# TESTPASSPORT

## PRACTICE QUESTIONS



High Quality | Better Service

https://www.testpassport.com

Free updates for one year

**Exam** : **Terraform Professional 002**

**Title** : Terraform Infrastructure Engineer Professional 002

**Version** : DEMO

1.A network team retains exclusive lifecycle ownership of an existing network and its subnets. It publishes their IDs through an approved configuration input channel. An application team must create, update, and retire service compute instances, and those instances must use the published subnets.
Which division of Terraform configuration responsibilities meets this contract?
A. Declare managed resources for the network, subnets, and service instances; import the existing network and subnets into application state.
B. Receive the network and subnet IDs as input variables, and represent the service instances with data sources.
C. Receive the network and subnet IDs as input variables, and declare managed resources for the service instances.
D. Create an application-owned network and subnets with managed resources, and declare managed service instances in those new subnets.
**Answer:** C
**Explanation:**
The application configuration must manage the service instances while consuming identifiers for networking owned elsewhere. Input variables can carry the published network and subnet IDs into the managed service resources. Importing the existing networking would associate it with managed resource addresses in application state, exceeding the ownership contract even though it would avoid creating another network.
A - Incorrect: Importing existing networking adopts it into the application's managed resources. The network team's exclusive lifecycle ownership rules out that responsibility.
B - Incorrect: Input variables appropriately carry the external network identifiers, but data sources read existing service instances. They do not implement the application's responsibility to create, update, and retire those instances.
C - Correct: The inputs identify the approved external networking, while managed service resources implement the lifecycle responsibilities assigned to the application team.
D - Incorrect: Creating separate networking changes the deployment destination. The contract requires service instances to use the existing published subnets.

2.The Orders and Billing teams deploy separate services using HCP Terraform. The organization supports workspace-level team permissions, and neither team has broader access grants.
Each service module accepts network and subnet IDs and exposes both a logical service HTTPS URL and backend instance IP addresses.
The design must meet these business requirements:

| Area | Requirement |
|---|---|
| Existing network | The network team retains sole lifecycle ownership and supplies network and subnet IDs through an approved input channel. |
| Service consumers | A webhook client obtains each logical service's HTTPS URL from deployment root outputs. Backend replacement must not require the client to select another instance address. |
| Deployment authority | Each team may apply only its own service, using run credentials scoped to that service. |

Which complete configuration design meets all requirements?
A. Use a separate root and HCP Terraform workspace for each service; pass the published network IDs

to its service module, forward its HTTPS URL as a root output, and configure service-specific credentials and apply permissions limited to its owning team.
B. Use one root and HCP Terraform workspace containing both service modules; pass the published network IDs to both modules, forward both HTTPS URLs as root outputs, and use credentials for both services with both teams granted apply permission on the workspace.
C. Use a separate root and HCP Terraform workspace for each service; manage the shared network and subnets in those roots, pass their resource IDs to each service module, forward its HTTPS URL as a root output, and configure service-specific credentials and apply permissions.
D. Use a separate root and HCP Terraform workspace for each service; pass the published network IDs to its service module, publish its backend instance IP addresses as root outputs, and configure service-specific credentials and apply permissions limited to its owning team.
**Answer:** A
**Explanation:**
The complete design must satisfy ownership, consumer access to service information, and deployment authority together. Each service configuration can consume the published network IDs and expose its module's logical service URL through a root output. Separate HCP Terraform workspaces, service-specific credentials, and effective apply permissions limited to the owning team provide the required delegation. A shared workspace can contain both modules and their outputs, but granting both teams apply permission on it gives both teams authority over the shared configuration.
A - Correct: Consuming network IDs preserves external ownership. Forwarding each module's HTTPS URL supplies the consumer interface. Separate workspaces with scoped credentials and apply permissions satisfy independent deployment authority.
B - Incorrect: The network consumption and URL outputs meet their requirements, but both teams can apply the workspace containing both services. Shared run credentials also fail the requirement for credentials scoped to each service.
C - Incorrect: The service outputs and deployment delegation are appropriate, but managing the shared network in the service roots assigns network lifecycle responsibility to those configurations.
D - Incorrect: The ownership and authority boundaries are appropriate, but backend instance addresses expose replaceable members of the service. They do not provide the logical HTTPS URL required by the webhook client.

3.You are publishing a reusable Terraform service module. Each caller supplies a service name and requested replica count. The module maintainers choose the compute and load balancer implementations and may change their provider resource types in later compatible releases. Monitoring consumers need the logical service's HTTPS URL, and their configuration must remain independent of those internal resource choices.
Which input/output contract meets these requirements?
A. Accept service_name and replicas as input variables; return the current load balancer resource object, with callers deriving the HTTPS URL from its attributes.
B. Accept service_name and replicas as input variables; return backend instance IDs, with callers looking up their addresses to construct the monitoring target.
C. Require service_name, replicas, and settings for the chosen compute and load balancer resource types as input variables; return a service_url string containing the HTTPS URL.
D. Accept service_name and replicas as input variables; return a service_url string containing the

complete HTTPS URL for the logical service.

**Answer:** D

**Explanation:**

The service name and requested replica count are caller-controlled choices, so they belong in the module's input interface. Monitoring needs the logical service's complete HTTPS URL, so the module should expose that value directly. The module can derive the URL from its current resources while preserving the output's meaning. Returning the whole load balancer object could provide enough information to construct a URL today, but it makes consumers depend on provider-specific attributes that the maintainers may change.

A - Incorrect: The inputs match the caller's choices, but returning the current load balancer resource object requires consumers to understand its attributes and construct the URL. That exposes the implementation through the output contract.

B - Incorrect: Backend instance IDs identify internal service members. Looking up their addresses makes monitoring depend on replaceable compute resources instead of the logical service endpoint.

C - Incorrect: The URL output meets the monitoring need, but requiring settings for the chosen compute and load balancer resource types makes callers participate in implementation choices assigned to the module maintainers.

D - Correct: The inputs represent the caller-controlled service choices, and the string output provides the connection value monitoring needs without exposing internal resource attributes.

4.A Terraform 1.15 root module will create a subnet and an application service in one apply. The service must attach to that subnet. The subnet resource exposes a provider-generated id that is unknown before creation, and the service accepts this ID through its subnet_id argument. The resource blocks are in network.tf and service.tf.

Which design supplies the required value and establishes the service's dependency on the subnet?

A. Set the service's subnet_id argument to a reference to the subnet resource's id attribute.

B. Set the service's subnet_id argument from a root input variable and add depends_on referencing the subnet resource.

C. Set the service's subnet_id argument from a root input variable and rename the files to 01-network.tf and 02-service.tf.

D. Set the service's subnet_id argument from a root input variable and place the subnet block before the service block in main.tf.

**Answer:** A

**Explanation:**

Referencing the subnet resource's id in the service's subnet_id argument supplies the provider-generated value and establishes an implicit dependency. Terraform can track this relationship while the ID is still unknown. Explicit depends_on establishes ordering but does not assign the subnet ID to an input variable. Filenames and block positions do not determine resource creation order. These relationships follow the documented [resource composition] (https://developer.hashicorp.com/terraform/language/v1.15.x/modules/develop/composition) and [resource-order rules] (https://developer.hashicorp.com/terraform/language/v1.15.x/style).

A - Correct: The reference connects the service argument to the ID produced by the subnet resource. It also gives Terraform the dependency needed to create the subnet before creating the service.

B - Incorrect: Explicit depends_on makes the service wait for the subnet, but it does not populate the root

input variable with the subnet's newly generated ID. Ordering alone does not supply the required value.
C - Incorrect: Numeric filename prefixes can organize configuration for readers, but they do not establish resource execution order. The root input variable also lacks a connection to the newly generated subnet ID.
D - Incorrect: Putting the subnet block first improves readability but does not establish a dependency. The separately supplied input variable does not automatically receive the subnet resource's ID.

5.A production service must remain available throughout a replacement triggered by a provider argument change. An engineer adds create_before_destroy = true, and terraform validate succeeds. The platform requires unique names for live service instances, the account has an instance quota, and clients use a separately managed routing entry. Only the validation result has been checked.
What additional evidence is needed to establish the feasibility of meeting the availability requirement?
A. A plan showing creation before destruction, treating that ordering as sufficient evidence for naming, quota, and routing feasibility.
B. Confirmation that names and quota permit both instances to coexist, with the routing entry switched after the current instance is removed.
C. Confirmation that routing can switch before removal, treating successful validation as evidence that names and quota permit coexistence.
D. Confirmation that names and quota permit both instances to coexist and that routing can switch to a healthy replacement before the current instance is removed.
**Answer:** D
**Explanation:**
The availability requirement needs both successful coexistence and uninterrupted client access during replacement. create_before_destroy changes replacement order, but concurrent objects must still satisfy naming and capacity constraints, and traffic must reach a healthy replacement before the current instance is removed. A plan showing that order does not establish all of those conditions. The [lifecycle reference] (https://developer.hashicorp.com/terraform/language/v1.15.x/meta-arguments/lifecycle) documents the coexistence constraints, while the [validate reference] (https://developer.hashicorp.com/terraform/cli/v1.15.x/commands/validate) explains that validation does not check remote services or provider APIs.
A - Incorrect: The plan can show the requested replacement order, but that order alone does not establish compatible concurrent names, spare quota, or a working traffic transition.
B - Incorrect: Naming and quota checks establish whether both instances can coexist. Switching routing only after removing the current instance leaves clients without the required continuous access during the transition.
C - Incorrect: A traffic switch before removal addresses client access, but successful validation does not establish whether the platform permits the proposed names or whether the account has sufficient quota.
D - Correct: This evidence addresses all stated feasibility gaps: permitted concurrent names, capacity for both instances, and client traffic reaching a healthy replacement before the current instance is removed.