or roles that defines what configuration to apply to a node.

    Answer: D
    Explanation: The run list is an ordered list that tells `chef-client` which recipes to run and in what order. It's fundamental to defining what configuration a node should have.

11. Which Chef resource is used to install, upgrade, or remove software packages on a node?

    A. `install`

    B. `service`

    C. `package`

D. `apt_update`

Answer: C
Explanation: The `package` resource is the abstract way to manage software. Chef automatically uses the correct provider (like `apt`, `yum`, or `homebrew`) based on the node's platform.

12. How do you ensure a service (like `nginx`) is running and enabled to start on boot?

    A. `service 'nginx' do action :start, :enable end`

    B. `service 'nginx' do action [:start, :enable] end`

    C. `service 'nginx' do start true, enable true end`

    D. `package 'nginx' do action [:start, :enable] end`

    Answer: B
    Explanation: To perform multiple actions on a resource, you pass them as an array of symbols to the `action` property. `:start` ensures it's running, and `:enable` ensures it starts on boot.

13. What is a "Data Bag" in Chef?

    A. A collection of node attributes.

    B. A place to store cookbooks.

    C. A global variable store, structured as JSON data, that is accessible from cookbooks.

    D. A log file container.

    Answer: C
    Explanation: Data bags are used to store global data that can be shared among nodes, such as user lists, app configurations, or settings. They are stored as JSON on the Chef server.

14. How would you load an item named `admin` from a data bag named `users`?

    A. `data_bag('users', 'admin')`

    B. `load_data_bag_item('users', 'admin')`

    C. `data_bag_item('users', 'admin')`

    D. `chef::data_bag('users', 'admin')`

    Answer: C
    Explanation: This is the helper method used within recipes to load a specific item (a JSON file) from a data bag.

15. What is the primary purpose of a "Role"?

    A. To define a set of attributes for a node.

    B. To define a run list and a set of attributes that can be applied to multiple nodes.

    C. To store secret data like passwords.

D. To test cookbooks.

Answer: B
Explanation: Roles are a way to group configurations. They contain a run list and a set of attributes that describe a function, like "webserver" or "database-server".

16. What is a Chef "Environment"?

    A. A way to manage a node's operating system.

    B. A way to organize nodes into groups, often representing stages of a workflow (e.g., development, production).

    C. The `/etc/chef` directory.

    D. A tool for running tests.

    Answer: B
    Explanation: Environments allow you to manage different groups of nodes (like "development", "staging", "production") and apply different cookbook versions or attributes to them.

17. Which `knife` command would you use to upload a cookbook named `my_app` to the Chef server?

    A. `knife upload cookbook my_app`

    B. `knife cookbook upload my_app`

    C. `knife send cookbook my_app`

    D. `knife cookbook push my_app`

    Answer: B
    Explanation: This is the standard command to upload a cookbook (and all its files) from your local workstation to the Chef server.

18. What is the purpose of the `notifies` keyword in a resource?

    A. It sends an email notification.

    B. It prevents the resource from running.

    C. It signals another resource to take an action, but only if the current resource changes state.

    D. It logs a message to the console.

    Answer: C
    Explanation: This creates a subscription. When the resource with `notifies` changes, it sends a notification to another resource, triggering an action (like a service restart).

19. A `template` resource for `/etc/nginx.conf` notifies a `service[nginx]` resource to `:restart`. When does this restart happen?

    A. Immediately after the template resource is processed.

    B. At the very beginning of the Chef run.

C. At the end of the Chef run, after all other resources have been processed.

D. It doesn't, you must also use `subscribes`.

Answer: C
Explanation: Notifications are collected during the converge phase and are all run at the very end. This prevents a service from restarting multiple times if several config files change.

20. What is the opposite of `notifies`?

A. `listens_to`

B. `subscribes`

C. `waits_for`

D. `ignores`

Answer: B
Explanation: `subscribes` is the "pull" version of `notifies`. A resource can `subscribe` to another resource, and it will take action if the resource it's subscribed to changes.

21. What is the purpose of a "Guard" like `only_if` or `not_if`?

A. To secure the Chef server.

B. To define a resource's idempotency.

C. To control whether a resource should run, based on the evaluation of a string or Ruby block.

D. To define user permissions.

Answer: C
Explanation: Guards (`only_if` and `not_if`) are conditional checks (using shell commands or Ruby blocks) that run just before the resource's action. They determine if the resource should run at all.

22. Which resource is used to run an arbitrary command on the node?

A. `script`

B. `bash`

C. `command`

D. `execute`

Answer: D
Explanation: The `execute` resource is a general-purpose tool for running any shell command. It's powerful but should be used sparingly.

23. Why is it generally discouraged to use the `execute` resource?

A. It is not secure.

B. It is often not idempotent by default, and a more specific resource (like `package`) should be used.

C. It only works on Linux.

D. It cannot be notified by other resources.

Answer: B
Explanation: `execute` resources will run every single time unless you add a guard (`not_if` or `only_if`) to make them idempotent. It's always better to use a specific resource if one exists.

24. What is "Test Kitchen"?

A. A Chef server component.

B. A testing framework that provisions instances, converges them with Chef, and runs verification tests.

C. A cookbook for setting up a kitchen.

D. A replacement for `knife`.

Answer: B
Explanation: Test Kitchen is the primary integration testing tool for Chef. It spins up a test instance (like a VM or container), applies your cookbook, and runs tests to verify the result.

25. In a `.kitchen.yml` file, what does the "provisioner" define?

A. The type of instance to create (e.g., Docker, Vagrant, EC2).

B. The tool to use for configuration (e.g., `chef_solo`, `chef_zero`, `chef_client`).

C. The testing framework to use (e.g., InSpec, Serverspec).

D. The operating system to use.

Answer: B
Explanation: The "provisioner" in Test Kitchen defines how to apply the configuration. `chef_zero` and `chef_solo` are common for local testing, while `chef_client` can be used for server testing.

26. What is "InSpec"?

A. A Chef resource for inspecting files.

B. A compliance and auditing framework that uses tests to check the state of your infrastructure.

C. The tool that gathers Ohai data.

D. A Chef server API endpoint.

Answer: B
Explanation: InSpec is Chef's testing and compliance framework. It provides a simple, human-readable language (RSpec-like) to define the desired state of your node for verification.

27. What is the "Chef Supermarket"?

A. A tool for purchasing Chef licenses.

B. A community-driven repository of publicly available cookbooks.

C. A monitoring dashboard for Chef.

D. A tool for managing data bags.

Answer: B
Explanation: The Supermarket is the public, community-hosted repository for Chef cookbooks. You can find and download cookbooks for thousands of common software and tasks.

28. What is "Berkshelf"?

    A. A tool for managing node run lists.

    B. A cookbook dependency manager that resolves and fetches cookbook dependencies.

    C. A shelf in the Chef server.

    D. An alternative to Test Kitchen.

    Answer: B
    Explanation: Berkshelf is a dependency manager. It reads your cookbook's `Berksfile` (or `metadata.rb`), finds all required dependent cookbooks, and downloads them.

29. Which file does Berkshelf use to manage a cookbook's dependencies?

    A. `metadata.rb`

    B. `Berksfile`

    C. `Cheffile`

    D. `dependencies.yml`

    Answer: B
    Explanation: While Berkshelf *can* use `metadata.rb`, the `Berksfile` is its own dedicated file for defining cookbook sources (like the Supermarket or a Git repo) and dependencies.

30. What is the correct order of attribute precedence in Chef (from lowest to highest)?

    A. Role, Environment, Node (Normal), Cookbook (Default)

    B. Cookbook (Default), Environment, Role, Node (Normal)

    C. Node (Normal), Role, Environment, Cookbook (Default)

    D. Cookbook (Default), Role, Environment, Node (Normal)

    Answer: B
    Explanation: This is a simplified version. The full order is complex, but generally, attributes set in cookbooks (`default`) are lowest, followed by environments, then roles, then attributes set directly on the node (`normal`).

31. Which attribute type has the HIGHEST precedence?

A. `default`

B. `normal`

C. `override`

D. `automatic`

Answer: D
Explanation: `automatic` attributes are those collected by `ohai`. They have the highest precedence because they represent the *actual* state of the node and cannot be overridden.

32. What are the two main phases of a `chef-client` run?

A. Upload and Download

B. Compile and Converge

C. Test and Deploy

D. Search and Apply

Answer: B
Explanation: The "Compile Phase" builds the resource collection. The "Converge Phase" executes the actions for each resource to bring the node to the desired state.

33. During which phase of a `chef-client` run are all the resources collected and put into a "resource collection"?

A. Converge Phase

B. Ohai Phase

C. Compile Phase

D. Notification Phase

Answer: C
Explanation: During the compile phase, `chef-client` loads all recipes in the run list, evaluates the Ruby code, and builds the list of all resources to be managed.

34. During which phase of a `chef-client` run are the actions for each resource actually executed?

A. Converge Phase

B. Compile Phase

C. Sync Phase

D. Attribute Phase

Answer: A
Explanation: After compiling, `chef-client` enters the converge phase, where it iterates through the resource collection and executes the specified action for each resource (if needed).

35. What is an "Encrypted Data Bag"?

    A. A data bag that can only be read by the Chef server.

    B. A data bag item that has been encrypted using a shared secret (a key).

    C. A special type of cookbook for storing secrets.

    D. A feature that encrypts all communication with the Chef server.

    Answer: B
    Explanation: This is a standard data bag item where the values (but not the keys) are encrypted using a symmetric key (a shared secret).

36. Which `knife` command is used to create an encrypted data bag item?

    A. `knife encrypt data bag item ...`

    B. `knife data bag create ... -encrypt`

    C. `knife data bag from file ... -secret-file ...`

    D. `knife vault create ...`

    Answer: C
    Explanation: This command encrypts the data bag item JSON file using the key specified in the `-secret-file` before uploading it.

37. What is "Chef-Vault"?

    A. The component of the Chef server that stores cookbooks.

    B. A password manager for your workstation.

    C. A tool that extends data bags to manage secrets in a more secure, per-node-permission way.

    D. The old name for Chef Automate.

    Answer: C
    Explanation: Chef-Vault is a more advanced secret management tool. It uses public-key cryptography to encrypt a data bag item *for* specific nodes, so only those nodes can decrypt it.

38. What is the `lazy` attribute property used for?

    A. To delay the execution of a resource until the end of the Chef run.

    B. To delay the evaluation of a resource's attribute until the execution (converge) phase.

    C. To mark a resource as optional.

    D. To make a Chef run go faster.

    Answer: B
    Explanation: The `lazy` property defers the evaluation of its code block until the converge phase. This is useful when you need a value that is set by *another* resource that hasn't run yet.

39. What is the purpose of the `chef-apply` command?

    A. To apply a local Chef recipe file without needing a full Chef server or client setup.

    B. To apply for a job at Chef.

    C. To apply a hotfix to the Chef server.

    D. To apply a new role to a node.

    Answer: A
    Explanation: `chef-apply` is a simple way to run a single recipe file. It's useful for quick tests or simple tasks without the overhead of a full client-server setup.

40. What is a "Custom Resource" (formerly LWRP)?

    A. A resource that is only available in Chef Automate.

    B. A way to define your own reusable resource by combining other resources and Ruby code.

    C. A resource that has been downloaded from the Supermarket.

    D. A resource that contains sensitive data.

    Answer: B
    Explanation: This is a powerful feature that lets you create your own reusable resources with custom properties and actions, making your recipes cleaner and more abstract.

41. Where are custom resources defined within a cookbook?

    A. `recipes/`

    B. `libraries/`

    C. `resources/`

    D. `providers/`

    Answer: C
    Explanation: Custom resources are defined in `.rb` files within the `resources/` directory of a cookbook. The filename typically matches the resource name.

42. In a custom resource, what does the `action` block define?

    A. The properties the resource can accept.

    B. The code that gets executed when the action is called.

    C. The default action to be taken.

    D. The nodes that this action can run on.

    Answer: B
    Explanation: The `action :action_name do ... end` block contains the Ruby code and core Chef resources that are executed when your custom resource is called with that action.

43. What is the `chef-solo` provisioner in Test Kitchen used for?

A. To test against a real Chef server.

B. To test cookbooks without a Chef server, by syncing the cookbook directory to the instance.

C. To test multi-node setups.

D. To run InSpec tests.

Answer: B
Explanation: `chef_solo` is a provisioner that simulates a server-less Chef run. Test Kitchen copies the cookbooks to the instance, and `chef-solo` runs them from the local disk.

44. What is the purpose of the `include_recipe` method in a recipe?

A. To download a recipe from the Supermarket.

B. To ensure that another recipe is loaded and executed during the compile phase.

C. To copy the contents of one recipe into another.

D. To schedule a recipe to run at the end.

Answer: B
Explanation: This method ensures that the specified recipe is parsed and its resources are added to the resource collection. It's idempotent; a recipe will only be included once.

45. What is the `remote_file` resource used for?

A. To download a file from a remote URL (like HTTP or FTP).

B. To copy a file from the cookbook's `files/` directory.

C. To manage files on a remote Chef server.

D. To create a template file.

Answer: A
Explanation: This resource is used to download files from remote URLs. It is idempotent and will only download the file if it's missing or has changed (if using checksums).

46. How can you create a file with specific content directly in a recipe?

A. `file '/path/to/file' do content 'My file content' end`

B. `file '/path/to/file' do write 'My file content' end`

C. `file '/path/to/file' do text 'My file content' end`

D. `cookbook_file '/path/to/file' do content 'My file content' end`

Answer: A
Explanation: The `file` resource is the most basic file management tool. Using the `content` property allows you to specify the exact content of the file directly in the recipe.

47. What is the `log` resource used for?

A. To configure log rotation.

B. To write a message to the Chef client's log output.

C. To manage system log files.

D. To log into a remote server.

Answer: B
Explanation: The `log` resource is used for debugging. It writes a message to the `chef-client` output log (e.g., STDOUT or `client.log`) at a specified log level.

48. What is the default action for the `package` resource?

A. `:upgrade`

B. `:install`

C. `:remove`

D. `:nothing`

Answer: B
Explanation: If you just write `package 'httpd'`, Chef assumes the action is `:install`.

49. What is the default action for the `template` resource?

A. `:create`

B. `:touch`

C. `:render`

D. `:nothing`

Answer: A
Explanation: If you just write `template '/etc/foo.conf'`, Chef assumes the action is `:create`, which means it will create the file from the template.

50. What is the purpose of the `user` resource?

A. To manage Chef server users.

B. To manage local user accounts on a node.

C. To set the user that `chef-client` runs as.

D. To manage data bag users.

Answer: B
Explanation: The `user` resource manages local users on the node, allowing you to create, delete, lock, or modify user accounts, home directories, and passwords.

51. What is "Policyfiles"?

A. A feature for managing firewall rules.

B. A newer, alternative way to manage cookbook dependencies, attributes, and run lists for a node in a single document.

C. A security feature in Chef Automate.

D. A file for defining InSpec profiles.

Answer: B
Explanation: Policyfiles are a more modern and robust way to manage a node's configuration. They lock cookbook versions and attributes into a single file, making deployments more reliable and testable.

52. Which file is used to define a Policy?

    A. `Policy.rb`

    B. `metadata.rb`

    C. `Policyfile.rb`

    D. `Berksfile`

    Answer: C
    Explanation: This is the Ruby file where you define the `name`, `run_list`, and cookbook sources for your policy.

53. What file is generated when you run `chef install` on a `Policyfile.rb`?

    A. `Policyfile.lock.json`

    B. `Berksfile.lock`

    C. `policy.json`

    D. `node.json`

    Answer: A
    Explanation: Similar to a `Gemfile.lock`, this lock file is generated by `chef install` and contains the exact set and versions of all cookbooks that satisfy the policy.

54. What is the primary benefit of using Policyfiles over Roles and Environments?

    A. It's simpler for beginners.

    B. It's faster.

    C. It provides a single, testable, and immutable artifact that defines a node's entire configuration.

    D. It doesn't require a Chef server.

    Answer: C
    Explanation: The `Policyfile.lock.json` (and the corresponding policy archive) is an immutable artifact. This means the *exact* same configuration can be promoted from dev to test to prod.

55. What is the `chef-client` "interval"?

    A. The timeout for a Chef run.

    B. The frequency (in seconds) at which the `chef-client` daemon runs automatically.

    C. The time between the compile and converge phases.

D. The network latency to the Chef server.

Answer: B
Explanation: When `chef-client` is run as a daemon (service), the `interval` setting in `client.rb` defines how often (in seconds) it wakes up and performs a run.

56. Which Chef Server component is the main API endpoint?

A. Bookshelf

B. Solr

C. PostgreSQL

D. Erchef

Answer: D
Explanation: Erchef is the main API server, written in Erlang. It handles requests from `knife` and `chef-client`, coordinating with the other backend components.

57. Which Chef Server component stores cookbooks?

A. Bookshelf

B. Erchef

C. RabbitMQ

D. PostgreSQL

Answer: A
Explanation: Bookshelf is the S3-backed (or filesystem-backed) storage service that holds the actual cookbook files (templates, files, etc.).

58. Which Chef Server component provides the search functionality?

A. Erchef

B. Solr

C. PostgreSQL

D. Nginx

Answer: B
Explanation: Chef uses a search engine (like Solr or Elasticsearch) to index all node data, attributes, and roles, enabling powerful search queries.

59. What is the `directory` resource used for?

A. To list the contents of a directory.

B. To manage a directory on a node (create, delete, set permissions).

C. To define a cookbook's directory structure.

D. To search for a directory.

Answer: B
Explanation: This resource is used to create, delete, and manage permissions for directories on the node.

60. How do you set file permissions (mode) to `0755` on a `directory` resource?

    A. `directory '/opt/app' do permissions '0755' end`

    B. `directory '/opt/app' do mode '0755' end`

    C. `directory '/opt/app' do chmod '0755' end`

    D. `directory '/opt/app' do access '0755' end`

    Answer: B
    Explanation: The `mode` property is used by `file`, `directory`, `template`, etc., to set the file system permissions (in octal or string format).

61. What does the `recursive true` property on a `directory` resource do?

    A. Deletes the directory and all its contents.

    B. Creates the parent directories if they do not exist.

    C. Applies the permissions to all subdirectories.

    D. Both B and C.

    Answer: B
    Explanation: Like `mkdir -p`, setting `recursive true` will automatically create any missing parent directories in the path.

62. What is the `cron` resource used for?

    A. To manage cron jobs on a node.

    B. To schedule Chef client runs.

    C. To time how long a Chef run takes.

    D. To run a command in the future.

    Answer: A
    Explanation: This resource manages entries in the system's cron table, allowing for scheduled tasks. It's idempotent and won't create duplicate entries.

63. What is the `group` resource used for?

    A. To group nodes in an environment.

    B. To manage local groups on a node.

    C. To group resources in a recipe.

    D. To define a Chef server user group.

    Answer: B
    Explanation: This resource manages local groups on the node (e.g., in `/etc/group`), allowing you to create, delete, or modify groups and their members.

64. What is the `link` resource used for?

    A. To create a hyperlink in a log file.

    B. To link two nodes together.

    C. To create a symbolic or hard link on a node.

    D. To link a node to an environment.

    Answer: C
    Explanation: This resource manages file system links. By default, it creates a symbolic link (`ln -s`).

65. Which property specifies the target of a symbolic link in a `link` resource?

    A. `target`

    B. `source`

    C. `to`

    D. `link_to`

    Answer: C
    Explanation: The `to` property specifies the path that the link should point to. The main resource name is the path of the link itself (e.g., `link '/usr/bin/foo' do to '/opt/foo/bin/foo' end`).

66. What is the purpose of the `variables` property in a `template` resource?

    A. To list the environment variables to set.

    B. To pass a hash of variables from the recipe to the `.erb` template file.

    C. To define the attribute variables for the node.

    D. To list the Ohai variables to use.

    Answer: B
    Explanation: This property takes a Hash. Each key-value pair in the hash is made available as an instance variable (e.g., `@my_var`) inside the `.erb` template.

67. What is `chef-zero`?

    A. A "zero-dependency" version of `chef-client`.

    B. A lightweight, in-memory Chef server that runs locally, often used by Test Kitchen.

    C. A Chef server with no nodes.

    D. A tool to reset a Chef server.

    Answer: B
    Explanation: `chef-zero` (now the default provisioner in Test Kitchen) acts as a local, in-memory Chef server. `kitchen converge` uploads the cookbooks to `chef-zero` and then `chef-client` runs against it.

68. In a recipe, what does `node.default['my_app']['version'] = '1.0'` do?

A. Sets a `normal` attribute.

B. Sets a `default` attribute, which has low precedence.

C. Sets an `override` attribute.

D. This syntax is invalid.

Answer: B
Explanation: `node.default` is the standard way to set attributes in an attribute file. These have the lowest precedence and are easily overridden by roles, environments, or normal attributes.

69. How would you set an attribute with the highest non-automatic precedence?

A. `node.force_override['my_app']['version'] = '1.0'`

B. `node.override['my_app']['version'] = '1.0'`

C. `node.normal['my_app']['version'] = '1.0'`

D. `node.default['my_app']['version'] = '1.0'`

Answer: B
Explanation: `override` attributes have a higher precedence than `default`, `role`, or `environment` attributes. They are typically set in recipes to ensure a value is used.

70. Where are a cookbook's default attributes typically defined?

A. `recipes/default.rb`

B. `metadata.rb`

C. `attributes/default.rb`

D. `files/default/attributes.rb`

Answer: C
Explanation: By convention, default cookbook attributes are defined in `attributes/default.rb`. You can have other attribute files (e.g., `attributes/web.rb`) but they must be loaded manually.

71. What does the `kitchen converge` command do?

A. Destroys the test instance.

B. Runs the InSpec tests.

C. Creates the instance and runs the provisioner (e.g., `chef-client`) on it.

D. Creates the instance only.

Answer: C
Explanation: This command runs the "converge" step of the Test Kitchen workflow, which typically involves creating the instance (if it doesn't exist) and then running `chef-client` to apply the run list.

72. What does the `kitchen verify` command do?

A. Verifies the `.kitchen.yml` syntax.

B. Runs the verifier (e.g., InSpec tests) on the instance.

C. Runs the Chef convergence.

D. Verifies connectivity to the instance.

Answer: B
Explanation: This command runs the "verify" step, which executes the test framework (defined by the "verifier", usually InSpec) to check if the convergence was successful.

73. What does the `kitchen test` command do?

A. Runs all steps in order: destroy, create, converge, verify, and destroy.

B. Runs only the `verify` step.

C. Runs only the `converge` and `verify` steps.

D. Creates and configures the instance, then leaves it running.

Answer: A
Explanation: This is the all-in-one command. It runs the full, idempotent test cycle: `destroy`, `create`, `converge`, `verify`, and finally `destroy` (to clean up).

74. In an InSpec test, how do you check if the `nginx` package is installed?

A. `describe package('nginx') do it  should be_installed  end`

B. `check package('nginx') do it  is_installed  end`

C. `describe 'nginx' do it  should be_package  end`

D. `assert_installed('nginx')`

Answer: A
Explanation: InSpec uses a "resource-based" language. You `describe` a `package` and then state its expected properties, such as `should be_installed`.

75. In an InSpec test, how do you check if the `nginx` service is running and enabled?

A. `describe service('nginx') do it  should be_running and be_enabled  end`

B. `describe service('nginx') do it  should be_running  it  should be_enabled end`

C. `check service('nginx') do it  is_running and is_enabled  end`

D. `describe 'nginx' do it  should be_service.running.enabled  end`

Answer: B
Explanation: Similarly, you `describe` the `service` and can chain multiple expectations. The format `it  should ...   it  should ...   end` is the standard for multiple checks.

76. In an InSpec test, how do you check if the file `/etc/nginx.conf` exists and is owned by `root`?

A. `describe file('/etc/nginx.conf') do it  should exist  it  should be_owned_by 'root'  end`

B. `describe '/etc/nginx.conf' do it  exists  it  is_root  end`

C. `check file('/etc/nginx.conf') do it  should exist and be_owned_by 'root' end`

D. `assert file('/etc/nginx.conf').owner == 'root'`

Answer: A
Explanation: The `file` resource in InSpec is very powerful. You can check for existence, permissions, owner, group, content, and more.

77. What is the purpose of the `bash` resource?

A. It is an alias for the `execute` resource, setting the interpreter to `/bin/bash`.

B. It installs the Bash shell.

C. It validates Bash script syntax.

D. It runs a command in the Bash shell on your workstation.

Answer: A
Explanation: The `bash`, `powershell`, `python`, etc. resources are all helpers that inherit from the `execute` resource but conveniently set the `interpreter` property for you.

78. What is a "partial search"?

A. A search that only returns half the results.

B. A Chef search optimization that returns only the specific attributes you request, rather than the entire node object.

C. A search that fails.

D. A search for nodes that are only partially converged.

Answer: B
Explanation: Full node objects can be very large. Partial search allows you to request *only* the data you need (e.g., `['ipaddress', 'hostname']`), reducing network traffic and memory usage.

79. How do you find all nodes with the role `webserver` using `knife`?

A. `knife nodes find role:webserver`

B. `knife search node 'role:webserver'`

C. `knife list nodes -role webserver`

D. `knife role:webserver list`

Answer: B
Explanation: This is the standard Solr query syntax used by `knife search`. You specify the index (`node`) and the query.

80. What is the purpose of the `source` property on a `template` resource?

A. It specifies which node to source the data from.

B. It specifies the name of the `.erb` template file in the cookbook's `templates/` directory.

C. It specifies a URL to download the template from.

D. It specifies the source code for the template.

Answer: B
Explanation: The `source` property defines the *name* of the `.erb` file to use. By default, Chef looks in the `templates/default/` directory of the *current* cookbook.

81. If you don't specify a `source` property for `template '/etc/foo.conf'`, what file will Chef look for?

A. `templates/default/foo.conf.erb`

B. `templates/foo.conf.erb`

C. `files/default/foo.conf.erb`

D. `templates/default/etc/foo.conf.erb`

Answer: A
Explanation: If `source` is not specified, Chef derives the template name from the resource's `name` property. It assumes an `.erb` extension and looks in the `templates/default/` folder.

82. What is the `reboot` resource used for?

A. To request a reboot of the node at the end of the Chef run.

B. To immediately reboot the node.

C. To schedule a reboot in cron.

D. To reboot the Chef server.

Answer: A
Explanation: This resource adds a pending reboot request. `chef-client` will finish its run and then, if a reboot is pending, it will reboot the node.

83. How can you make a resource run at compile time?

A. `my_resource 'foo' do action :run, :compile_time end`

B. `my_resource 'foo' do action :nothing end.run_action(:run)`

C. `my_resource 'foo' do phase :compile end`

D. `compile_time  my_resource 'foo'`

Answer: B
Explanation: Attaching `.run_action(:action_name)` to the end of a resource block *breaks* the normal flow. It forces the resource to run immediately (at compile time) instead of being added to the resource collection.

84. What is a common use case for running a resource at compile time?

A. Installing a package (like a gem) that is needed by another resource's custom code (like a library).

B. Restarting a service.

C. Creating a user.

D. Writing a log file.

Answer: A
Explanation: This is the classic example. If a recipe needs a Ruby gem to parse JSON *during* the compile phase, you must install that gem at compile time using `chef_gem 'my_gem' do compile_time true end` or the `run_action` method.

85. What is a "why-run" mode in `chef-client`?

A. A mode that asks "why" each resource is being run.

B. A dry-run mode that shows what changes *would* be made, without actually making them.

C. A debugging mode that steps through each resource.

D. A mode that only runs resources with a `why_run true` property.

Answer: B
Explanation: Why-run mode is a "dry run" feature. `chef-client` runs, but instead of executing actions, it just reports what actions it *would* have executed.

86. What is the command to run `chef-client` in "why-run" mode?

A. `chef-client -dry-run`

B. `chef-client -W`

C. `chef-client -why-run`

D. Both B and C.

Answer: D
Explanation: `-W` is the short flag for `-why-run`. Both commands enable why-run mode.

87. What is the `apt_update` resource used for?

A. To update all packages on an Apt-based system.

B. To run `apt-get update` to refresh the package cache.

C. To install the `apt` package.

D. To update the `chef-client` itself.

Answer: B
Explanation: On Debian/Ubuntu systems, the package cache can get stale. The `apt_update` resource is used to run `apt-get update`, and other `package` resources can notify it to run.

88. What is the `kitchen.yml` "driver" responsible for?

A. Driving the tests.

B. Installing the `chef-client`.

C. Creating, destroying, and managing the test instances (e.g., Vagrant, Docker, EC2).

D. Connecting to the instance.

Answer: C
Explanation: The "driver" is the "platform" for your test instance. `kitchen-vagrant` (Vagrant/VirtualBox), `kitchen-docker` (Docker), and `kitchen-ec2` (AWS) are common drivers.

89. What is the `kitchen.yml` "verifier" responsible for?

A. Verifying the Chef server connection.

B. The framework used to run tests (e.g., InSpec).

C. Verifying the instance can be created.

D. Verifying the provisioner is installed.

Answer: B
Explanation: The "verifier" defines the testing framework. `kitchen-inspec` is the modern default, replacing the older `kitchen-serverspec`.

90. What is the `kitchen.yml` "transport" responsible for?

A. Transporting cookbooks to the Chef server.

B. The mechanism for connecting to the instance (e.g., SSH, WinRM).

C. The network configuration of the instance.

D. The protocol for running tests.

Answer: B
Explanation: The "transport" is the protocol Test Kitchen uses to communicate with the instance, run commands, and copy files. This is typically `ssh` for Linux and `winrm` for Windows.

91. How do you add a recipe to a node's run list from the command line?

A. `knife node run_list add my_node 'recipe[my_app::server]'`

B. `knife run_list add my_node 'recipe[my_app::server]'`

C. `knife node add_recipe my_node 'my_app::server'`

D. `knife node my_node run_list set 'recipe[my_app::server]'`

Answer: A
Explanation: This command modifies the node object on the Chef server, adding the specified recipe (or role) to its run list.

92. What is the difference between `node.default` and `node.override`?

A. They are the same.

B. `node.default` has higher precedence.

C. `node.override` has higher precedence.

D. `node.default` is for recipes, `node.override` is for roles.

Answer: C
Explanation: Attributes are merged in a specific order. `override` attributes (from recipes, roles, or environments) will always "win" over `default` attributes.

93. What is the `client.rb` file used for?

A. To configure the `knife` tool.

B. To configure the `chef-client` agent on a node.

C. To define a custom resource.

D. To store the node's private key.

Answer: B
Explanation: This is the configuration file for the `chef-client` agent itself. It's typically located at `/etc/chef/client.rb` on a node.

94. Which setting in `client.rb` points the `chef-client` to the Chef server?

A. `chef_server_url`

B. `server_url`

C. `node_name`

D. `chef_server`

Answer: A
Explanation: This is the most important setting. It tells the `chef-client` which Chef server API to contact for its run list and cookbooks.

95. What is the `knife.rb` file used for?

A. To configure the `knife` command-line tool.

B. To configure the `chef-client` agent.

C. To store your Chef server password.

D. To define a role.

Answer: A
Explanation: This is the configuration file for your `knife` workstation tool. It's typically located at /̃`.chef/knife.rb` (or `.chef/config.rb`) and stores your credentials and server URL.

96. What is the "resource collection"?

A. All the cookbooks on the Chef server.

B. The list of all resources defined in all recipes in the run list, built during the compile phase.

C. A data bag for storing resources.

D. A list of nodes.

Answer: B
Explanation: This is the in-memory list of all resources that `chef-client` builds during the compile phase. The converge phase then just executes this list in order.

97. What does it mean if a resource's state is "green" (or "up-to-date") in the `chef-client` output?

A. The resource ran and made a change.

B. The resource was skipped due to a guard.

C. The resource was already in the desired state, and no action was needed.

D. The resource failed.

Answer: C
Explanation: Green text in the output means the resource was checked, and the node was *already* in the desired state, so no changes were made. This confirms idempotency.

98. What does it mean if a resource's state is "cyan" (or "changed") in the `chef-client` output?

A. The resource ran and made a change to the system.

B. The resource was already in the desired state.

C. The resource failed.

D. The resource is new.

Answer: A
Explanation: Cyan (or yellow/magenta in some terminals) text means the resource was *not* in the desired state, and `chef-client` successfully ran the action to correct it.

99. What is the `ruby_block` resource used for?

A. To install a Ruby gem.

B. To write a Ruby file.

C. To execute an arbitrary block of Ruby code during the converge phase.

D. To test Ruby code.

Answer: C
Explanation: This resource is a "catch-all" that allows you to run any Ruby code you want as part of the converge phase. It's useful for tasks that don't have a specific resource.

100. What is the purpose of the `libraries` directory in a cookbook?

A. To store binary library files.

B. To store Ruby code (e.g., helper methods) that can be included in recipes or custom resources.

C. To store documentation.

D. To store `.erb` templates.

Answer: B
Explanation: The `libraries/` directory is for pure Ruby code. Helper methods defined here can be `include`d in recipes or custom resources to share logic and keep recipes clean.