TERRAFORM INTERVIEW QUESTIONS

1. What do you understand about Terraform in AWS?

Terraform is a part of the AWS DevOps Competency and also an AWS Partner Network (APN) advanced technology partner. It is similar to AWS Cloud Formation in the sense that it is also an “infrastructure as code” tool that allows you to create, update, and version your AWS infrastructure.

2. What are the key features of Terraform?

Terraform helps you manage all of your infrastructures as code and construct it as and when needed. Here are its key main features:

- A console that allows users to observe functions
- The ability to translate HCL code into JSON format
- A configuration language that supports interpolation
- A module count that keeps track of the number of modules applied to the infrastructure.

3. Define IAC?

IAC or Infrastructure as Code allows you to build, change, and manage your infrastructure through coding instead of manual processes. The configuration files are created according to your infrastructure specifications and these configurations can be edited and distributed securely within an organization.

4. What are the most useful Terraform commands?

Some of the most useful Terraform commands are:

- terraform init - initializes the current directory
- terraform refresh - refreshes the state file
- terraform output - views Terraform outputs
- terraform apply - applies the Terraform code and builds stuff
- terraform destroy - destroys what has been built by Terraform
- terraform graph - creates a DOT-formatted graph
- terraform plan - a dry run to see what Terraform will do

5. Are callbacks possible with Terraform on Azure?

By using the Azure Event Hubs, callbacks are probable on Azure. Terraform’s Azure supplier provides effortless functionality to users. Microsoft Azure Cloud Shell provides an already installed Terraform occurrence.

6. What is Terraform init?

Terraform init is a control to initialize an operational index that contains Terraform pattern files. This control can be looped multiple times. It is the first command that should be run after writing the new Terraform design.

7. What is Terraform D?

Terraform D is a plugin used on most in-service systems and Windows. Terraform init by default searches next directories for plugins.

8. Is history the same as it is on the web while using TFS API to provide resources?

Yes, the narration is similar to on the web because UI keeps API as the base. The whole thing that is on the UI is availed during other methods and the API.

9. Why is Terraform used for DevOps?

Terraform uses a JSON-like configuration language called the HashiCorp Configuration Language (HCL). HCL has a very simple syntax that makes it easy for DevOps teams to define and enforce infrastructure configurations across multiple clouds and on-premises data centers.

10. Define null resource in Terraform.

null_resource implements standard resource library, but no further action is taken. The triggers argument allows an arbitrary set of values that will cause the replacement of resources when changed.

11. What do you mean by Terraform cloud?

Terraform Cloud is a platform that enables teams to use Terraform together, either on-demand or in response to various events. It is deeply integrated with Terraform's workflows and data, unlike a general-purpose continuous integration system. It includes easy access to shared state and secret data, detailed policy controls for updating infrastructure and governing the contents of Terraform, a private registry for sharing Terraform modules, and lots more.

12. Explain Oracle Cloud Infrastructure.

Oracle cloud offered by Oracle Corporation is a cloud computing service providing storage, servers, applications, services, and network through a global network of managed data centers by Oracle Corporation. These services are provisioned on-demand over the Internet by the company.

13. What do you understand about terraform backend?

Each Terraform configuration can specify a backend, which defines two main things:

- Where operations are performed
- Where the state is stored (Terraform keeps track of all the resources created in a state file)

14. What are the version controls supported by Terraform besides GitHub?

The version controls supported GitLab EE, GitLab CE, and Bucket cloud.

15. Name some major competitors of Terraform?

Some of the top competitors and alternatives to Terraform are Azure Management Tools, Morpheus, CloudHealth, Turbonomic, and CloudBolt.

Next up, let us see some intermediate terraform interview questions!

16. Explain the uses of Terraform CLI and list some basic CLI commands?

The Terraform Command-Line Interface (CLI) is used to manage infrastructure and interact with Terraform state, configuration files, providers, etc.

Here are some basic CLI commands:

- terraform init - prepares your working directory for other commands
- terraform destroy - destroys the previously-created infrastructure
- terraform validate - check whether the configuration is valid
- terraform apply - creates or updates the infrastructure
- terraform plan - shows changes needed by the current configuration

17. What are modules in Terraform?

A jug for numerous resources that are used jointly is known as a module in Terraform. The root module includes resources mentioned in the .tf files and is required for every Terraform.

18. What is a Private Module Registry?

A Private Module Registry is a feature from Terraform Cloud that allows you to share Terraform modules across the organization. You can enforce rules or “sentinel policies” on the registry that specify how members of your organization can use the modules.

19. Is Terraform usable for an on-prem infrastructure?

Yes, Terraform can be used for on-prem infrastructure. As there are a lot of obtainable providers, we can decide which suits us the best. All that we need is an API.

20. Does Terraform support multi-provider deployments?

Yes, multi-provider deployments are supported by Terraform, which includes on-prem like Openstack, VMware, and we can manage SDN even using Terram too.

21. How is duplicate resource error ignored during terraform apply?

We can try the following options:

1. Delete those resources from the cloud provider(API) and recreate them using Terraform
2. Delete those resources from Terraform code to stop its management with it
3. Carry out a terraform import of the resource and remove the code that is trying to recreate them

22. Name all version controls supported by Terraform

The supported version controls are:

- Azure DevOps Services
- Azure DevOps Server
- Bitbucket Server
- Bitbucket Cloud
- Gitlab EE and CE
- Gitlab.com
- GitHub Enterprise
- GitHub.com (OAuth)
- GitHub.com

23. What are some of the built-in provisioners available in Terraform?

Here is the list of built-in provisioners in Terraform:

- Salt-masterless Provisioner
- Remote-exec Provisioner
- Puppet Provisioner
- Local-exec Provisioner
- Habitat Provisioner
- File Provisioner
- Chef Provisioner

24. Which command destroys Terraform managed infrastructure?

The given command is used for this purpose:

terraform destroy [options] [dir]

25. Tell us about some notable Terraform applications.

The applications of Terraform are pretty broad due to its facility of extending its abilities for resource manipulation. Some of the unique applications are:

- Software demos development
- Resource schedulers
- Multi-cloud deployment
- Disposable environment creations
- Multi-tier applications development
- Self-service clusters

- Setup of Heroku App

26. What are the components of Terraform architecture?

The Terraform architecture includes the following features:

- Sub-graphs
- Expression Evaluation
- Vertex Evaluation
- Graph Walk
- Graph Builder
- State Manager
- Configuration Loader
- CLI (Command Line interface)
- Backend

27. Define Resource Graph in Terraform.

A resource graph is a visual representation of the resources. It helps modify and create independent resources simultaneously. Terraform establishes a plan for the configuration of the graph to generate plans and refresh the state. It creates structure most efficiently and effectively to help us understand the drawbacks.

28. Can you provide a few examples where we can use for Sentinel policies?

Sentinels are a powerful way to implement a variety of policies in Terraform. Here are a few examples:

- Enforce explicit ownership in resources
- Restrict roles the cloud provider can assume
- Review an audit trail for Terraform Cloud operations
- Forbid only certain resources, providers, or data sources
- Enforce mandatory tagging on resources
- Restrict how modules are used in the Private Module Registry

29. What are the various levels of Sentinel enforcement?

Sentinel has three enforcement levels - advisory, soft mandatory, and hard mandatory.

- Advisory - Logged but allowed to pass. An advisory is issued to the user when they trigger a plan that violates the policy.
- Soft Mandatory - The policy must pass unless an override is specified. Only administrators have the ability to override.
- Hard Mandatory - The policy must pass no matter what. This policy cannot be overridden unless it is removed. It is the default enforcement level in Terraform.

30. How to Store Sensitive Data in Terraform?

Terraform requires credentials to communicate with your cloud provider's API. But most of the time, these credentials are saved in plaintext on your desktop. GitHub is exposed to

thousands of API and cryptographic keys every day. Hence, your API keys should never be stored in Terraform code directly. You should use encrypted storage to store all your passwords, TLS certificates, SSH keys, and anything else that shouldn't be stored in plain text.